Cloth Parameters and Motion Capture

by

David Pritchard

B.A.Sc., University of Waterloo, 2001

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

We accept this thesis as conforming
to the required standard

The University of British Columbia

October 2003

Abstract

Recent years have seen an increased interest in cloth simulation. There has been little analysis, however, of the parameters controlling simulation behaviour. In this thesis, we present two primary contributions. First, we discuss a series of experiments investigating the influence of the parameters of a popular cloth simulation algorithm. Second, we present a system for motion capture of deformable surfaces, most notably moving cloth, including both geometry and parameterisation. This data could subsequently be used for the recovery of cloth simulator parameters. In our motion capture system, we recover geometry using stereo correspondence, and use the Scale Invariant Feature Transform (SIFT) to identify an arbitrary pattern printed on the cloth, even in the presence of fast motion. We describe a novel seed-and-grow approach to adapt the SIFT algorithm to deformable geometry. Finally, we interpolate feature points to parameterise the complete geometry.

Contents

List of Tables

List of Figures

Acknowledgements

I would like to acknowledge my supervisor, Wolfgang Heidrich, for his ideas and discussions related to this thesis. I am grateful for the assistance of several members of my department, including Eddy Boxerman, David Lowe, Robert Bridson and Michiel van de Panne.

For their support, I would like to thank my parents, my brothers, and Kathryn. Finally, I would like to thank the friends who kept me sane during my research, including Dave Burke, Hendrik Kück, Eric Brochu, Lisa Streit, Roger Tam, Ben Forsyth, Ritchie Argue, Matthew Brown, Derek Olive, Kristin Kopra, Alex Aylett, Shuzhen Wang, Matthew Thorne and Dan Archambault.

David Pritchard

The University of British Columbia
October 2003

Chapter 1

Introduction

Over the past several years, interest in computer simulated cloth algorithms has grown steadily. Recent advances in simulation algorithms allow complex and realistic looking cloth to be simulated for use in the motion picture industry. Unfortunately, it remains quite difficult to adjust the parameters of these simulations to match a given real world cloth material.

It may be possible to acquire motion data for moving cloth, and subsequently recover the cloth simulation parameters that best fit this motion data. A recent paper [11] investigated this possibility, but used only 3D geometry data for the recovery. Better simulation parameter recovery would be possible if motion data contained information about stretching, shearing and bending of the cloth, and if occlusions due to folds could be better understood.

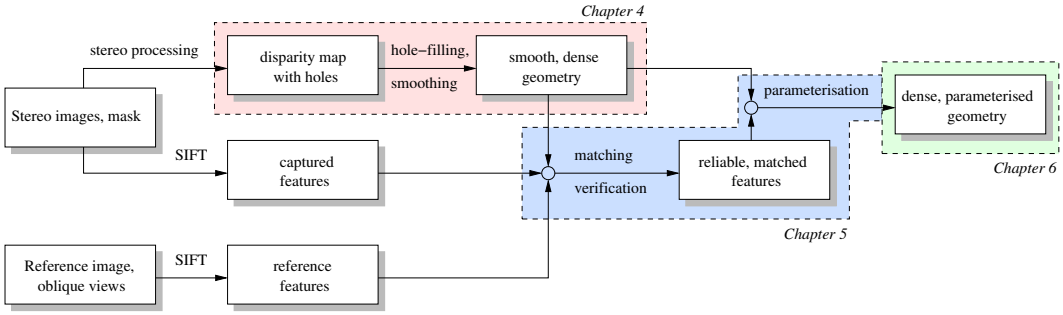


Figure 1.1: Overview of cloth motion capture system.

In this thesis, we present a new method for measuring the 3D geometry and surface parameterisation of moving cloth, captured from a stereo video sequence. Stretching, shearing and bending could be calculated from the data we measure, and it may also be possible to recover cloth simulation parameters.

In Chapter 2, the previous work is discussed including papers in cloth simulation, cloth parameter recovery and cloth motion capture. Chapter 3 contains an analysis of the influence of simulator parameters on a popular cloth simulation algorithm. The subsequent chapters deal with various stages of the new cloth motion capture algorithm, as shown in Figure 1.1. Chapter 4 describes the disparity map construction, Chapter 5 describes the recovery of a surface parameterisation of the cloth, and Chapter 6 demonstrates the results obtained by the cloth motion capture algorithm. The thesis closes with some conclusions and future work in Chapter 7.

Some of this material has been published previously in a different form. Chapters 5 and 6 were originally part of a paper entitled *Cloth Motion Capture* [88] written with Wolfgang Heidrich.

Chapter 2

Previous Work

2.1 Cloth Simulation

Cloth simulation has been a topic of active research in the computer graphics community since it was introduced by Terzopolous et al. [102] in 1987. We refer the reader to Bridson's Ph.D. thesis [18] for an excellent overview of recent papers in the field, and to the second chapter [14] of Breen and House's book [52] for a broader overview of the modelling techniques used in the cloth simulation literature. See also Ng and Grimsdale's survey paper [82] for a more complete review of early cloth simulation methods. Each of these sources has a distinct interpretation, and we personally tend to agree more with Bridson's analysis than with Breen's approach.

Most cloth simulation techniques can be classified into three general groups: elastic deformation models, particle-based models and finite element (or continuum mechanics) models. We refer the reader to [5, 38, 39, 103, 114, 117].

Feynman [42] treated cloth draping using a thin plate flexure model in his 1986 thesis. He modelled cloth as a grid of points, invented his own energy expressions for internal strain and bending resistance and then minimised the energy using a multigrid method.

In their 1987 and 1988 papers [101, 102], Terzopolous et al. considered a range of types of deformable surfaces, including cloth. They gave a broad intro-

duction to the theory of elasticity, and referred readers to texts in the calculus of variations. For their simulation, they used differential equations of motion and an implicit integration scheme, and they discretised the equations using a regular grid.

In 1992, Breen et al. [13, 15, 53] adopted a different formulation for cloth. They minimised energy functions in a manner similar to Feynman, but their energies were not derived from the assumption that cloth is a continuous flexible material, nor did they use elastic shells. Instead, they modelled the static drape of cloth using a set of interacting particles, and insisted that cloth is a *mechanism*, not a continuum. Each particle is intended to represent the crossing-point of warp and weft threads. This interpretation is problematic, since it makes computation intractable if every thread crossing-point must be simulated. In their demonstrations, a large tablecloth was modelled using a small number of particles (51×51), much less than the number needed to represent every thread crossing-point.

One interesting aspect of Breen et al.'s system was their consideration of material properties [16, 17]. They used a Kawabata [66] system to measure the physical properties of real cloth, and used these material properties directly in their cloth simulation model. They observed that shearing behaviour (which they called *trellising*) was responsible for much of the differences between materials, and for maintaining constant surface area. Later work with DeVaul [34, 54] extended the Breen algorithm to include dynamic motion. Finally, this group has also had extensive contact with the textile research community, including publication in journals in that field.

Members of MIRAlab have studied the problem of generating clothes for virtual actors over the last decade. The earliest contribution of researchers at MIRALab was Carignan et al.'s synthesis of a full set of clothing on a digital character in a 1992 paper [25], which was a first for computer graphics at the time. This paper described collision processing and garment panel stitching, and used Terzopolous' dynamics with added damping forces. In 1995, Volino et al. [107] represented the

cloth mechanics using an elastic deformation model, but removed topological restrictions on the model by allowing an unstructured triangle mesh; a later 1997 paper [108] simplified their approach. More recent papers from the lab have covered a range of topics, including a comparison of integration methods [110], collision detection [109] and many other subjects [30, 47, 111].

In a 1995 paper, Provot [89, 90] used a mass-spring model for cloth, which he correctly identified as a variant of the elastic deformation models. Provot made the important observation that traditional methods of cloth simulation produce a "rubbery" appearance in the cloth motion. To get around this, most users set a high spring stiffness, but explicit integration of the resulting stiff system is slower. He introduced an *ad hoc* method to keep spring lengths constant, greatly reducing the elastic appearance of cloth motion. Provot also treated collision response in a later paper [91].

Researchers at the University of Tübingen have been studying cloth since the mid-1990s. Eberhardt et al. [37] improved Breen's method to include cloth motion. A later paper by Eberhardt [35] studied implicit-explicit schemes for animation using particle systems, while papers by Hauth et al. [49, 50, 51] compared the performance of a range of numerical methods and preconditioners. In an upcoming paper, Etzmuß et al. [40] establish a connection between continuum mechanics and particle systems. Finally, Etzmuß et al. [41] and Mezger et al. [79, 80, 81] studied collision response in some detail, and Eberhardt et al. [36] studied knit fabrics.

More recently, a 1998 paper by Baraff and Witkin [6, 7] returned to the approach of Terzopolous et al., and introduced a new, more efficient technique for animating cloth and clothing. One major contribution of their scheme was an implicit integration scheme using a conjugate gradient solver. Since Terzopolous' work, researchers had used explicit integration schemes, leading to either "rubbery" cloth movement or a stiff system that was hard to solve, as observed by Provot. By using an implicit scheme, Baraff and Witkin were able to take large timesteps and avoid

"rubbery" looking cloth. Baraff and Witkin's system also included position and velocity constraints on individual nodes, and they followed Volino's lead by using an unstructured triangular mesh. Their force model was based on energy functions, with bend forces calculated using the dihedral angle between adjacent triangles. They simulated cloth with more nodes than previous work, demonstrating one example with 8800 vertices. Unfortunately, their bending model and integration scheme required the introduction of nonphysical damping terms to reduce instability, and their system tended to remove fine wrinkles and geometric detail in the cloth.

Baraff and Witkin later extended [8] their system to solve the problem of cloth tangles when self-intersection occurs in character animation. Ascher and Boxerman [4] improved the efficiency of Baraff and Witkin's integration scheme, and demonstrated the convergence of their modified conjugate gradient method. Desbrun et al. [32, 33, 78] made major approximations to Baraff and Witkin's model to achieve realtime performance with a coarse cloth discretisation, and Kang et al. [61, 62, 63, 64, 65] made further approximations for the sake of speed.

Choi and Ko [27] addressed the issue of buckling in cloth. In their 2002 paper, they observed that the behaviour of cloth is quite different under compressive and tensile forces. Cloth strongly resists a tensile force, acting like a stiff spring to preventing stretching. Under a compressive load, however, it prevents compression by buckling. Choi and Ko used a heuristic model of the post-buckling shape of cloth, and introduced a more complicated spring model to separate compressive and tensile forces. They used an implicit integration scheme for efficient computation, but unfortunately required a structured rectangular mesh. A recent paper [28] addressed this limitation by extending their model to triangular meshes, and discussed instabilities in traditional formulations of shear forces.

Bridson et al. [19] initially tackled cloth collision processing. Most of their 2002 paper dealt with collision detection and response, and also with friction. To process the collisions, they also introduced a new integration scheme similar to cen-

tral time differencing. They also adopted and corrected some of Provot's techniques.

In a recent 2003 paper [20], Bridson et al. treated problems more directly related to cloth simulation, focusing particularly on the preservation of wrinkles. They noted that fine tessellation of the cloth surfaces required small timesteps when damping forces were treated with an explicit integration scheme. To resolve this, they proposed an implicit integration scheme for damping forces combined with an explicit integration scheme for internal cloth dynamics, which also avoided the excessive damping of wrinkles seen in Baraff and Witkin's work. In the same paper, Bridson et al. introduced a new bending model, based on the angle between adjacent triangles in a manner similar to Baraff and Witkin. Much of their simulation model is similar to Baraff and Witkin's approach, except for a separated treatment of compression and tensile forces, an approach first used by Choi and Ko. They also treated further details of collision processing.

There have been a wide range of other interesting papers touching on cloth simulation. Van Gelder et al. [104, 105] examined the relation between mass-spring models and elasticity theory. Parks and Forsyth [85] studied the generalised-α integration scheme. Hutchinson et al. [56], Zhang et al. [116], Volkov et al. [112] and Villard et al. [106] have studied adaptive refinement techniques. Aono et al. [1, 2, 3] studied the novel problem of "dart insertion" in the aerospace industry, where cloth must precisely fit a particular surface shape without wrinkles. Li et al. [70, 71, 72, 73, 74] studied the motion of cloth in an air flow. Xu et al. [113] used a specialised rod-based model model to simulate curtain motion. Romero et al. [92, 93, 94] and Lario et al. [67, 68] considered parallel algorithms for cloth simulation, and Zara et al. [115] looked at solving cloth simulation using a cluster. Fuhrmann et al. [43], Mezger et al. [79, 80, 81] and Huh et al. [55] all studied collision detection in cloth simulation.

2.2 Simulation Parameter Recovery

Louchet et al. [75] were the first to attempt to recover cloth simulation parameters from motion data. They assumed that some unspecified vision system would provide their input data, which consisted of 3D positions of gridpoints on the cloth; for their tests, they used synthetic data. They used a genetic algorithm to solve an optimisation problem, yielding five unknown cloth simulation parameters. Their cloth simulator was a simple mass-spring system, and their input data was essentially a combination of geometric and parametric (fixed gridpoint) information.

Jojic [57, 58, 59, 60] considered static draped cloth, whose shape is captured via a range scanner. The cloth range data was treated as a solid surface, and a cloth simulation was run to drape the cloth over the range data. An external force was applied to attract the cloth to the range data. The process was then repeated using different parameters until a solution was found, using an optimisation approach. For the cloth simulation, he used a simple mass-spring model. His error metric in the optimisation process included only distance from the range data.

Jojic's approach had several limitations. First, he treated only static simulation parameters and did not consider dynamic movement of cloth, although his approach might be extended to include dynamic behaviour. Second, he only made use of the cloth's geometry, and did not incorporate surface parameterisation data, apart from the manually detected corners of the cloth. Third, he made several assumptions about the cloth drape, including a known drape configuration (e.g., known underlying geometry). Finally, his results were mostly synthetic and were not very convincing, and he did not attempt to work with a range of fabrics.

Bhat et al. [11] adopted the same general framework as Jojic, using an optimisation technique to fit simulation parameters to range data. Their input was a video sequence under structured light, and they did consider dynamic behaviour, such as damping parameters. Their error metric was based on the fit to the video data, using both silhouette matching and a thresholded "angle map" representing re-

gions of high curvature or discontinuities. They used the cloth simulation algorithm introduced by Baraff and Witkin [6], but were forced to use explicit integration to avoid excessive damping. They also noted some problems with scale invariance of parameters in Baraff and Witkin's model, and suggested a new model for air drag forces.

Like Jojic, Bhat et al. did not make use of surface parameterisation in their optimisation, although they did present some useful data about the optimisation space. They did study a range of real-world fabrics, and also tested the generalisation of their measurements by synthesising new motion using the estimated parameters. Finally, they presented a comparison of complex synthesised motion and real cloth movement, although the results were not very convincing.

The effectiveness of Bhat et al.'s error metric is very unclear. In Figure 3 of their paper, they show angle map error as a function of the bend and stretch parameters, varied over one order of magnitude. In this figure, error seems to be almost independent of the bend parameter, implying that they cannot reliably measure this parameter. In Figures 8 and 9, they have a good match between the simulated cloth and the captured video of static cloth. In Figure 10, however, the match is much poorer for moving cloth.

It is difficult to evaluate the quality of the results obtained by either Jojic or Bhat et al. Both show clear discrepancies between the physical data and the simulation, but it is hard to determine whether these errors are due to their own methods, or due to the cloth simulation system used in their inner simulation loop. Bhat et al. describe some problems with the damping model used by Baraff and Witkin's cloth simulation, and similar issues have been discussed at length in the cloth simulation community.

Both Jojic and Bhat et al. relied only on 3D geometric data. This data should contain information about folds and wrinkles in the cloth, but cannot capture local internal deformation such as stretching and shearing. Consequently, they were

able to measure the relative strength of bend forces, but should have had difficulty determining the strength of stretch and shear forces. Naturally, stretch and shear forces will have some impact on the global shape and drape of the cloth, but much of the effects will be hard to detect using only geometric data. Finally, without parametric information it is very difficult to deal with occluded regions of the cloth. Near a fold, it is impossible to tell how much fabric is contained in the occluded region without parametric information.

Consequently, our measurement of both geometric and parametric data is a valuable contribution to the cloth parameter measurement problem.

2.3 Cloth Capture

The computer vision community has shown interest in recovering the motion of non-rigid surfaces, such as cloth.

The work of Guskov et al. [44, 46] is most relevant to the problem of cloth parameterisation. Guskov used cloth with a checkered pattern printed on it, and he tracked the corners of the quads to recover sparse parametric information. In a recent paper [45], he extended his earlier work to include 3D reconstruction of the surface geometry using multiple cameras. His system was targeted at real-time performance, and could not capture a large number of feature points. His accuracy degraded as the size of the quads drops below 20×20 pixels. Furthermore, he could not use stereo correspondence to recover detailed geometry, since his checkered pattern lacks high-frequency texture.

Torresani et al. [103] used rank constraints on optical flow to track features and reconstruct 3D geometry from monocular video data. They showed some promising results for human torso, face and shoe motion, but did not attempt the more challenging problem of cloth tracking. Their tracking approach would still require a recognition phase to form an initial correspondence between features and parametric (u, v) positions.

Carceroni [23, 24] attempted the extremely difficult task of recovering 3D shape, reflectance and non-rigid motion of a dynamic scene using a surfel model. His results were quite promising, demonstrating the tracked position and velocities of a sheet of patterned cloth under reasonably quick motion. His approach dealt reasonably well with difficult problems such as occlusion. Like Torresani et al., Carceroni's approach would still require an initial recognition phase to establish the starting (u, v) correspondence.

Each of these methods suffers from one common failing: a relatively small number of features is tracked. Our approach tracks an order of magnitude more features than any of the previous work. Finally, all of the previous work tackled the motion *tracking* problem, while our approach is more of a *recognition* method. As a result, we do not need any temporal history to allow recognition of features in an arbitrary deformed state.

There have been a number of less directly relevant efforts at cloth capture. Haddon and Forsyth [48] tracked the formation of folds and wrinkles in video sequences, but did not recover geometry or parameterisation. Ruiz and Buxton [95] attempted to improve the capture of fine wrinkles in static cloth geometry, but ignored parameterisation.

Chapter 3

Cloth Simulation Experiments

As part of our investigation of cloth simulation parameters, we wrote a cloth simulator. We implemented the simulator described by Baraff and Witkin [6], and released the software publicly with the name Freecloth. In Section 3.1, we discuss some corrections to Baraff and Witkin's paper. Using this simulator, we investigated the influence of each of the simulator's parameters. Details of this are discussed in Section 3.2.

3.1 Cloth Simulator

For our experiments, we used a simulator based on the approach of Baraff and Witkin. However, we modified their approach to include corrections made by other researchers, as well as improvements based on our own insights. Our implementation of Baraff and Witkin's algorithm included Ascher and Boxerman's corrections [4] to the modified preconditioned conjugate gradient algorithm, as well as their improved preconditioner.

Bhat et al. [11] noted issues with Baraff and Witkin's air drag formulation, and suggested corrections. We did not implement these corrections to air drag in our own cloth simulator. They also observed problems with excessive damping using the first-order implicit Euler time integration scheme used by Baraff and Witkin.

We note here a few corrections of our own. In particular, Baraff and Witkin's parameters are not scale-invariant. If a mesh is subdivided while all parameters are kept constant, the shape of the cloth changes substantially, beyond what would be expected through the simple introduction of a few edges for bending.

Baraff and Witkin defined forces in terms of what they called *condition functions* ($\mathbf{C}$), which are closely related to energy (E):

$$E = \frac{k}{2}\mathbf{C}^T\mathbf{C} \tag{3.1}$$

They defined separate energies for stretching, shearing and bending. They presented a correct argument for keeping the bend condition function independent of triangle area, but defined the stretch and shear condition functions to be directly proportional to triangle area. Consequently, the stretch and shear energies are proportional to the square of the triangle area. If a triangle is refined in a standard 1-to-4 split, the total energy of the four new triangles will be one quarter of the original energy. Since bend and gravity energies are independent of tessellation, this makes stretch and shear forces relatively weak at higher tessellations, and dramatically changes the shape of the cloth drape as the cloth is refined.

We recommend modifying Baraff and Witkin's condition functions to be proportional to the square root of the triangle area, making the energies directly proportional to triangle area. The stretch condition function in Baraff and Witkin's equation (10) should be changed to use the square-root of the area,

$$\mathbf{C}(\mathbf{x}) = \sqrt{a}\begin{pmatrix} ||\mathbf{w}_u(\mathbf{x}) - b_u|| \\ ||\mathbf{w}_v(\mathbf{x}) - b_v|| \end{pmatrix}$$

and the shear condition in section 4.3 of their paper should be likewise be changed,

$$C(\mathbf{x}) = \sqrt{a}\mathbf{w}_u(\mathbf{x})^T\mathbf{w}_v(\mathbf{x}).$$

This will make the cloth simulation parameters independent of mesh tessellation. Bhat et al. independently made the same observation.

Finally, in equation (11) of Baraff and Witkin's paper, damping forces are calculated using $\dot{\mathbf{C}}(\mathbf{x})$, which they defined as

$$\dot{\mathbf{C}}(\mathbf{x}) = \left(\frac{\partial \mathbf{C}(\mathbf{x})}{\partial \mathbf{x}} \right)^T \dot{\mathbf{x}}.$$

Correct partial differentiation of $\dot{\mathbf{C}}(\mathbf{x})$ should give

$$\dot{\mathbf{C}}(\mathbf{x}) = \sum_m \left(\frac{\partial \mathbf{C}(\mathbf{x}_m)}{\partial \mathbf{x}_m} \right)^T \dot{\mathbf{x}}_m,$$

where m ranges over all particles influencing $\mathbf{C}$. This was likely the authors' intent, but it is unclear (and even a little misleading) in their paper.

3.2 Influence of Parameters

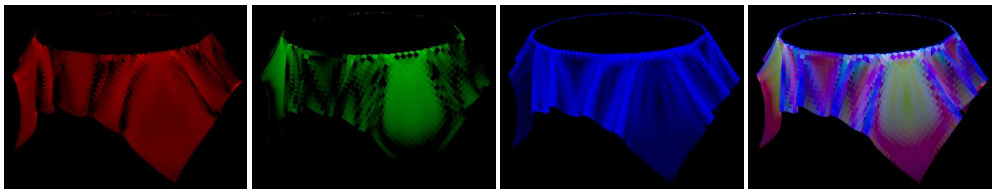
For some cloth simulation algorithms [17], the relationship between the simulator's parameters and the mechanical behaviour of real cloth has been well-established. However, Baraff and Witkin's simulator has not been subjected to such study, and it is not intuitively obvious how the simulator's parameters affect cloth motion, or how they interact. We set out to investigate the effect of the individual parameters on cloth movement.

Bhat et al. [11] are the only authors to present plausible parameter values for Baraff and Witkin's cloth simulator. The base values used in our simulation are shown in Table 3.1. Bhat et al. used a different drag model, and did not use the MPCG algorithm or adaptive time stepping requiring a stretch limit. They did report stretch, shear and bend resistances and damping parameters that were within an order of magnitude of the parameters we used.

To evaluate the influence of the individual parameters, a series of simulations was performed. The setup involved the draping of a tablecloth over a circular table. The cloth was modelled as a square sheet of 66×66 square patches with a regular split into triangles. In each experiment, only a single parameter was varied, and a number of statistics were recorded, including the appearance of the cloth and the

Parameter	Symbol	Value
Stretch resistance	k_{st}	100
Shear resistance	k_{sh}	10
Bend resistance	k_b	1.0×10^{-5}
Stretch damping	d_{st}	20
Shear damping	d_{sh}	2
Bend damping	d_b	2.0×10^{-6}
Air drag	k_a	0.1
MPCG tolerance		0.01
stretch limit		5.0×10^{-5}

Table 3.1: Parameter values for the modified scale invariant version of Baraff and Witkin's simulator.



(a) Stretch energy (b) Shear energy (c) Bend energy (d) Combined

Figure 3.1: Stretch, shear and bend energies are combined as the red, green and blue channels of a single image to visualise the overall energy distribution during cloth simulation.

spatial distribution of stretch, shear and bend energy in the cloth. In the results that follow, spatial energy distributions are visualised by encoding the logarithm of the stretch, shear and bend energies in the red, green and blue channels of the image respectively, as shown in Figure 3.1.

The experiments were limited to a single drape configuration, and only one parameter was changed at a time. Consequently, we can draw few substantial conclusions about the general behaviour of Baraff & Witkin's cloth model; we limit this section to our observations.

3.2.1 Number of patches

The effect of changing the discretisation of the cloth surface is demonstrated in Figure 3.2. In this particular tablecloth example, it appears that 66×66 patches is

a sufficiently fine discretisation for a reasonable simulation. Both the final drape and the energy profile are very similar for the 66×66 case and the much finer 132×132 example.

It is interesting to observe the changing energy profile in lower tessellation examples. When the surface was discretised coarsely, the cloth was very limited in its ability to bend, and was consequently forced to shear substantially. With finer discretisations, bending rose slightly and shearing dropped dramatically. The only real difference between the finest discretisations was a slight change in bending behaviour.

Clearly, the appropriate level of discretisation is highly application-dependent. A discretisation of 66×66 was sufficient for this tablecloth, but might be a poor choice for a complicated piece of clothing with finer wrinkles and many regions of high curvature.

3.2.2 Timestep

Large timesteps are known to introduce numerical damping, as demonstrated here. In this experiment a fixed timestep was used, and the damping impact can be seen in the mid-swing pose of the cloth shown in Figure 3.3 and also in the energy graph. Baraff and Witkin's goal of using large timesteps will clearly also force the cloth motion to be damped.

In terms of cloth parameter recovery, this result is quite significant. It implies that cloth parameters can only be reused in other simulations operating at the same timescale—if at all. Clearly, this behaviour needs further study if cloth parameter recovery is to be practical.

When using adaptive timestepping, varying amounts of damping were introduced into the system as the timestep grew and shrank, making it difficult to compare results between different trials. To reduce the effect of this behaviour, timesteps were kept small (5 ms) in the experiments that follow.

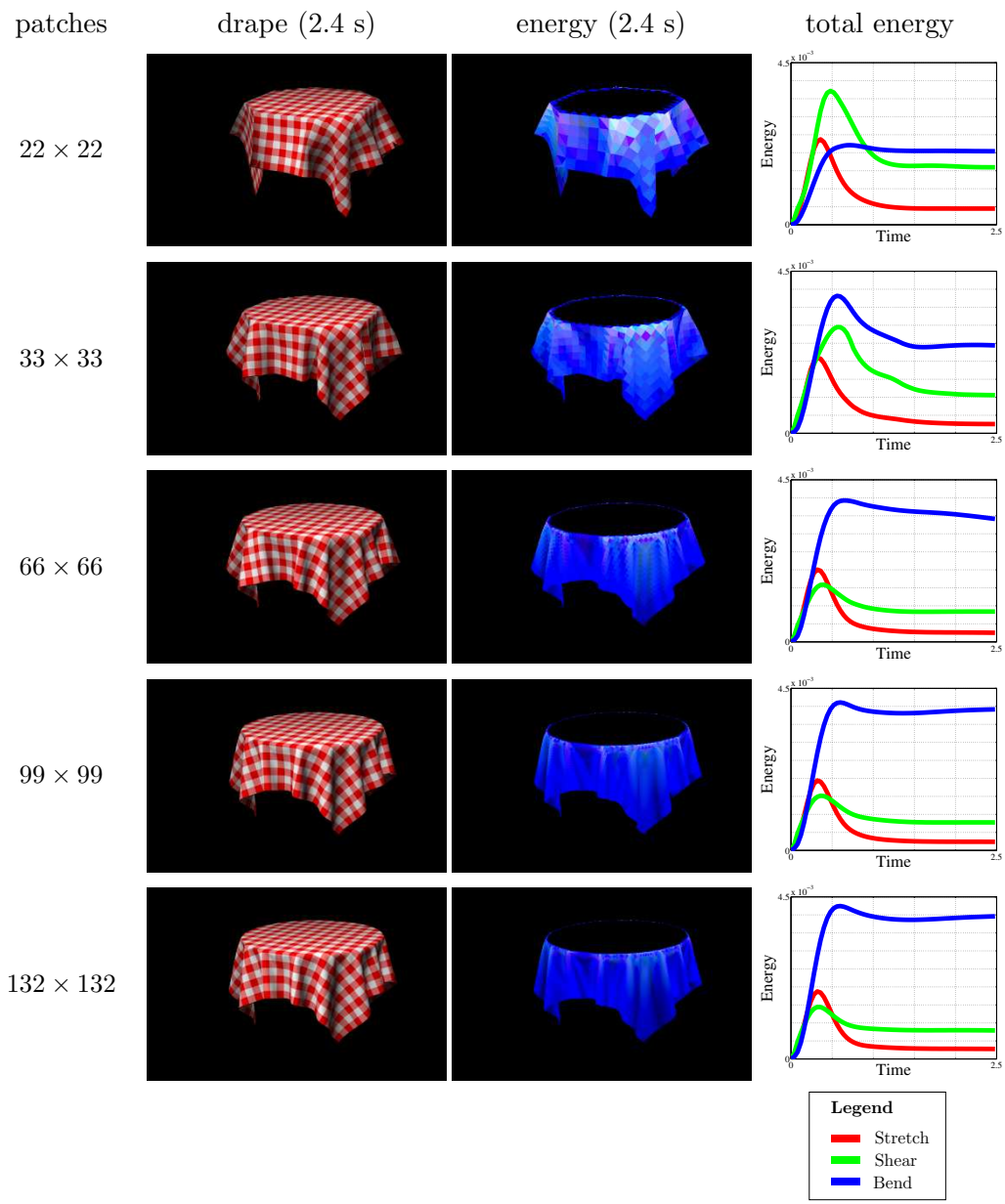


Figure 3.2: Impact of discretisation on tablecloth drape and energy distribution.

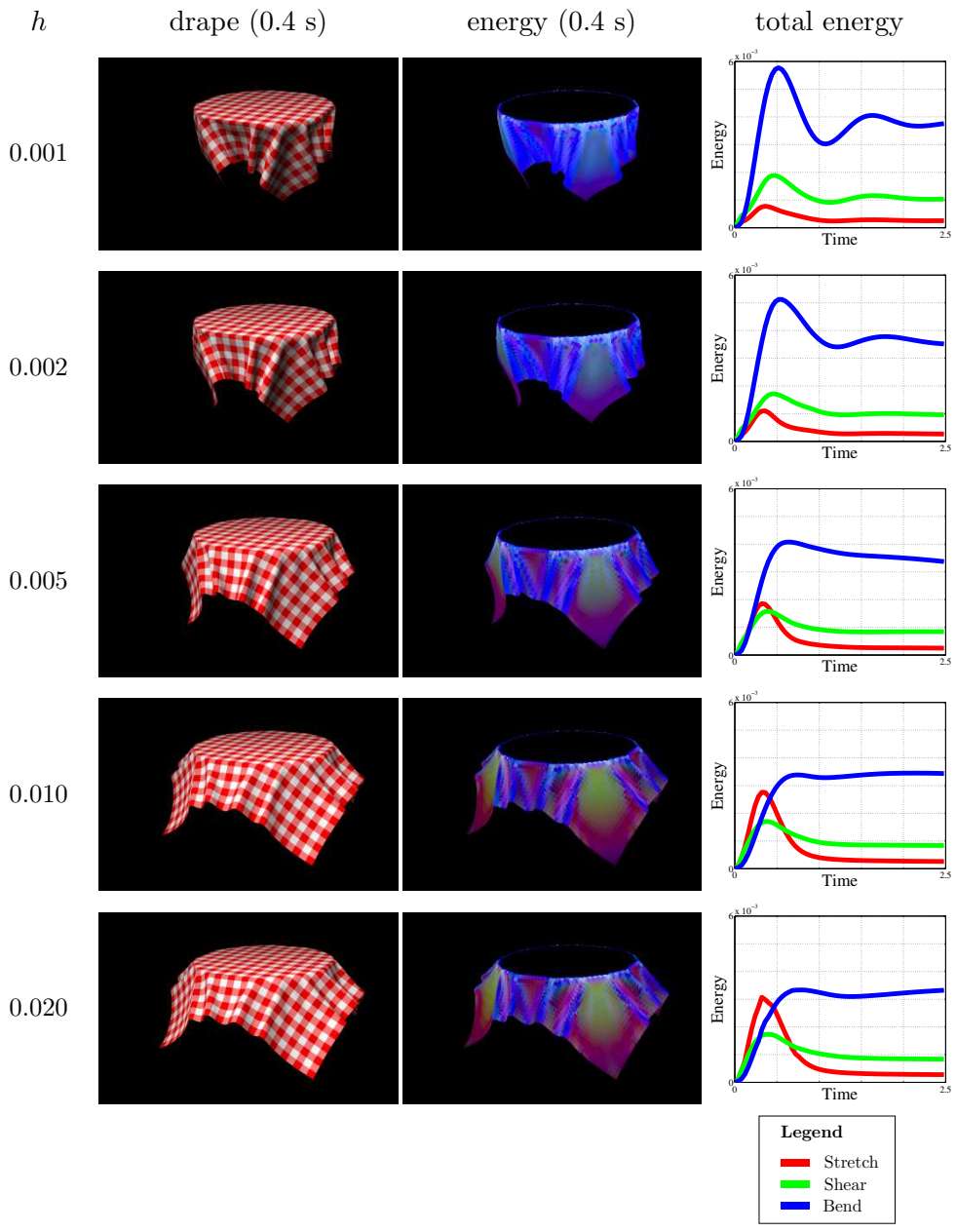


Figure 3.3: Impact of timestep on drape and energy distribution.

3.2.3 Stretch, shear and bend resistance

The effect of changing the cloth's resistance to stretch, shear and bend is demonstrated in Figures 3.4–3.6.

Changes to the stretch resistance had little effect on the movement of the cloth. The final drape and the transient motion were essentially unaffected over a wide range of values, although the total stretch energy stored in the cloth did change. With higher stretch resistance, stretch energy reached equilibrium rapidly, and less stretch energy was stored in the cloth in the transient régime.

Modifying the cloth's shear resistance had a more visible effect. With low shear resistance, the cloth's final drape showed more sag, and the shearing was quite evident in the cloth texture. When shear resistance was low, more shear energy was stored in the cloth and the cloth swung freely. When the shear resistance was higher, the cloth motion appeared highly damped.

Changes to bend resistance obviously influenced the cloth's shape. With high bend resistance, relatively few wrinkles formed, and the bends that did form stored a large amount of bend energy.

3.2.4 Damping constants

From our observations of both real and simulated cloth, the only aspect of cloth motion that was visibly damped was bending. Both stretching and shearing behaviour seemed to be overdamped, while bending behaviour was either underdamped or overdamped, depending on the cloth material.

In the tablecloth draping experiments, the cloth either settled slowly on the table or else fell quickly onto the table and swung back and forth several times before settling to a steady state. This corresponded to overdamped and underdamped behaviour, respectively. In the energy graphs, this can generally be seen by looking for ringing behaviour in the total bend energy.

The impact of the stretch damping constant was fairly minimal. Subtle

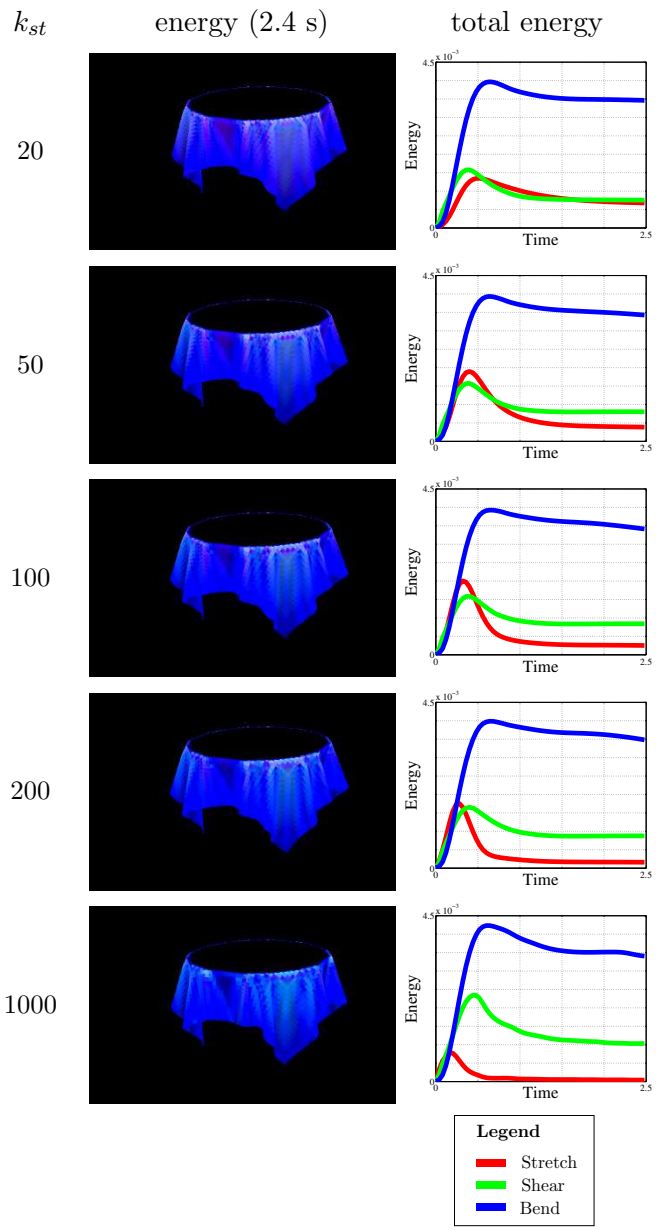


Figure 3.4: Impact of stretch resistance on energy distribution.

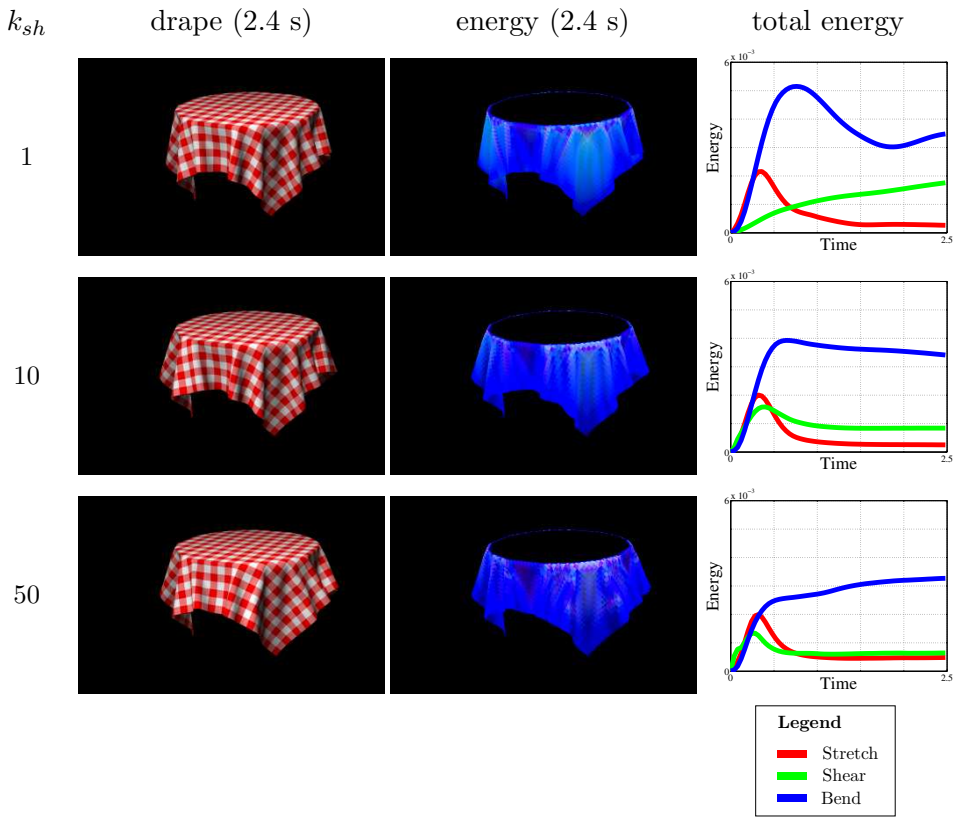


Figure 3.5: Impact of shear resistance on drape and energy distribution.

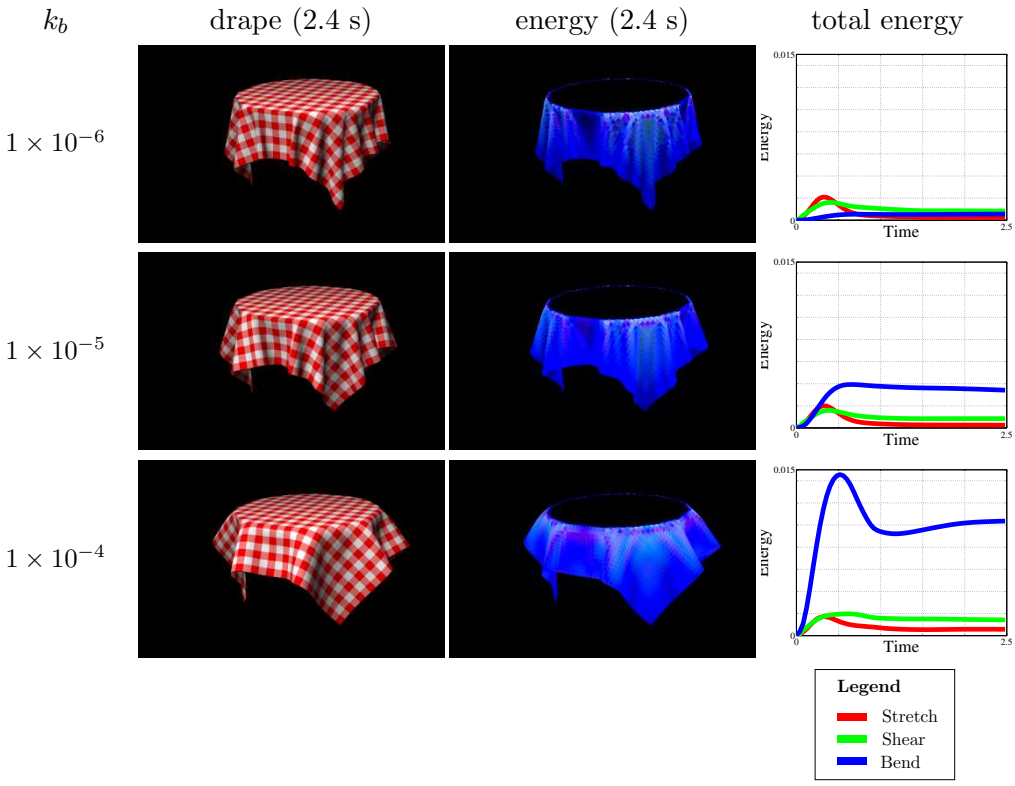


Figure 3.6: Impact of bend resistance on drape and energy distribution.

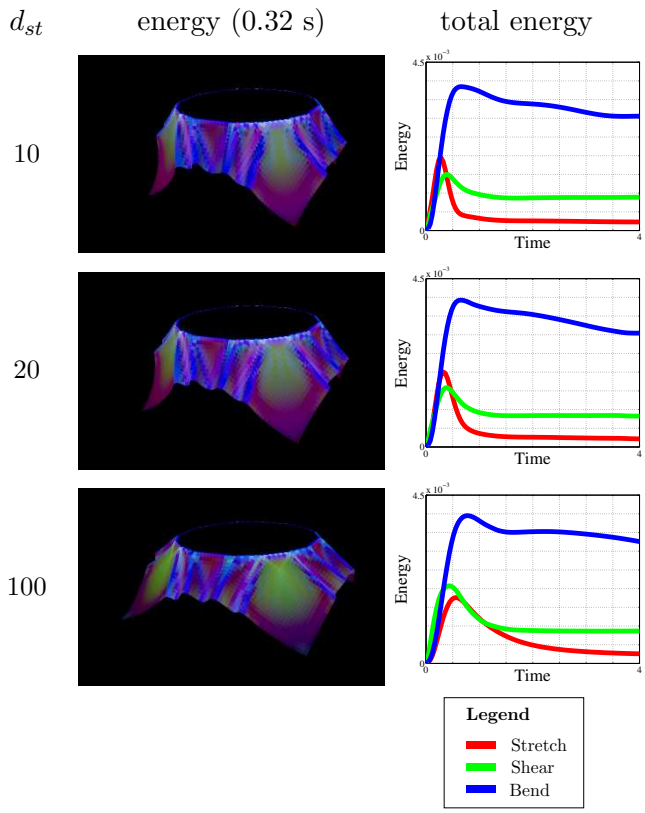


Figure 3.7: Impact of stretch damping constant on energy distribution.

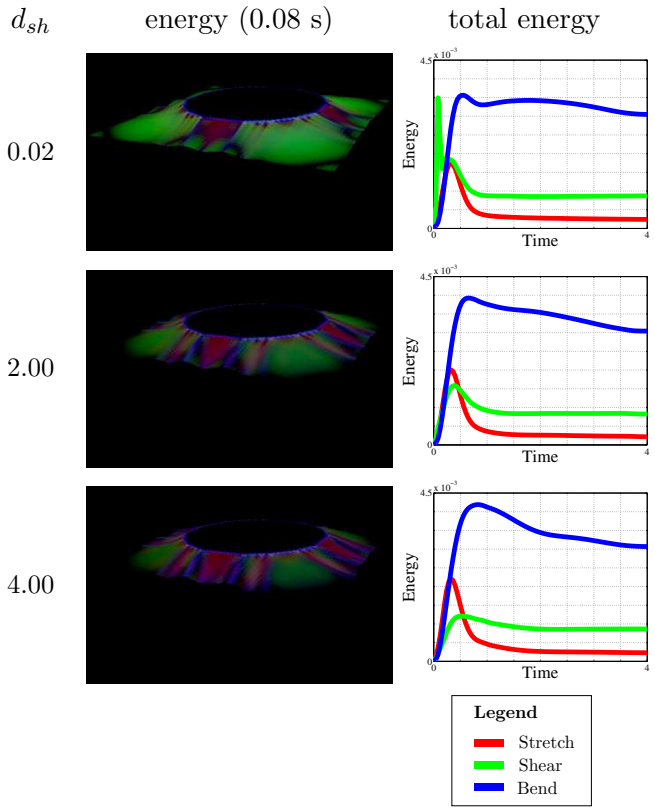


Figure 3.8: Impact of shear damping constant on energy distribution.

damping in the motion of the cloth was evident during the first second of movement, but the final drape was unaffected. As shown in Figure 3.7, the energy graph shows changes to the transient energy distribution, with obvious damping effects in the stretch energy as the damping constant rose. Curiously, with very low stretch damping (e.g., $d_{st} \leq 2$), the simulation could not be computed.

The shear damping constant also had only a minor effect on the cloth's behaviour. The cloth's final drape and motion were unaffected by changes to this constant. The energy graph shows predictable underdamped and critically damped behaviour in the shear energy, but the transient behaviour was brief enough that it had no major effect on the cloth's motion. Figure 3.8 shows the cloth energy distribution near the beginning of the simulation.

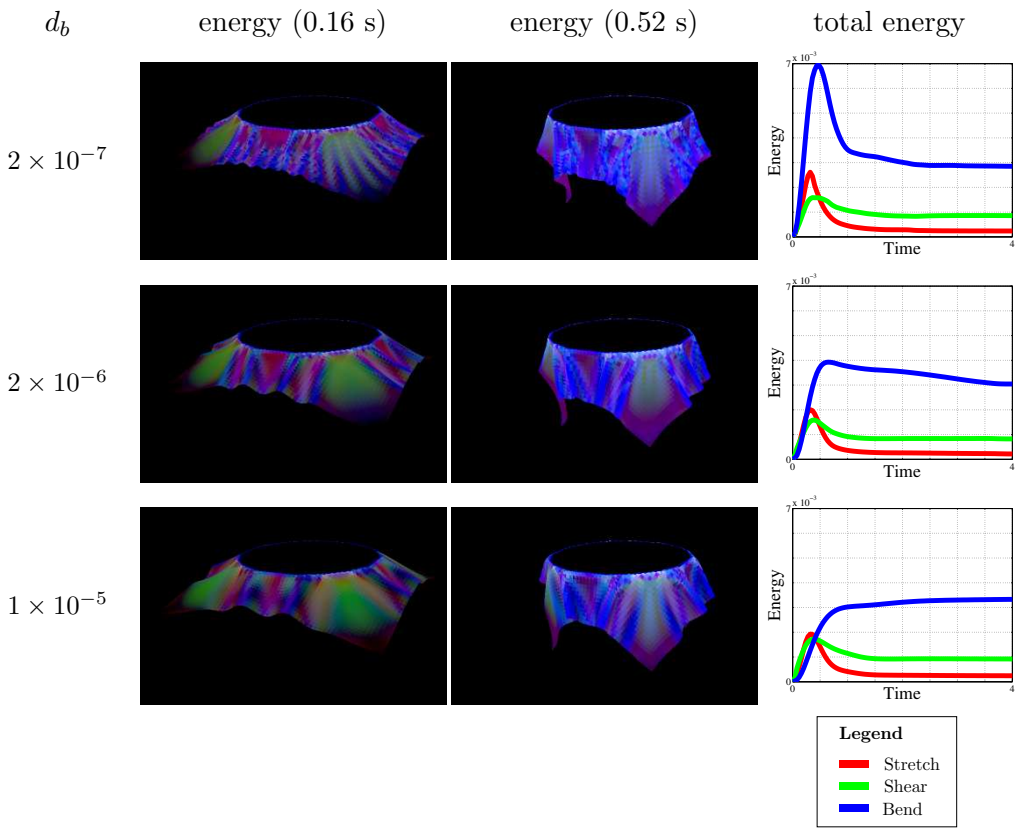


Figure 3.9: Impact of bend damping constant on energy distribution.

The bend damping constant had a very visible effect on the cloth's movement. As shown in the energy graphs in Figure 3.9, the bend energy was quite different as the constant was changed, and this could also be seen in the cloth's movement, particularly at the corners. The damping was predictable and followed a typical underdamped/overdamped form, but there was also some interesting smoothing. When the bending damping constant was low, fine wrinkles formed in the cloth during the early transient motion, while damping prevented these wrinkles from forming when the bending damping constant was higher. The final drape position was the same in all cases, however.

Chapter 4

The Disparity Map

As described in the introduction, the bulk of this thesis addresses the issue of cloth motion capture. In this chapter, some of the details of the first stage of the cloth motion capture system are discussed, covering the construction of a disparity map from input multibaseline stereo images.

The output of this stage of our system is three images of equal size: a rectified greyscale camera image of the cloth and backdrop; a mask to distinguish the cloth from the backdrop; and a disparity map, from which the depth at every pixel can be inferred. The greyscale image and disparity map can be generated with a standard stereo vision system, and the mask can be easily defined using background subtraction.

Rectified images and epipolar geometry are a well-understood subject in computer vision. Given a suitable system for camera calibration, it is easy to produce rectified images [84]. Stereo correspondence algorithms take two (or sometimes more) rectified greyscale images as input, and produce a disparity map $d(x, y)$ for each pixel in one of the input images, typically stored as a greyscale image. Figure 4.1 demonstrates this process, although the disparity map shown here is somewhat idealised. The term *disparity* was originally used to describe the 2D vector between the positions of corresponding features seen by the left and right eyes. It

is inversely proportional to depth, and it is possible to define a mapping from an (x, y, d) triple to a three-dimensional position.

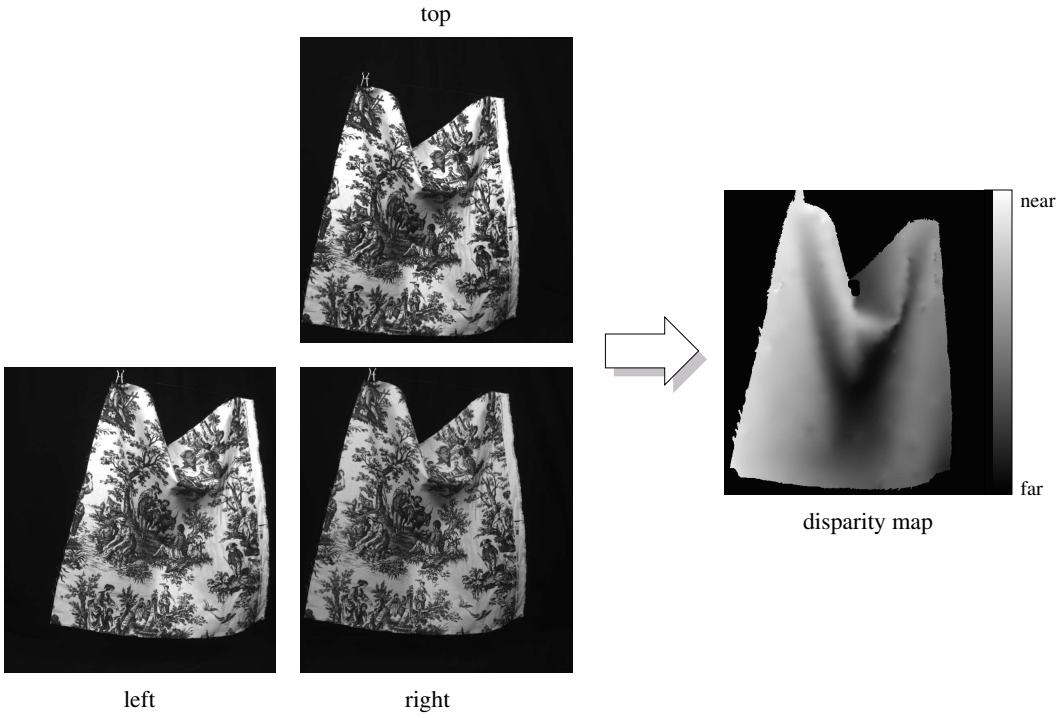
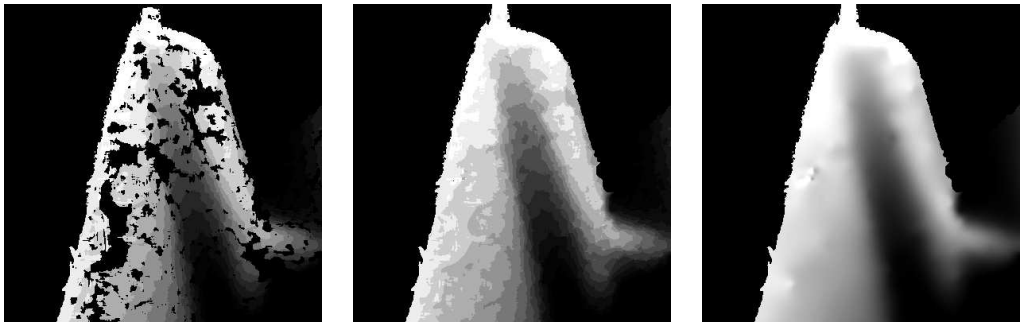


Figure 4.1: Stereo correspondence algorithms take two (or more) rectified images as input and produce a disparity map.

There are a wide range of stereo correspondence algorithms. We refer the reader to the excellent survey by Scharstein and Szeliski [97, 98, 99] for a taxonomy of the available techniques. We used the Sum of Absolute Differences (SAD) correlation method to reconstruct disparity maps. This is a very simple approach with a number of major artefacts, and in the remainder of this chapter we discuss our solutions to the shortcomings of SAD correlation. However, it should be noted that a more sophisticated stereo correspondence algorithm (such as the graph cuts approach) might be a more suitable solution, and could be easily substituted.

The SAD correlation method yields three major types of artefacts. First, in some regions disparities are uncertain, and are left as "holes" in the disparity map,

as demonstrated in Figure 4.2(a). Uncertainty can occur for a variety of reasons, including insufficient texture, depth discontinuities or noisy images.



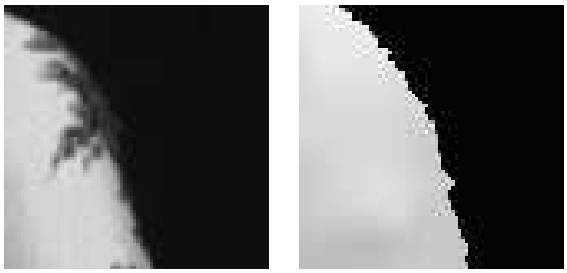
(a) (b) (c)

Figure 4.2: (a) original disparity map with holes; (b) hole-filled integer disparity map; (c) after sub-pixel estimation. Intensity levels have been exaggerated to emphasise quantisation.

Second, most disparity maps are only computed to integer precision, i.e. $d(x, y) \in \mathbb{Z}$. When these disparities are inverted to obtain depth, the resulting depthmap is visibly quantised, yielding a very jagged surface. Some algorithms attempt to calculate a fractional part for each disparity using *sub-pixel estimation*, but such techniques are still tentative and can produce incorrect results [100]. An example of the errors corrected by sub-pixel estimation is shown in Figure 4.2(b).

Third, window-based stereo correspondence algorithms often exhibit a "foreground fattening" effect near depth discontinuities between two objects. When this happens, samples from the far object are mistakenly measured as having the same disparity as samples on the near object, as demonstrated in Figure 4.3.

The stereo system we used is prone to all three of these problems. We have developed a technique that smoothly fills holes and finds a fractional part to each disparity to create a smoother surface, but we have no way to solve the problem of foreground fattening. The fractional part is not measured from the input images, as in the traditional sub-pixel estimation algorithms used by the vision community, but is instead smoothly interpolated from the measured integer disparities.



(a) (b)

Figure 4.3: Demonstration of the foreground fattening effect: (a) input image, with foreground cloth on left and black background. (b) disparity map produced by stereo correspondence algorithm.

We do not claim that our solution is a novel contribution to the field; we merely document our approach in the interest of thoroughness. Our solution is adapted to the particular needs of the stereo system we used, but it is likely possible to find an existing stereo system that exhibits none of these problems. In the future, we expect that standard stereo systems will solve these problems, and output from a stereo system can be used directly without the modifications described here.

We have the option of operating on either the two-dimensional disparity map, or the corresponding three-dimension surface. Given the structured nature of our input data, we choose to operate directly on the disparity map in a two-dimensional manner for the sake of efficiency.

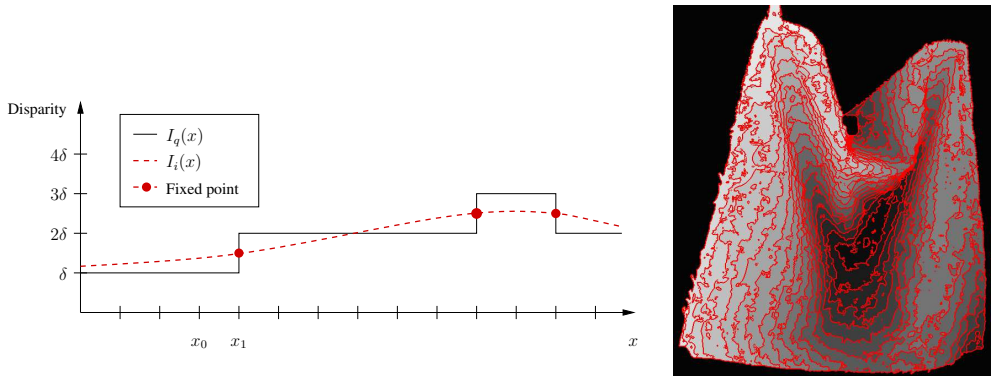
4.1 The PDE Approach

Both hole-filling and sub-pixel estimation can be formulated as image interpolation problems. Image interpolation is an image-based technique that involves filling holes in an image with plausible data. The hole is not necessarily filled with smooth data, but may sometimes involve extending discontinuities at the hole edge into the hole. It has been well studied by researchers such as Bertalmio et al. [10], Caselles et al. [26] and Pérez et al. [86], and is also studied as part of the larger problem of

image inpainting. The general approach involves fixing the boundary of the hole, and then solving a boundary-value partial differential equation (PDE) to interpolate the interior of the hole. This is equivalent to applying a diffusion process, such as isotropic diffusion or one of the many forms of anisotropic diffusion. For more details on isotropic and anisotropic diffusion, refer to Black et al. [12]. For a detailed description of the relation between diffusion and partial differential equations, see Sapiro's excellent book [96].

Caselles et al. also considered a different problem. They started with a quantised image, i.e. only a limited set of intensity values, such as multiples of 30. From this, they wished to produce a smooth image with integer intensity values using an image interpolation algorithm. This would also produce a satisfactory solution to our sub-pixel estimation problem; the disparity map can be viewed as a quantised image, and smooth interpolation of the data would be a satisfactory solution.

Caselles' approach can be explained using a one-dimensional example, as shown in Figure 4.4. Suppose that the image $I(x)$ is quantised to image $I_q(x)$ by rounding image intensities to the nearest integer multiple of δ. We aim to interpolate $I_q(x)$ to form a smooth interpolated image $I_i(x)$. Consider two adjacent points, x_0 and x_1 at a step edge, $I_q(x_1) = I_q(x_0) + \delta$. Clearly, the intensity of the unquantised image I crossed the mid-intensity point $I_q(x_0) + \frac{\delta}{2}$ somewhere between x_0 and x_1. We can take advantage of this and force the high point of each step edge down to the mid-intensity point, i.e. fix $I_i(x_1) = I_q(x_0) + \frac{\delta}{2}$. Finally, these fixed points are interpolated. A special case is required for step edges larger than δ, in which case both the high and low side of the edge are fixed. Extension to a two-dimensional image is straightforward, with the fixed points forming closed Jordan curves in the plane. Caselles called these the *boundaries of the level sets*, but it should be noted that his level sets are not directly related to standard levelset methods in computer graphics [83].



(a) (b)

Figure 4.4: (a) one-dimensional example of Caselles' approach, with black showing quantised disparity map and red showing disparity map after interpolation using diffusion; (b) boundaries of level sets in two-dimensional case.

We implemented Caselles' approach, using a simpler isotropic diffusion process for both hole-filling and interpolation of the quantised disparity map. An isotropic diffusion process finds a solution to

$$\nabla^2(I) = 0, \tag{4.1}$$

which is also known as *Laplace's equation*. Laplace's equation is a special case of *Poisson's equation*, and is a classic example of an elliptic PDE. In this formula, the ∇^2 operator is the Laplacian, where

$$\nabla^2(I) = \text{div}(\nabla I).$$

Implementation of this approach was quite straightforward. The PDE was solved by using an explicit integration scheme, with

$$I(t + \Delta t) = I(t) + \Delta t \frac{\partial I}{\partial t}$$

where

$$\frac{\partial I}{\partial t} = \nabla^2 \left(I(t) \right).$$

and $\Delta t = 1$. To improve performance, a hierarchical approach was used, solving first on a low-resolution image, then using those results as the starting point for diffusion on a higher-resolution image.

Initially, the results from this approach seemed reasonable. Performance was quick, requiring only about two minutes per frame. Holes were smoothly filled, showing C^0 continuity and C^1 continuity with the fixed hole boundary. Quantisation artefacts were resolved, with C^0 continuity. However, there were C^1 continuity problems at the fixed boundaries of the level sets used to interpolate the quantised disparity map. The diffusion process did yield C^1 continuity on either side of the levelset boundary line, but the derivatives on either side of the boundary were not identical. Consequently, the image gradient had clearly visible discontinuities along the boundaries of the level sets.

Ultimately, more control was needed at the boundaries during diffusion. Like Caselles, we fixed the value of the function at the boundary, known in the PDE literature as imposing *Dirichlet conditions* on the boundary value partial differential equation. *Cauchy conditions* are a standard alternative, where both the value and the gradient are specified at the boundary as described in [87]. Unfortunately, Cauchy conditions cannot be used, since the gradient at the boundary is unknown; we only want the gradient on either side of the levelset boundaries to be the same. It is possible that the biharmonic equation could provide this control over the system, but we leave this as future work.

4.2 The Optimisation Approach

PDE methods were satisfactory for hole-filling, but could not solve the sub-pixel estimation problem. Using a diffusion-based approach, the only way to interpolate the existing data to perform sub-pixel estimation was by introducing fixed boundaries of the level sets. However, the interpolation can be achieved in a different manner.

As before, a solution to Laplace's equation (Equation 4.1) is desired, starting

from a variant,

$$\nabla^2(I + \Delta I) = 0, \tag{4.2}$$

with I held constant and solving for ΔI. A strict interpolation of the quantised data can be achieved by constraining ΔI to lie between $-\frac{\delta}{2}$ and $\frac{\delta}{2}$. Under this constraint, an exact zero solution to Laplace's equation may not be found, but the equation can be solved in a least-squares sense. In other words, find ΔI that minimises

$$\sum_{x,y} \left[\nabla^2 \left(I_{x,y} + \Delta I_{x,y}\right)\right]^2 \tag{4.3}$$

subject to $-\frac{\delta}{2} \leq \Delta I_{x,y} \leq \frac{\delta}{2}$. This is an optimisation problem, not a partial differential equation problem.

We use a finite-difference representation of the Laplacian,

$$\nabla^2(I + \Delta I) = (I + \Delta I)_{x,y-1} + (I + \Delta I)_{x,y+1} + (I + \Delta I)_{x-1,y} + (I + \Delta I)_{x+1,y} - 4(I + \Delta I)_{x,y}. \tag{4.4}$$

With a little juggling and remapping of indices, this can be expressed as a linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b}.$$

The image ΔI is flattened to form the vector $\mathbf{x}$. Suppose that ΔI is w pixels wide by h pixels high. Then, the first w entries of $\mathbf{x}$ are filled with the top row of ΔI, followed by the second row, and so on. The constant coefficients of ΔI are placed in the $\mathbf{A}$ matrix, and the constant I values are placed in vector $\mathbf{b}$.

As shown in Figure 4.5 the matrix $\mathbf{A}$ is sparse with the standard 5-point Laplacian structure, sometimes known as "tridiagonal with fringes." The main diagonal represents the central count in the finite-differencing scheme, the upper and lower diagonals correspond to the right and left neighbours respectively, and the fringe diagonals correspond to the lower and upper neighbours. The value a_i on the main diagonal is initially adjusted to ensure that the sum of each row is zero.

$$\begin{bmatrix}
-a_1 & 1 & & & & 1 & & & & & \\
1 & -a_2 & 1 & & & & 1 & & & & \\
 & 1 & -a_3 & 1 & & & & \ddots & & & \\
 & & 1 & \ddots & \ddots & & & & 1 & & \\
 & & & \ddots & -a_i & \ddots & & & & & 1 \\
1 & & & & \ddots & -a_{i+1} & \ddots & & & & \\
 & 1 & & & & \ddots & \ddots & 1 & & & \\
 & & \ddots & & & & 1 & -a_{n-2} & 1 & & \\
 & & & 1 & & & & 1 & -a_{n-1} & 1 & \\
 & & & & 1 & & & & 1 & -a_n &
\end{bmatrix}$$

Figure 4.5: General structure of **A** matrix used by optimisation approach for sub-pixel estimation. The neighbour count a_i is equal to the sum of the other elements of the row (typically 4).

The problem can now be expressed in a simple, classic form. Find $\mathbf{x}$ that minimises

$$||\mathbf{Ax} - \mathbf{b}||, \tag{4.5}$$

subject to $-\frac{\delta}{2} \leq \mathbf{x}_i \leq \frac{\delta}{2}$

This is a constrained linear least-squares minimisation problem. We use Matlab to solve this equation, and Matlab uses a subspace trust-region method based on the interior-reflective Newton method, as described by Coleman et al. [29].

Some small modifications are made to this scheme for practical purposes. Not all entries in I are valid, and this must be accounted for. Invalid samples are excluded from both the **A** and **b** matrix. We adjust the central count a_i to include only the number of valid neighbour samples. Finally, for hole-filling, the boundary values surrounding the hole are included in **b** but excluded from **A**.

In practical terms, this algorithm performs quite slowly. A hierarchical version performs much better, and some minor edits to Matlab's source files can improve performance to about three minutes per frame. The appearance of the results

is satisfactory, exhibiting both C^0 and C^1 continuity.

As noted before, the optimisation approach is only needed for sub-pixel estimation. Both the PDE and optimisation approaches produce satisfactory results for hole-filling, and the PDE approach yields better performance.

Chapter 5

Parameterisation

The parameterisation of the cloth surface follows several stages, similar in principle to stages in many computer vision systems. First, features are detected in the intensity image. Each feature is then matched with features in a flat reference image of the cloth. The global structure of the parameterisation is analysed, and invalid features are rejected. Finally, parameter values are interpolated for every pixel in the input image.

5.1 Feature Detection

For feature detection, we use Lowe's Scale-Invariant Feature Transform (SIFT) [76, 77]. Features detected using SIFT are largely invariant to changes in scale, illumination, and local affine distortions. Each feature has an associated scale, orientation and position, measured to subpixel accuracy. Features are found at edges using the scale-space image gradient. Each feature has a high-dimensional "feature vector," which consists of a coarse multiscale sampling of the local image gradient. The Euclidean distance between two feature vectors provides an estimate of the features' similarity. Lowe used SIFT features for the *object recognition* task, and considered only rigid objects with a very small number of degrees of freedom. See Brown and Lowe's 2002 paper [21] for an example of object recognition. An upcoming paper

by the same authors [22] uses SIFT for image registration in panorama stitching, a very different problem. We make heavy use of SIFT features, but we must adapt the matching to the parameterisation of cloth, a deformable surface with a very high number of degrees of freedom.

We detect features in two different images. A scan of the flattened cloth is used to obtain the *reference* image, a flat and undistorted view of the cloth. We use the 2D image coordinates of points in the reference image directly as (u, v) parameters for the features. This 2D parametric *reference space* is denoted $\mathcal{R}$. The second image is the input intensity image, called the *captured* image here. We refer to this 2D image space as the *capture space*, and denote it $\mathcal{C}$.

We also work in *world space* $\mathcal{W}$, the three-dimensional space imaged by the stereo system. Capture space is a perspective projection of world space, and the disparity map provides a discretised mapping from capture space to world space. We map disparity values at discrete locations back to world space and use linear interpolation to obtain a continuous mapping. Finally, we also work in the *feature space* $\mathcal{F}$. This is a 128-dimensional space containing the SIFT feature vectors for both the reference and the captured features.

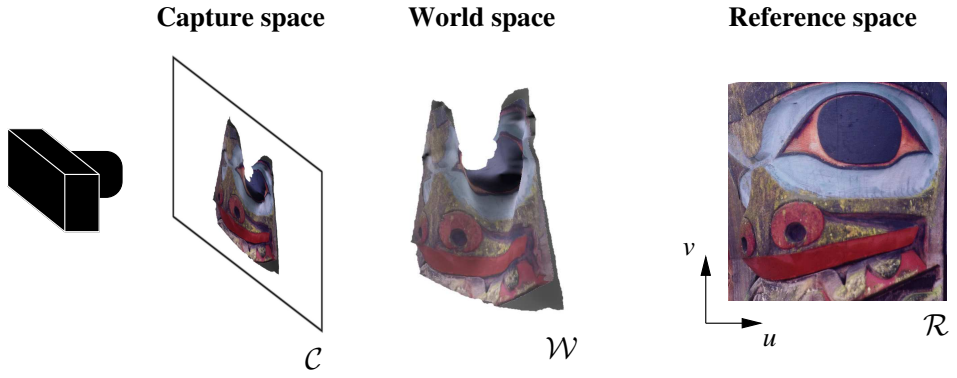


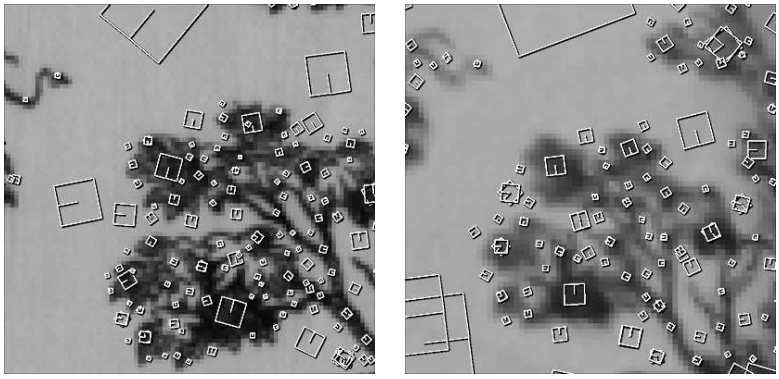
Figure 5.1: Capture space $\mathcal{C}$ is an image of 3D world space $\mathcal{W}$. Reference space $\mathcal{R}$ is a flattened view of the cloth.

After applying SIFT to the reference and captured images, we obtain two

sets of features,

$$\mathbf{F}_r = \{r \,|\, \mathbf{p}(r) \in \mathcal{R}, \mathbf{f}(r) \in \mathcal{F}\}$$
$$\mathbf{F}_c = \{c \,|\, \mathbf{p}(c) \in \mathcal{C}, \mathbf{f}(c) \in \mathcal{F}\}$$

where $\mathbf{p}(x)$ is the position of feature x within the image, and $\mathbf{f}(x)$ is the feature vector associated with x. Each feature also has an associated scale $\mathbf{s}(x) \in \mathbb{R}$. An example of these feature sets is shown in Figure 5.2.



(a) (b)

Figure 5.2: (a) reference feature set $\mathbf{F}_r$; (b) captured feature set $\mathbf{F}_c$.

If we can establish a one-to-one mapping between reference features and captured features, then we know both the world space position and the reference space position of every captured feature, allowing parameterisation. In the matching stage of the algorithm described in Section 5.2, we construct this one-to-one mapping, which we label $\Phi : \mathcal{C} \rightarrow \mathcal{R}$. It should be noted that a one-to-one mapping is only feasible if the pattern in the reference image has no repetitions.

Cloth strongly resists stretching, but permits substantial bending; folds and wrinkles are a distinctive characteristic of cloth. This behaviour means that sections of the cloth are often seen at oblique angles, leading to large affine distortions of features in certain regions of the cloth. Unfortunately, SIFT features are not invariant to large affine distortions.

To compensate for this, we use an expanded set of reference features. We generate a new reference image by using a 2×2 transformation matrix $\mathbf{T}$ to scale the reference image by half horizontally. We repeat three more times, scaling vertically and along axes at $\pm 45°$, as shown in Figure 5.3. This simulates different oblique views of the reference image. For each of these scaled oblique views, we collect a set of SIFT features. Finally, these new SIFT features are merged into the reference feature set. When performing this merge, we must adjust feature positions, scales and orientations by using $\mathbf{T}^{-1}$. This approach is compatible with the recommendations made by Lowe [77] for correcting SIFT's sensitivity to affine change.

Figure 5.3: Top row: a reference image and a horizontally scaled oblique view. Bottom row: other oblique views.

5.2 Matching

The Euclidean distance in $\mathcal{F}$ given by $||\mathbf{f}(r) - \mathbf{f}(c)||$ is the simplest metric for finding a match between a reference feature $r \in \mathbf{F}_r$ and a given captured feature $c \in \mathbf{F}_c$. Unfortunately, in our tests with cloth this metric is not sufficient for good matching, and tends to produce a sizable number of incorrect matches.

We would like to enforce an additional constraint while performing feature

matching. The spatial relationship between features can help to eliminate bad matches: any pair of features that are close in reference space must have matches which are close in capture space. The converse is not always true, since two nearby captured features may lie on opposite sides of a fold. If we could enforce this capture/reference distance constraint during the matching process, we could obtain better results.

We can extend this notion by thinking about distances between features in world space. Suppose that we have complete knowledge of the cloth surface in world space (including occluded areas), and can calculate the geodesic distance in $\mathcal{W}$ between two captured features $c_s, c_n \in \mathbf{F}_c$:

$$\Delta d_c = g\left(c_s, c_n\right). \tag{5.1}$$

Now, consider two reference features $r_s, r_n \in \mathbf{F}_r$, which are hypothetical matches for c_s and c_n. We know the distance in $\mathcal{R}$ between r_s and r_n, but we do not know the distance between them in $\mathcal{W}$. By performing a simple calibration step, we can establish a scalar multiple relating distances in these two spaces. We will multiply by α_r to map a distance from $\mathcal{R}$ to $\mathcal{W}$, and multiply by α_r^{-1} for the opposite mapping.

Using α_r, the world space distance between the reference features can be calculated.

$$\Delta d_r = \alpha_r \cdot ||\mathbf{p}(r_s) - \mathbf{p}(r_n)|| \tag{5.2}$$

We will use these two distances to define the *compression constraint* and the *stretch constraint*:

$$\Delta d_r(1 - k_s) < \Delta d_c < \Delta d_r(1 + k_s) \tag{5.3}$$

where k_s is a constant defining the maximum allowable stretch.

We refer to the lower bound on Δd_c as the compression constraint, and the upper bound is called the stretch constraint. If $\Delta d_c > \Delta d_r(1+k_s)$, then this choice of

match implies that the captured cloth is very stretched; similarly, if the compression constraint is violated, then this choice of match implies that the captured cloth is very compressed. Provided that a reasonable choice is made for k_s, we can safely reject matches that violate the stretch constraint or the compression constraint. Figure 5.4 illustrates these constraints.

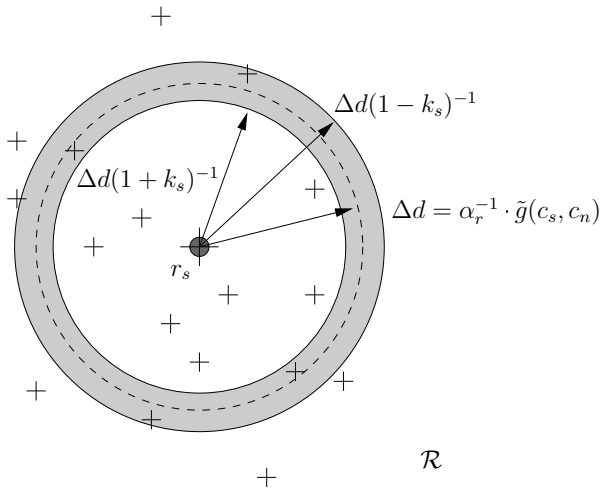


Figure 5.4: If we fix two captured features c_s and c_n and one reference feature r_s, the stretch and compression constraints require the remaining reference feature to lie in a ring centred on r_s. The ring's inner and outer radii are derived from Equations 5.2 and 5.3.

In our real-world setting, finding the geodesic distance between captured features is more difficult. In situations where the entire cloth surface between c_s and c_n is visible, we define a straight line between c_s and c_n in $\mathcal{C}$, project this line onto the surface in $\mathcal{W}$, and integrate along the line. This will not find the geodesic distance, but will closely approximate it.

$$\Delta d_c = \tilde{g}(c_s, c_n) \tag{5.4}$$

While this tends to overestimate $g(c_s, c_n)$, it is still preferable to computing the actual geodesic distance, which is prohibitively expensive.

In some situations, sections of the cloth surface on the geodesic line between

c_s and c_n will be occluded. We can detect such situations using the same line integration method as before, scanning for discontinuities in depth along the line. When occlusion occurs, there is no way of estimating the actual geodesic distance $g(c_s, c_n)$. However, we can still use $\tilde{g}(c_s, c_n)$, which in this case is likely to be an underestimate of $g(c_s, c_n)$. The stretch constraint can be applied to these features, but we cannot use the compression constraint, since the amount of fabric hidden in the fold is unknown at this point.

In contrast to the distance metric in feature space, the stretch and compression constraints are applied to pairs of matched features. To accommodate this, we adopt a seed-and-grow approach. First, a small number of seeds are selected, and these seeds are then matched using only the feature space distance metric. For each seed, we "grow" outwards in capture space, finding nearby features and matching them. As we find features, we can use a nearby pre-matched feature to enforce the stretch constraint.

5.2.1 Seeding

The seeding process is straightforward. We select a small subset of captured features $\mathbf{F}'_c \subset \mathbf{F}_c$, and find matches for them in a brute force manner. For each $c \in \mathbf{F}'_c$, we compare against the entire reference feature set $\mathbf{F}_r$, and we use the feature-space distance between c and $r \in \mathbf{F}_r$ to define the quality of a match. To improve the speed of the brute force matching, we use Beis & Lowe's *best bin first* algorithm [9]; this is an approximate search in a k-d tree. (It is approximate in that it always returns a close match, but not always the best match possible.) We then sort $\mathbf{F}'_c$ by the feature-space distance, and apply the growth process on each seed in order, from best-matched to worst. The growth process classifies captured features into three sets: matched, rejected and unknown. If a seed fails to grow, the seed itself is classified as rejected. After all seeds have been grown or rejected, we construct a new $\mathbf{F}'_c$ from the remaining unknown captured features.

To help the process, we prefer captured features with a large SIFT scale $\mathbf{s}(c)$ when selecting $\mathbf{F}'_c$. In the first iteration, $\mathbf{F}'_c$ consists of the largest features, followed by a smaller group, and so on until a minimum scale is reached. Large features are only found in relatively flat, undistorted, and unoccluded regions of the cloth. In these regions, the growth process will be able to proceed rapidly without encountering folds or occlusions, rapidly reducing the number of unknown features. This rapid growth reduces the number of features which must be considered as seed candidates. The use of the seeding process should be reduced as much as possible, since it cannot make use of the stretch and compression constraints, and hence must resort to relatively inefficient and unreliable brute force matching.

5.2.2 Growing

The growth process is controlled with a priority queue. Each entry in the priority queue is a matched *source feature* $c_s \in \mathbf{F}_c$ on the edge of the growth region. The queue is sorted by capture space distance from the seed, ensuring an outward growth from the seed. The queue is initialised with the seed point alone. The source features are extracted from the queue one at a time.

Let us consider one such source feature, consisting of c_s and $r_s = \Phi(c_s)$. To grow outwards, we iterate over all features c_n in the neighbourhood $N(c_s)$ of c_s in capture space. $N(c_s)$ is a circle of radius r_c centred on c_s. For a given c_n, the match candidates are the reference space features which pass the stretch and compression constraints. These candidate features lie in a ring around r_s, as shown in Figure 5.4.

To select the best match among the match candidates, we use the feature space distance $||\mathbf{f}(c_n) - \mathbf{f}(r_n)||$ for each candidate r_n. The closest match is accepted, provided that the distance in $\mathcal{F}$ is below a threshold.

The growth process requires knowledge of neighbouring features in capture space, and neighbours within a ring in reference space. We efficiently retrieve these neighbours by performing binning in a preprocessing stage.

5.3 Verification

The growth algorithm enforces constraints during the matching process, but it only works with two features at a time. A feature matched by the seed-and-grow process may be acceptable when compared with one of its neighbours, but it may be clearly incorrect when all neighbours are examined. During the growth process, however, it is difficult to perform any global verification, since information about the cloth is sparse and incomplete. After the seed-and-grow algorithm has completed, we can verify the accuracy of matches. At this stage, we will only reject bad matches, and will not attempt to make any changes to $\Phi(c)$.

We attempt to correct two types of errors in the matching process. In the following, we will refer to the features matched during growth from a single seed as a *seed group*. A *feature error* occurs within a seed group, when a few isolated features in the group are badly matched but the bulk of the group is valid. A *seed error* occurs when a bad seed is accepted, in which case the entire seed group is invalid. We propose a three-stage solution to deal with these errors.

The stages are very similar, so we describe the general operation first. We operate on the Delaunay triangulation of the captured features, and we use a voting scheme to determine the validity of features or seed groups. One vote is assigned to each outwards edge. For a feature, every incident edge is used; for a seed group, every edge connecting a seed group feature to a different seed group is used. The vote is decided by evaluating the stretch and compression constraints on the edge. Finally, we calculate a mean vote for each feature or seed group, and reject the features or seed groups with the poorest mean vote. We repeat the process until all features or seed groups pass a threshold mean vote.

In the first stage of verification, we operate on each seed group in turn, and consider only feature errors within that seed group. Subsequently, we consider only seed errors between the seed groups. Finally, we do a repeat search for feature errors, this time operating on the entire set of remaining features. Typically, this

final stage helps to eliminate bad features at the edge of the seed groups.

The entire verification process could be formulated as a simulated annealing algorithm. This would have the benefit of a better theoretical grounding; a continuous measure of error instead of a pass/fail threshold; and it would be easier to extend to include different types of errors. A simulated annealing scheme might also be suitable for correcting interframe errors, improving temporal coherence. This is left as future work.

5.4 Geometry Parameterisation

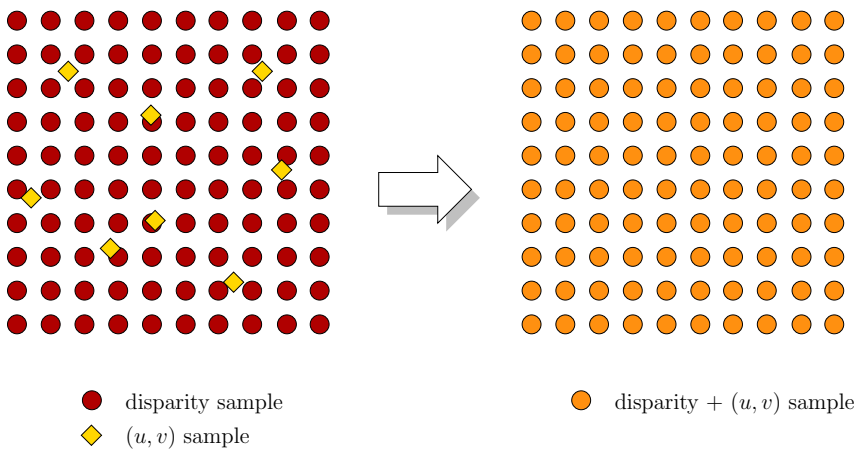


Figure 5.5: After verification, we have dense, regular disparity samples and sparse irregular (u, v) samples. We interpolate the (u, v) samples to achieve a uniform regular sampling of both geometry and parameterisation.

After verification, we are left with a set of reliable features, and a dense, regularly sampled disparity map, as shown in Figure 5.5. We would like to construct a unified representation that contains both 3D and parametric data, sampled in the same pattern. We choose to interpolate the parametric information given by the features to construct a dense, regularly sampled parametric map corresponding directly to the disparity map.

An interpolation in capture space is not sufficient, as demonstrated in Fig-

ures 5.6 and 5.7. As can be seen, linear interpolation in capture space leads to unacceptable distortions on the surface in world space. Instead, what is needed is linear interpolation along the surface (the arc in Figure 5.6). This must be extended from the one-dimensional example in the figure to a surface.

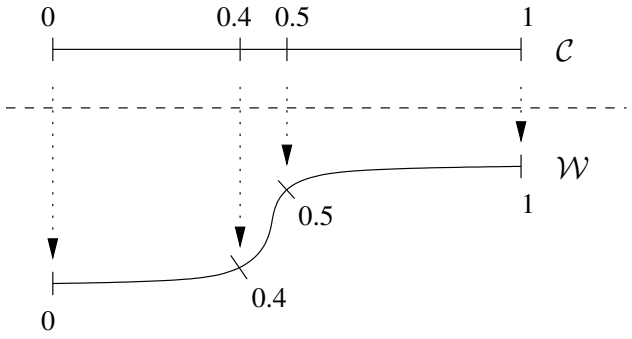


Figure 5.6: Example where linear interpolation of parameter values in $\mathcal{C}$ results in distortion of parameters when projected into $\mathcal{W}$.

This problem is similar in principle to the non-distorted texture mapping problem described by Lévy and Mallet [69] and others. Their technique enforced two primary constraints, perpendicularity and constant spacing of isoparametric curves on the surface. These goals are unfortunately not the same as our own: we desire constant spacing of isoparametric curves, but we would like to allow non-perpendicularity. In the language of the cloth literature, little or no stretch is permitted, while shearing may take place. Our problem is therefore subtly distinct from many of the standard problems in non-distorted texture mapping or mesh parameterisation.

First and foremost, we aim to perform a pure interpolation, retaining the parameterisation at all feature points. We choose to operate on individual triangles within the capture space Delaunay triangulation of the feature points. Within each such triangle the goal, like Lévy & Mallet, is to have constant spacing of isoparametric curves. We make no guarantees of C^1 or C^2 continuity across triangles.

Our interpolation scheme is recursive, and operates on a triangle mesh in

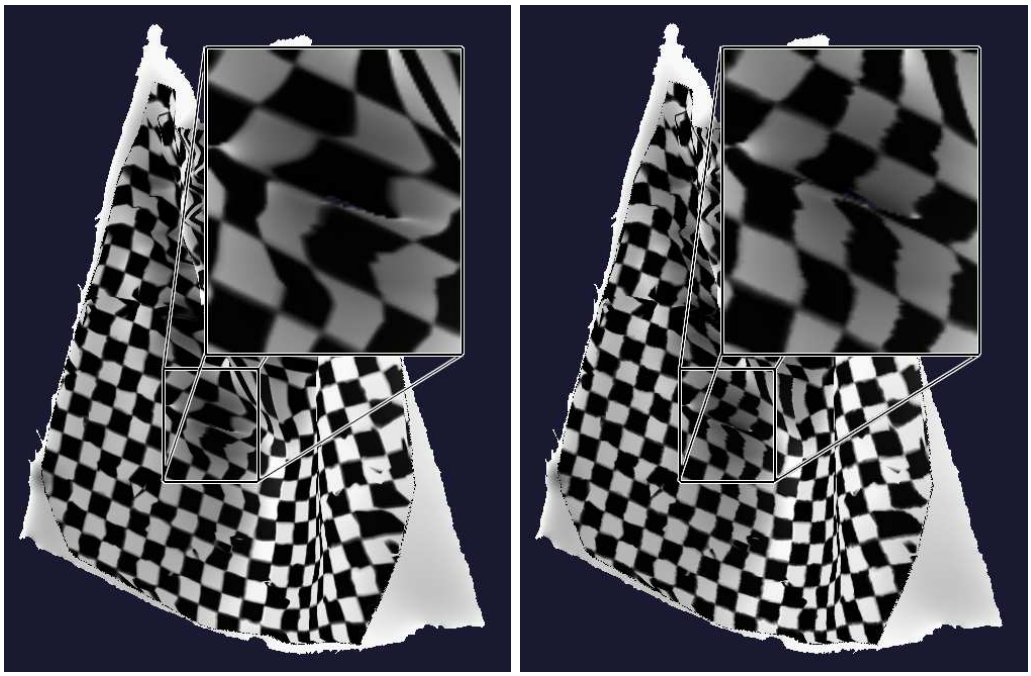


Figure 5.7: Left: capture space interpolation. Right: our interpolation method.

capture space, typically a Delaunay triangulation of the input features. Parameters are known at every vertex of the mesh. Each triangle represents a curved surface patch, with the shape of the patch defined by the underlying disparity map.

We recursively subdivide each triangle into four smaller triangles using the standard 4-to-1 split, but with one slight difference. Instead of inserting new vertices at the capture space midpoint of each edge, we insert at the geodesic midpoint. In other words, if the endpoints of an edge are given by c_1 and c_2, the new vertex $v \in \mathcal{C}$ satisfies $\tilde{g}(c_1, v) = \tilde{g}(v, c_2)$ (where $\tilde{g}$ is the approximate geodesic distance from Equation 5.4), but it does not in general satisfy $||\mathbf{p}(c_1) - v|| = ||v - \mathbf{p}(c_2)||$. Since this point lies midway between the endpoints, its parametric position is the average of the endpoints' parameters. We form four new triangles using the three original vertices and the three new midpoint vertices, and proceed recursively on the smaller triangles.

The recursion stops when a triangle encloses exactly one disparity sample.

At this point, the triangle can be treated as flat. To find the parameters at the disparity sample location, we associate barycentric co-ordinates with the sample location and linearly interpolate the parameters of the triangle's vertices.

This interpolation scheme still has several problems. It is possible that the correct interpolation between two features in $\mathcal{C}$ follows a slightly curved path in $\mathcal{R}$, instead of the straight line path used in this interpolation algorithm. The distortion caused by this approximation should be relatively subtle. More importantly, folds should receive special treatment during interpolation. In theory, it may be possible to make a reasonable guess about the world-space position of occluded regions hidden by the fold, but we leave this as future work.

The final issue in interpolation is finding an appropriate way to resist shearing. In our approach, shearing is not dealt with directly (as Lévy and Mallet did), but we are not certain that this is the best decision. Cloth permits shearing, but it does also resist it. Our scheme does not explicitly incorporate this behaviour. Any algorithm which does mix stretch and shear resistance will have to choose a means of balancing resistance to these two types of forces. It is hard to envision a suitable way of balancing stretching and shearing without some knowledge of the cloth material; we leave this as future work.

Chapter 6

Results

In this chapter, the results produced by our cloth capture system are described in detail. A 63×67 cm cloth was selected for capture, with line art images printed on it in a distinct, non-repeating pattern. The SIFT system detects features using edges, and line art provided a natural way of obtaining a high density of edges. The system was tested with several cloth motions. The principal test consisted of drawing one corner of the cloth along a string, over the course of 20 frames. The numbers cited here refer to this dataset.

Figure 6.1: The Digiclops camera used for triocular video acquisition.

Input data was acquired using a triocular Digiclops camera from Point Grey Research, shown in Figure 6.1. Images were captured at a resolution of 1024×768 and a rate of 10 Hz. The Triclops SDK was used to create a disparity map using a Sum of Absolute Differences (SAD) correlation method, and conservative settings yielded a sparse but reliable disparity map. The stereo mask was kept to a small 7×7 window to limit foreground fattening. A mask image of the cloth was constructed by thresholding and combining the intensity and disparity images. The reference image was acquired using a flatbed scanner and image stitching tools, and was scaled down to a resolution of 992×1024.

The feature detector found 21 000 features in the reference image, and an additional 43 000 features in the oblique views of the reference image. The captured images yielded 4200–6400 features, with the number of features typically directly proportional to the visible cloth area. Feature vectors of 128 dimensions were used, but smaller sizes would also likely be suitable.

The seed-and-grow algorithm accepted matches for 50–60% of the captured features. Stretch and compression of up to 10% was permitted. This margin allowed for error in our approximation of geodesic distance, $\tilde{g}(c_s, c_n)$, and permitted some diagonal stretch (i.e., shear) in the cloth, but was still sufficient to perform quality matching.

In the main dataset, the first ten seeds were typically sufficient to classify over 50% of $\mathbf{F}_c$, and the first 80% of $\mathbf{F}_c$ was usually classified using the first thirty seeds. This process was fairly quick and efficient, and yielded a good dense map of features in the flat regions of the cloth.

Classification of the final 20% of $\mathbf{F}_c$, however, was much slower. These features were typically near folds or poorly illuminated regions of the cloth, and little growth was possible. Consequently, many of these features had to be matched with a slow brute force algorithm, and many were later rejected by the verification algorithm. Nevertheless, a few good matches were made, justifying the continued

search.

We found that the oblique reference views for the SIFT algorithm were definitely valuable for the matching process. Of the matched captured features, over half were matched with reference features from oblique views. Some extremely oblique views were also attempted, scaling the reference image by a factor of four. These views gave very small improvements, usually amounting to less than 5% of all matches, and we therefore chose not to use them.

The verification algorithm was fairly conservative in its acceptance of features, rejecting over 40% of the matched features. Table 6.1 shows the number of accepted features after feature detection, matching, and verification. As can be seen, only 46% of the detected features were accepted. Despite using a conservative verification, it was still possible to track roughly an order of magnitude more features than would be feasible with traditional motion capture or using Guskov's method. [44, 46, 45]

Frame	Visible area	Initial features	Matched features	Verified features
1	271k	6464	4978	2980
6	271k	6458	4966	2948
11	241k	5710	4349	2481
16	207k	4731	3558	2064
20	190k	4249	3233	1861
Average	236k	5567	4408	2578

Table 6.1: Number of features found, matched, and verified for selected frames.

The performance of the system is shown in Table 6.2. Matching was clearly a bottleneck in the system, and the seeding process was the slowest part of matching. The speed of matching on each frame was highly dependent on the initial success of the growth algorithm.

Our final results after parameterisation are shown in Figure 6.2. A checkered texture is used to illustrate the parameterisation of the surface, but clearly any texture could be applied.

Frame	Hole filling, smoothing	Feature detection	Matching	Verification & parameterisation
1	3:15	0:14	2:15	0:36
6	2:53	0:15	2:18	0:42
11	2:34	0:14	2:03	0:33
16	2:47	0:14	1:41	0:28
20	2:17	0:16	1:27	0:25
Average	2:46	0:15	1:57	0:34

Table 6.2: Performance of our system in selected frames, measured in seconds on a Pentium IV 1.8GHz system.

Capture of fast-moving cloth was practical using this system. Figure 6.3 demonstrates one example, where the top left corner of the cloth fell and pivoted about the fixed corner in the top right. This image was taken at the start of the fall, where the left side of the cloth is moving quickly while the right side stays still. Motion blur is evident in the fast-moving left side. As can be seen, capture and parameterisation were successful in both the slow-moving and fast-moving sections of the cloth. SIFT features are scale-invariant, and consequently large features could still be found in the presence of motion blur. We are unaware of any other tracking technology that could achieve similar results.

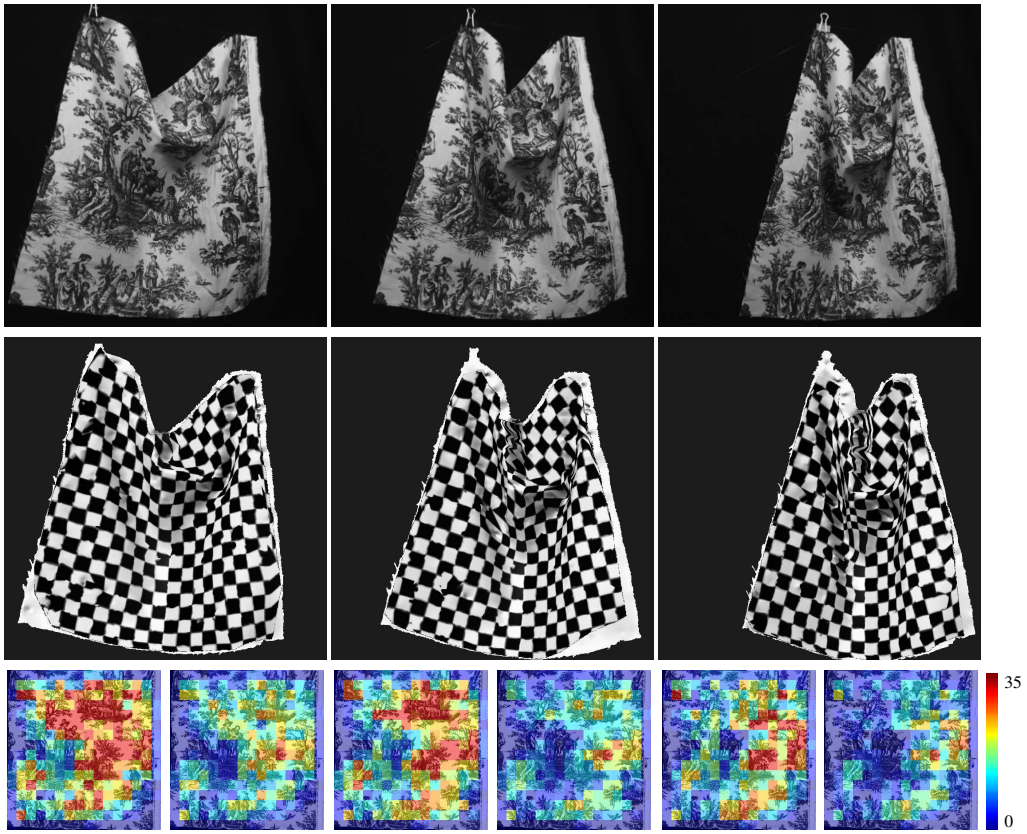


Figure 6.2: Top row: input images, frames 6,11,16. Middle row: parameterised geometry with checkered texture. Bottom row: comparison of matched and verified feature density in $\mathcal{R}$

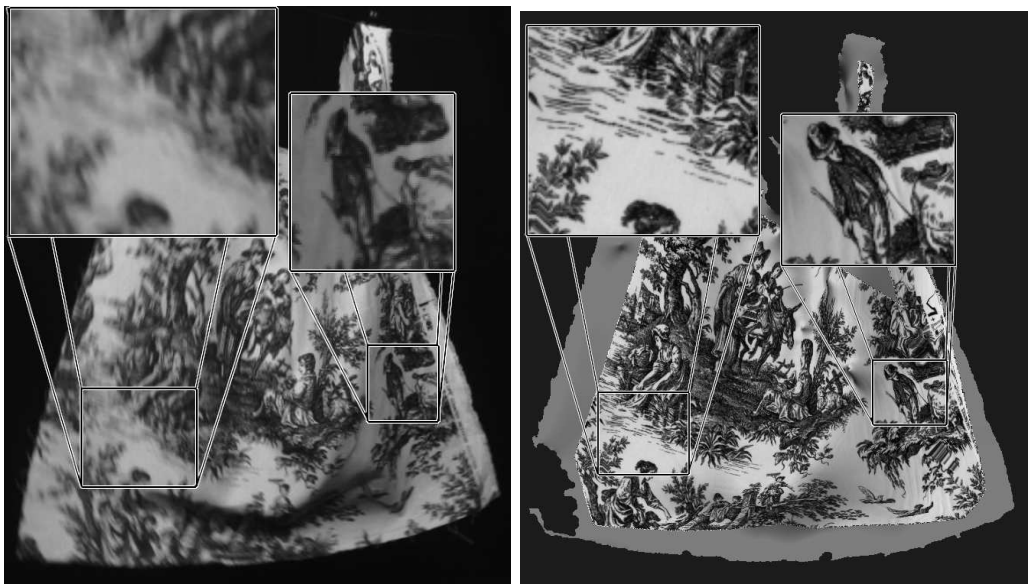


Figure 6.3: Left: captured image of fast moving cloth. Right: parameterised geometry. Left inset is moving quickly while right inset is still.

Chapter 7

Conclusions

In this thesis, we have studied various aspects of cloth simulation parameters, focusing on a novel method for capturing the motion of cloth. Additionally, our experiments in Chapter 3 demonstrated the influence of the parameters of one cloth simulator, and also highlighted the damping effects of large timesteps.

Our cloth capture method is based on a multi-baseline stereo algorithm to capture partial geometry, and the SIFT feature detection algorithm for recovering the parameterisation on that geometry. We employ smoothing and interpolation to fill holes in the geometry due to occlusion or lack of texture, but emphasise that a more sophisticated stereo algorithm could easily be substituted to eliminate these problems.

We have presented a novel seed-and-grow algorithm for recovering the parameterisation of cloth surfaces. One of the advantages of our approach is that we can track features even if they move rapidly and are therefore blurred in the frames of the animation. None of the previous work is capable of dealing with situations like this. This success is made possible by using the SIFT approach (which works for blurred features due to its multi-resolution character), and by not relying on temporal coherence between frames (i.e. by solving the *recognition* rather than the *tracking* problem). On the down side, by not making use of frame-to-frame coher-

ence, we risk having cloth animations that are not as stable as they could be. In the future, we would like to apply temporal filtering to the feature positions to improve frame-to-frame coherence. This would still allow tracking of fast moving parts of the cloth, but would also stabilise slow moving and static parts, and could be achieved through a more sophisticated verification algorithm using simulated annealing.

In our specific implementation, we have used a single trinocular vision system for the geometry recovery. This limits our field of view so that we can only recover single-sided cloth such as towels, curtains, and similar objects. However, it is important to note that our method will extend to calibrated camera systems with any number of cameras. Systems with many synchronised and calibrated cameras are already quite common for traditional motion capture. In our setting, they should allow us to capture objects such as clothing.

Even with multiple cameras, however, there will always be regions where folds occlude sections of the cloth. The parametric information found by our algorithm could be used to estimate the area of the occluded region and hence to infer the probable geometry in occluded regions. We leave this as future work.

The use of a passive algorithm such as multi-baseline stereo has the advantage that colour and possibly reflectance can be acquired at the same time as the geometry and parameterisation. Our feature detection complements the stereo geometry acquisition, as both systems benefit from a richly detailed pattern printed on the cloth. In order to preserve the possibility for colour and reflectance capture, the pattern (and hence the stereo acquisition) could be restricted to a frequency outside the visible spectrum. For example, we could print the patterns with a paint that only changes infrared reflectance. The stereo cameras would then have to operate in the infrared spectrum, similar to the setup in Light Stage 2 [31].

Finally, the captured cloth geometry and parameterisation could be used to solve the problem of cloth parameter recovery, improving the results obtained by Bhat et al. [11]

The premise of cloth parameter recovery is that a single set of parameters can be inferred from a series of experiments with a given cloth material, and then retargetted to novel cloth motion to imitate the material's behaviour. However, this premise may not be valid. As our experiments demonstrated, cloth behaviour in Baraff and Witkin's simulator is highly dependent on the choice of timestep, with large timesteps causing a strong damping effect on cloth motion. This makes the recovery of damping parameters ill-posed, since a given set of recovered damping parameters cannot necessarily be retargetted to yield similar motion. Instead, retargetting will produce variable amounts of damping proportional to the timestep, a parameter which cannot be recovered. Further study of this problem is necessary, including experiments with other cloth simulators.

Once damping in cloth simulation models is sufficiently well understood, the cloth motion capture algorithm presented here should be a useful tool for recovering cloth simulation parameters. This area appears to be a fruitful direction for future research.

Bibliography

[1] M. Aono. *Computer-aided geometric design for forming woven cloth composites*. PhD thesis, Rensselaer Polytechnic Institute, 1994.

[2] M. Aono, D. Breen, and M. Wozny. Fitting a woven cloth model to a curved surface: mapping algorithms. *Computer-Aided Design*, 26(4):278–292, April 1994.

[3] M. Aono, P. Denti, D. Breen, and M. Wozny. Fitting a woven cloth model to a curved surface: dart insertion. *IEEE Computer Graphics and Applications*, 16(5):60–70, September 1996.

[4] U. Ascher and E. Boxerman. On the modified conjugate gradient method in cloth simulation. *The Visual Computer (to appear)*, 2003.

[5] J. Ascough, H. Bez, and A. Bricis. A simple beam element, large displacement model for the finite element simulation of cloth drape. *Journal of the Textile Institute*, 87(1):152–165, 1996.

[6] D. Baraff and A. Witkin. Large steps in cloth simulation. In *Proceedings of ACM SIGGRAPH 98*, pages 43–54. ACM Press, 1998.

[7] D. Baraff and A. Witkin. *Cloth Modeling and Animation*, chapter 6, Rapid dynamic simulation, pages 145–173. A.K. Peters, 2000.

[8] D. Baraff, A. Witkin, and M. Kass. Untangling cloth. *ACM Transactions on Graphics (ACM SIGGRAPH 2003)*, 22(3):862–870, July 2003.

[9] J. Beis and D. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1997)*, pages 1000–1006. IEEE Computer Society, 1997.

[10] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proceedings of ACM SIGGRAPH 2000*, pages 417–424. ACM Press, 2000.

[11] K. Bhat, C. Twigg, J. Hodgins, P. Khosla, Z. Popović, and S. Seitz. Estimating cloth simulation parameters from video. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003)*, pages 37–51. ACM Press, 2003.

[12] M. Black, G. Sapiro, D. Marimont, and D. Heeger. Robust anisotropic diffusion. *IEEE Transactions on Image Processing*, 7(3):421–432, March 1998.

[13] D. Breen. *A particle-based model for simulating the draping behavior of woven cloth*. PhD thesis, Rensselaer Polytechnic Institute, 1993.

[14] D. Breen. *Cloth Modeling and Animation*, chapter 2, A survey of cloth modeling methods, pages 19–53. A.K. Peters, 2000.

[15] D. Breen, D. House, and P. Getto. A physically-based particle model of woven cloth. *The Visual Computer*, 8(5–6):264–277, June 1992.

[16] D. Breen, D. House, and M. Wozny. A particle-based model for simulating the draping behavior of woven cloth. *Textile Research Journal*, 64(11):663–685, 1994.

[17] D. Breen, D. House, and M. Wozny. Predicting the drape of woven cloth using interacting particles. In *Proceedings of ACM SIGGRAPH 94*, pages 365–372. ACM Press, 1994.

[18] R. Bridson. *Computational aspects of dynamic surfaces*. PhD thesis, Stanford University, 2003.

[19] R. Bridson, R. Fedkiw, and J. Anderson. Robust treatment of collisions, contact and friction for cloth animation. *ACM Transactions on Graphics (ACM SIGGRAPH 2002)*, 21(3):594–603, July 2002.

[20] R. Bridson, S. Marino, and R. Fedkiw. Simulation of clothing with folds and wrinkles. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003)*, pages 28–36. ACM Press, 2003.

[21] M. Brown and D. Lowe. Invariant features from interest point groups. In *Proceedings of British Machine Vision Conference (BMVC 2002)*, pages 253–262, 2002.

[22] M. Brown and D. Lowe. Recognising panoramas. In *Proceedings of IEEE International Conference on Computer Vision (to appear)*. IEEE Computer Society, 2003.

[23] R. Carceroni. *Recovering non-rigid 3D motion, shape, and reflectance from multi-view image sequence: a differential-geometric approach*. PhD thesis, University of Rochester, 2001.

[24] R. Carceroni and K. Kutulakos. Multi-view scene capture by surfel sampling: from video streams to non-rigid 3D motion, shape and reflectance. In *Proceedings of IEEE International Conference on Computer Vision (ICCV 2001)*, volume 2, pages 60–67. IEEE Computer Society, 2001.

[25] M. Carignan, Y. Yang, N. Magnenat-Thalmann, and D. Thalmann. Dressing animated synthetic actors with complex deformable clothes. In *Computer Graphics (Proceedings of ACM SIGGRAPH 92)*, pages 99–104. ACM Press, 1992.

[26] V. Caselles, J.-M. Morel, and C. Sbert. An axiomatic approach to image interpolation. *IEEE Transactions on Image Processing*, 7(3):376–386, March 1998.

[27] K.-J. Choi and H.-S. Ko. Stable but responsive cloth. *ACM Transactions on Graphics (ACM SIGGRAPH 2002)*, 21(3):604–611, July 2002.

[28] K.-J. Choi and H.-S. Ko. Extending the immediate buckling model to triangular meshes for simulating complex clothes. In *Eurographics 2003 Short Presentations*, pages 187–191, 2003.

[29] T. Coleman and Y. Li. A reflective Newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM Journal on Optimization*, 6(4):1040–1058, 1996.

[30] F. Cordier and N. Magnenat-Thalmann. Real-time animation of dressed virtual humans. *Computer Graphics Forum (Eurographics 2002)*, 21(3):327–336, September 2002.

[31] P. Debevec, A. Wenger, C. Tchou, A. Gardner, J. Waese, and T. Hawkins. A lighting reproduction approach to live-action compositing. *ACM Transactions on Graphics (ACM SIGGRAPH 2002)*, 21(3):547–556, July 2002.

[32] M. Desbrun, M. Meyer, and A. Barr. *Cloth Modeling and Animation*, chapter 9, Interactive animation of cloth-like objects in virtual reality, pages 219–239. A.K. Peters, 2000.

[33] M. Desbrun, P. Schröder, and A. Barr. Interactive animation of structured deformable objects. In *Proceedings of Graphics Interface (GI 1999)*, pages 1–8. Canadian Computer-Human Communications Society, 1999.

[34] R. DeVaul. Cloth dynamics simulation. Master's thesis, Texas A&M University, 1997.

[35] B. Eberhardt, O. Etzmuß, and M. Hauth. Implicit-explicit schemes for fast animation with particle systems. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation 2000 (CAS 2000)*. Springer-Verlag, 2000.

[36] B. Eberhardt, M. Meißner, and W. Straßer. *Cloth Modeling and Animation*, chapter 5, Knit fabrics, pages 123–144. A.K. Peters, 2000.

[37] B. Eberhardt, A. Weber, and W. Straßer. A fast, flexible, particle-system model for cloth draping. *IEEE Computer Graphics and Applications*, 16(5):52–59, September 1996.

[38] J. Eischen and R. Bigliani. *Cloth Modeling and Animation*, chapter 4, Continuum versus particle representations, pages 79–122. A.K. Peters, 2000.

[39] J. Eischen, S. Deng, and T. Clapp. Finite-element modeling and control of flexible fabric parts. *IEEE Computer Graphics and Applications*, 16(5):71–80, September 1996.

[40] O. Etzmuß, J. Groß, and W. Straßer. Deriving a particle system from continuum mechanics for the animation of deformable objects. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):538–550, October 2003.

[41] O. Etzmuß, M. Hauth, M. Keckeisen, S. Kimmerle, J. Mezger, and M. Wacker. A cloth modelling system for animated characters. In *Proceedings of Graphiktag 2001*. Wilhelm-Schickard-Institut für Informatik, Graphisch-Interaktive Systeme, Universität Tübingen, 2001.

[42] C. Feynman. Modeling the appearance of cloth. Master's thesis, Massachusetts Institute of Technology, 1986.

[43] A. Fuhrmann, C. Groß, and V. Luckas. Interactive animation of cloth including self collision detection. In *Proceedings of Winter School of Computer Graphics (WSCG 2003)*, pages 203–208, 2003.

[44] I. Guskov. Efficient tracking of regular patterns on non-rigid geometry. In *Proceedings of IEEE International Conference on Pattern Recognition (ICPR)*, volume 2, pages 1057–1060. IEEE Computer Society, 2002.

[45] I. Guskov, S. Klibanov, and B. Bryant. Trackable surfaces. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003)*, pages 251–257. ACM Press, 2003.

[46] I. Guskov and L. Zhukov. Direct pattern tracking on flexible geometry. In *Proceedings of Winter School of Computer Graphics (WSCG 2002)*, pages 203–208, 2002.

[47] S. Hadap, E. Bangerter, P. Volino, and N. Magnenat-Thalmann. Animating wrinkles on clothes. In *Proceedings of IEEE Visualization 1999*, pages 175–182. IEEE Computer Society, 1999.

[48] J. Haddon and D. Forsyth. Shading primitives: finding folds and shallow grooves. In *Proceedings of IEEE International Conference on Computer Vision (ICCV 1998)*, pages 236–241. IEEE Computer Society, 1998.

[49] M. Hauth and O. Etzmuß. A high performance solver for the animation of deformable objects using advanced numerical methods. *Computer Graphics Forum (Eurographics 2001)*, 20(3):319–328, September 2001.

[50] M. Hauth, O. Etzmuß, and W. Straßer. Numerical methods for deformable models. Technical Report WSI-2001-7, Eberhardt-Karls-Universität Tübingen, August 2001.

[51] M. Hauth, O. Etzmuß, and W. Straßer. Analysis of numerical methods for the simulation of deformable models. *The Visual Computer (to appear)*, 2003.

[52] D. House and D. Breen, editors. *Cloth Modeling and Animation*. A.K. Peters, 2000.

[53] D. House and D. Breen. *Cloth Modeling and Animation*, chapter 3, Particle representation of woven fabrics, pages 55–78. A.K. Peters, 2000.

[54] D. House, R. DeVaul, and D. Breen. Towards simulating cloth dynamics using interacting particles. *International Journal of Clothing Science and Technology*, 8(3):75–94, 1996.

[55] S. Huh, D. Metaxas, and N. Badler. Collision resolutions in cloth simulation. In *Proceedings of Computer Animation 2001*, pages 122–127. IEEE Computer Society, 2001.

[56] D. Hutchinson, M. Preston, and T. Hewitt. Adaptive refinement for mass/spring simulations. In *Proceedings of the Eurographics Workshop on*

Computer Animation and Simulation (CAS 1996), pages 31–45. Springer-Verlag, 1996.

[57] N. Jojic. Computer modeling, analysis and synthesis of dressed humans. Master's thesis, University of Illinois at Urbana-Champaign, 1997.

[58] N. Jojic, J. Gu, H. Shen, and T. Huang. Computer modeling, analysis and synthesis of dressed humans. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1998)*, pages 528–534. IEEE Computer Society, 1998.

[59] N. Jojic and T. Huang. Estimating cloth draping parameters from range data. In *Proceedings of International Workshop on Synthetic-Natural Hybrid Coding and 3-D Imaging*, pages 73–76, 1997.

[60] N. Jojic and T. Huang. On analysis of cloth drape range data. In *Proceedings of Asian Conference on Computer Vision*, pages 463–470. Springer-Verlag, 1998.

[61] Y.-M. Kang and H.-G. Cho. Bilayered approximate integration for rapid and plausible animation of virtual cloth with realistic wrinkles. In *Proceedings of Computer Animation*, pages 203–214. IEEE Computer Society, 2002.

[62] Y.-M. Kang and H.-G. Cho. Complex deformable objects in virtual reality. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST 2002)*, pages 49–56. ACM Press, 2002.

[63] Y.-M. Kang, J.-H. Choi, H.-G. Cho, and D.-H. Lee. An efficient animation of wrinkled cloth with approximate implicit integration. *The Visual Computer*, 17(3):147–157, May 2001.

[64] Y.-M. Kang, J.-H. Choi, H.-G. Cho, D.-H. Lee, and C.-J. Park. Real-time animation technique for flexible and thin objects. In *Proceedings of the Winter School of Computer Graphics (WSCG 2000)*, pages 322–329, 2000.

[65] Y.-M. Kang, J.-H. Choi, H.-G. Cho, and C.-J. Park. A fast and stable cloth animation with approximation of implicit method. In *Proceedings of Computer Graphics International (CGI 2000)*, pages 257–255. IEEE Computer Society, 2000.

[66] S. Kawabata. *The standardization and analysis of hand evaluation*. The Textile Machinery Society of Japan, 1980.

[67] R. Lario, C. García, M. Prieto, and F. Tirado. Rapid parallelization of a multilevel cloth simulator using OpenMP. In *Proceedings of European Workshop on OpenMP (EWOMP 2001)*, 2001.

[68] R. Lario, C. García, M. Prieto, and F. Tirado. A parallel cloth simulator using multilevel algorithms. In *Proceedings of the International Parallel and Distributed Processing Symposium*, pages 403–408. IEEE Computer Society, 2002.

[69] B. Lévy and J.-L. Mallet. Non-distorted texture mapping for sheared triangulated meshes. In *Proceedings of ACM SIGGRAPH 98*, pages 343–352. ACM Press, 1998.

[70] L. Li. *Modeling and computer animation of cloth motion in air flow.* PhD thesis, Nanyang Technological University, 1994.

[71] L. Li. *Cloth Modeling and Animation*, chapter 7, Aerodynamics effects, pages 175–195. A.K. Peters, 2000.

[72] L. Li, M. Damodaran, and R. Gay. A quasi-steady force model for animating cloth motion. In *Proceedings of the International Conference on Computer Graphics (ICCG 1993)*, pages 357–364. North-Holland, 1993.

[73] L. Li, M. Damodaran, and R. Gay. Aerodynamic force models for animating cloth motion in air flow. *The Visual Computer*, 12(2):84–104, February 1996.

[74] L. Li, D. Murali, and R. Gay. A model for animating cloth motion. *Computers and Graphics*, 20(1):137–156, January 1996.

[75] J. Louchet, X. Provot, and D. Crochemore. Evolutionary identification of cloth animation models. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation (CAS 1995)*, pages 44–54. Springer-Verlag, Sep 1995.

[76] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of IEEE International Conference on Computer Vision (ICCV 1999)*, pages 1150–1157. IEEE Computer Society, 1999.

[77] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (submitted)*, 2003.

[78] M. Meyer, G. Debunne, M. Desbrun, and A. Barr. Interactive animation of cloth-like objects in virtual reality. *Journal of Visualization and Computer Animation*, 12(1):1–12, February 2001.

[79] J. Mezger, S. Kimmerle, and O. Etzmuß. Improved collision detection and response techniques for cloth animation. Technical Report WSI-2002-5, Universität Tübingen, August 2002.

[80] J. Mezger, S. Kimmerle, and O. Etzmuß. Progress in collision detection and response techniques for cloth animation. In *Proceedings of 10th Pacific Conference on Computer Graphics and Applications (PG 2002)*, pages 444–445. IEEE Computer Society, 2002.

[81] J. Mezger, S. Kimmerle, and O. Etzmuß. Hierarchical techniques in collision detection for cloth animation. In *Proceedings of Winter School of Computer Graphics (WSCG 2003)*, pages 322–329, 2003.

[82] H. Ng and R. Grimsdale. Computer graphics techniques for modeling cloth. *IEEE Computer Graphics and Applications*, 16(5):28–41, September 1996.

[83] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer Verlag, 2002.

[84] D. Papadimitriou and T. Dennis. Epipolar line estimation and rectification for stereo image pairs. *IEEE Transactions on Image Processing*, 5(4):672–676, April 1996.

[85] D. Parks and D. Forsyth. Improved integration for cloth simulation. In *Eurographics 2002 Short Presentations*, 2002.

[86] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Transactions on Graphics (ACM SIGGRAPH 2003)*, 22(3):313–318, July 2003.

[87] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1999.

[88] D. Pritchard and W. Heidrich. Cloth motion capture. *Computer Graphics Forum (Eurographics 2003)*, 22(3):263–271, September 2003.

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

[90] X. Provot. *Animation réaliste de vêtements*. PhD thesis, Université de Paris 5, 1997.

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

[92] S. Romero, L. Romero, and E. Zapata. Approaching real-time cloth simulation using parallelism. In *Proceedings of 16th IMACS World Congress on Scientific Computation, Applied Mathematics and Simulation*, pages 220–227. Rutgers University, 2000.

[93] S. Romero, L. Romero, and E. Zapata. Fast cloth simulation with parallel computers. In *Proceedings of 6th International Euro-Par Conference*, pages 491–499. Springer-Verlag, 2000.

[94] S. Romero, L. Romero, and E. Zapata. Parallel algorithm for fast cloth simulation. In *Proceedings of 4th International Conference on Vector and Parallel Processing (VECPAR 2000)*, pages 529–535. Springer-Verlag, 2000.

[95] M. Ruiz and B. Buxton. A model-based procedure for fitting a surface to scans of clothed people. In *Proceedings of Scanning 2001 (Numérisation 3D and Human Modeling)*, 2001.

[96] G. Sapiro. *Geometric Partial Differential Equations and Image Analysis*. Cambridge University Press, 2001.

[97] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Technical Report MSR-TR-2001-81, Microsoft Research, 2002.

[98] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1–3):7–42, April–June 2002.

[99] D. Scharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *Proceedings of IEEE Workshop on Stereo and Multi-Baseline Vision*, pages 131–140. IEEE Computer Society, 2001.

[100] M. Shimizu and M. Okutomi. Precise sub-pixel estimation on area-based matching. In *Proceedings of IEEE International Conference on Computer Vision (ICCV 2001)*, volume 1, pages 90–97. IEEE Computer Society, 2001.

[101] D. Terzopolous and K. Fleischer. Deformable models. *The Visual Computer*, 4(6):306–331, December 1988.

[102] D. Terzopolous, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *Computer Graphics (Proceedings of ACM SIGGRAPH 87)*, pages 205–214. ACM Press, July 1987.

[103] L. Torresani, D. Yang, E. Alexander, and C. Bregler. Tracking and modeling non-rigid objects with rank constraints. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2001)*, volume 1, pages 493–500. IEEE Computer Society, 2001.

[104] A. van Gelder. Approximate simulation of elastic membranes by triangulated spring meshes. *Journal of Graphics Tools*, 3(2):21–41, October 1998.

[105] A. van Gelder and J. Wilhelms. Simulation of elastic membranes with triangulated spring meshes. Technical Report UCSC-CRL-97-12, University of California Santa Cruz, July 1997.

[106] J. Villard and H. Borouchaki. Adaptive meshing for cloth animation. In *Proceedings of the 11th International Meshing Roundtable (IMR 2002)*, pages 243–252. Sandia National Laboratories, 2002.

[107] P. Volino, M. Courchesne, and N. Magnenat-Thalmann. Versatile and efficient techniques for simulating cloth and other deformable objects. In *Proceedings of ACM SIGGRAPH 95*, pages 137–144. ACM Press, 1995.

[108] P. Volino and N. Magnenat-Thalmann. Developing simulation techniques for an interactive clothing system. In *Proceedings of International Conference on Virtual Systems and Multimedia*, pages 109–118. IEEE Computer Society, 1997.

[109] P. Volino and N. Magnenat-Thalmann. Implementing fast cloth simulation with collision response. In *Proceedings of Computer Graphics International 2000 (CGI 2000)*, pages 257–268. IEEE Computer Society, 2000.

[110] P. Volino and N. Magnenat-Thalmann. Comparing efficiency of integration methods for cloth simulation. In *Proceedings of Computer Graphics International 2001 (CGI 2001)*, pages 265–274. IEEE Computer Society, 2001.

[111] P. Volino, N. Magnenat-Thalmann, S. Jianhua, and D. Thalmann. An evolving system for simulating clothes on virtual actors. *IEEE Computer Graphics and Applications*, 16(5):42–51, September 1996.

[112] V. Volkov and L. Li. Adaptive local refinement and simplification of cloth meshes. In *Proceedings of the First International Conference on Information Technology and Applications (ICITA 2002)*, 2002.

[113] Y.-Q. Xu, C. Cheng, J. Shi, and H.-Y. Shum. Physically based real-time animation of curtains. Technical Report MSR-TR-2000-34, Microsoft Research, April 2000.

[114] L. You, J. Zhang, and P. Comninos. Cloth deformation modelling using a plate bending model. In *Proceedings of Winter School of Computer Graphics (WSCG 1999)*, pages 485–491, 1999.

[115] F. Zara, F. Faure, and J.-M. Vincent. Physical cloth simulation on a PC cluster. In *Proceedings of 4th Eurographics Workshop on Parallel Graphics and Visualization*, pages 105–112. ACM Press, 2002.

[116] D. Zhang and M. Yuen. Cloth simulation using multilevel meshes. *Computers & Graphics*, 25(3):383–389, June 2001.

[117] Y. Zhao, T. Wong, S. Tan, and W. Chen. A model for simulating flexible surfaces of cloth objects. *Computers and Structures*, 63(1):133–147, April 1997.