 GRAPHICS 2003 / P. Brunet and D. Fellner
(Guest Editors)

Volume 22 (*2003*), *Number 3*

Cloth Motion Capture

D. Pritchard and W. Heidrich

Imager Laboratory, The University of British Columbia, Vancouver, Canada

Abstract

Recent years have seen an increased interest in motion capture systems. Current systems, however, are limited to only a few degrees of freedom, so that effectively only the motion of linked rigid bodies can be acquired. We present a system for the capture of deformable surfaces, most notably moving cloth, including both geometry and parameterisation. 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.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Physically based modeling I.4.8 [Image Processing and Computer Vision]: Scene analysis

1. Introduction

In the last several years, the interest in computer simulated cloth algorithms has greatly increased. Due to the development of fast cloth simulation algorithms such as those proposed by Baraff & Witkin[1] and more recent methods introduced by Choi & Ko[4] and Bridson et al.[3], it is now possible to simulate complex and realistic looking cloth. Unfortunately, these realistic approaches are still much too demanding for realtime applications. In addition, it is quite hard to adjust the parameters so that the simulations match a given specific real world cloth material.

One way of getting around these difficulties would be a data-driven approach in which the motion of real-world cloth is acquired. Such methods have been hugely successful recently in the context of motion capture for human movements. These systems use optical or magnetic tracking of individual points. Unfortunately, they are usually limited to only a few degrees of freedom, and can therefore not be used to track deformable surfaces with many degrees of freedom.

In this paper, we consider a relatively new problem: the acquisition of both the geometry and parameterisation of a moving sheet of cloth. We choose to use multi-baseline stereo[13] for geometry acquisition. Our current system only has three fixed cameras and can therefore only capture one side of the cloth geometry, limiting it to applications such as moving curtains, flags or draped cloth. Using a larger number of calibrated cameras (traditional motion capture systems often use 8 or more, sometimes as many as 50), our method could be extended to applications such as clothing.

In addition to obtaining partial geometry from the stereo algorithm, we also identify feature points in an arbitrary pattern printed on the cloth. To this end, we extend Lowe's scale invariant feature transform (SIFT)[11] to deal with deformable objects. We find that with this approach we are able to recognise even quickly moving features that are only depicted in a blurred fashion. The partial geometric information is then merged with parametric information from the feature points to create a comprehensive parametric model of the cloth surface for each frame. An overview of the system is shown in Figure 1.

The remainder of this paper is organised as follows: in Section 2 we briefly review the related work on motion capture, cloth simulation, and feature tracking. We then discuss our system, beginning with the generation of the disparity information (Section 3), followed by feature detection (Section 4.1), matching (Section 4.2), and verification (Section 4.3). We then describe how to generate the final geo-

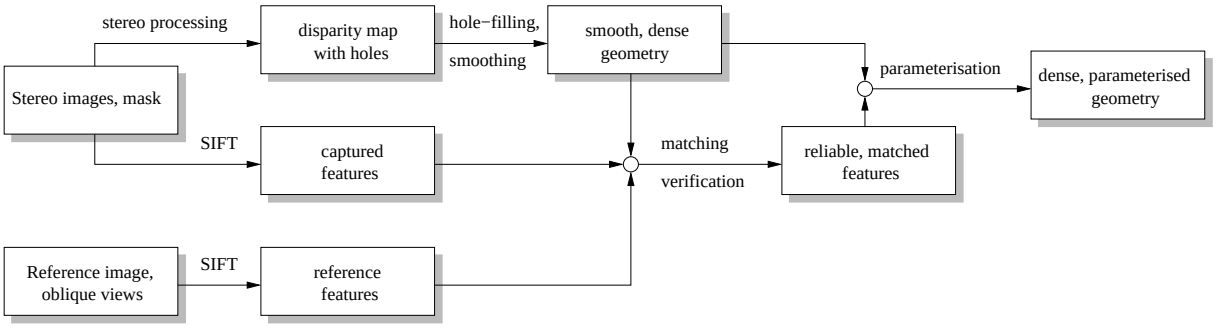

Figure 1: *System overview.*

metric model in Section 5, present results in Section 6, and make concluding remarks in Section 7.

2. Related Work

Jojic et al.[10] were the first to address the problem of cloth capture. They used range data to estimate parameters for a cloth model. Their system was intended to work with draped cloth, and is not very suitable for moving cloth. Their method relied heavily upon a spring-based model of internal cloth forces, while our approach makes very few assumptions about the forces acting on the cloth.

Haddon et al.[8] recovered information about folds from still images of cloth, but made no effort to examine the three-dimensional shape of the cloth or its parameterisation.

The work of Guskov[6, 7] is closest in spirit to our own. He tracked a checkered pattern printed on cloth, recovering sparse parametric information. He suggested integration of his system with a stereo correspondence system to generate dense 3D data, but did not attempt it. Our approach allows a much wider range of patterns to be printed on the cloth, and we demonstrate the recovery of dense 3D data with parametric information, suitable for rendering in a 3D graphics system. Furthermore, Guskov solved the *tracking* problem using temporal prediction to calculate the parameterisation. We tackle the harder and more general problem of *recognition*, with the aim of finding a more robust solution even in the presence of fast motion.

Lowe[11] described a Scale Invariant Feature Transform (SIFT) for greyscale images. 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 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. We make heavy use of SIFT features, but adapt the matching to the parameterisation of cloth, a deformable surface.

Baraff & Witkin[1], House & Breen[9], and Choi & Ko[4] created cloth simulation systems. In all cases, they relied upon the assumption that woven cloth does not stretch significantly along the warp or weft directions under normal loads. Depending upon the tightness of the weave, cloth may allow some shear, which can also be seen as stretch along the diagonal. In our system, we also rely upon the assumption that the cloth will not stretch significantly, and use this to constrain the parameterisation process.

3. Disparity Map

The input data to our algorithm consists of 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.

Most stereo systems provide incomplete disparity maps, with numerous holes present in the map. Holes occur for a variety of reasons, including insufficient texture, image noise and depth discontinuities. As a preprocessing step, we smoothly fill all holes using image-processing operations.

For each hole, we construct a vector $\mathbf{x}$ whose entries correspond to the disparities of the samples inside the hole. We then find $\mathbf{x}$ that minimises

$$\sum \nabla^2(\mathbf{x}_i)^2, \tag{1}$$

where ∇^2 is the spatial Laplacian operator, which will include special terms to ensure a smooth connection with the hole boundary. We formulate this as a linear least-squares problem and solve. A multiscale approach is used to improve performance in large holes.

A typical disparity map contains an integer disparity at

every known pixel. This disparity can be inverted to obtain a depth at every pixel. Many stereo systems using calibrated cameras can convert from a (row,column,disparity) triple to a position in 3D space. However, by using integer disparities, the depth values are visibly quantised, giving a jagged surface.

In order to eliminate these artefacts, we also calculate a smoothly varying fractional part for each disparity sample, without making any modification to the integer part. We use a similar scheme to that used for hole filling, defining a vector $\Delta\mathbf{x}$ subject to the bounds $-0.5 \leq \Delta\mathbf{x}_i \leq 0.5$. We then find $\Delta\mathbf{x}$ that minimises

$$\sum \nabla^2(\mathbf{x}_i + \Delta\mathbf{x}_i)^2. \tag{2}$$

Samples at the boundary of the surface are special cases, with outside disparities excluded from the Laplacian. The bounds on $\Delta\mathbf{x}_i$ serve to limit changes to the fractional part of the disparity sample, and can be relaxed if further smoothing is desired. We solve this in a multiscale fashion as a constrained linear least-squares problem and achieve excellent results (see Figure 2), albeit somewhat slowly.

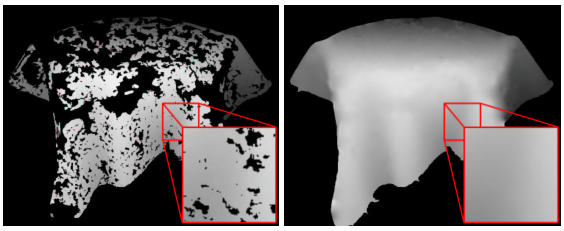

Figure 2: *Left: original disparity map of a draped tablecloth. Right: hole-filled, smoothed disparity map.*

4. 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.

4.1. Feature Detection

For feature detection, we use the Scale-Invariant Feature Transform (SIFT) described by Lowe[11]. We detect features in two 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 parameters for the cloth. This 2D parametric *reference space* is denoted $\mathcal{R}$.

The second image is the input intensity image, which we call the *captured* image here. We refer to this 2D image space as *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 us with 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.

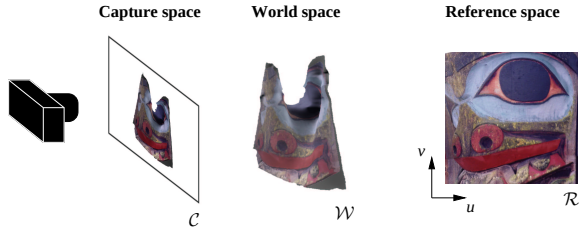

Figure 3: *Capture space is an image of 3D world space. Reference space is a flattened view of the cloth.*

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 4.

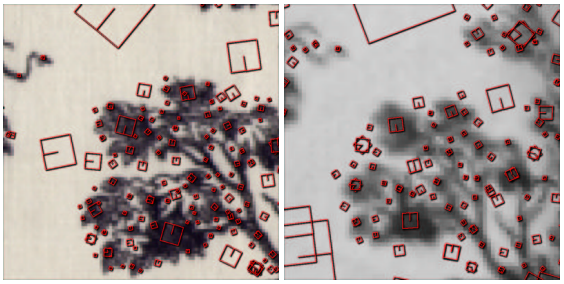

Figure 4: *Left: reference feature set* $\mathbf{F}_r$*. Right: captured feature set* $\mathbf{F}_c$*.*

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 4.2, 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. Consequently, 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. 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 $\mathbf{T}^{-1}$.

Figure 5: *Top row: a reference image, horizontally scaled oblique view. Bottom row: other oblique views.*

4.2. 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 $c_s, c_n \in \mathbf{F}_c$:

$$\Delta d_c = g(c_s, c_n). \tag{3}$$

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 in $\mathcal{W}$ between them. 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.

$$\Delta d_r = \alpha_r \cdot \|\mathbf{p}(r_s) - \mathbf{p}(r_n)\| \tag{4}$$

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

$$\Delta d_r(1 - k_s) < \Delta d_c < \Delta d_r(1 + k_s) \tag{5}$$

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 6 illustrates these constraints.

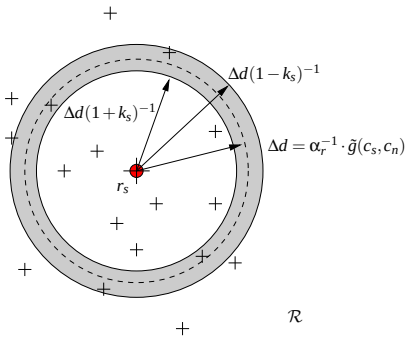

Figure 6: *If we fix two captured features c_s and c_n and one reference feature r_s, the stretch and compression constraints require the remaining reference feature to lie in a ring centred on r_s. The ring's inner and outer radii are derived from Equations 4 and 5.*

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.

$$\Delta d_c = \tilde{g}(c_s, c_n) \tag{6}$$

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.

4.2.1. 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,[2] which is essentially an approximate search in a k-d tree. 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.

4.2.2. 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 6.

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.

4.3. 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.

5. Geometry Parameterisation

After verification, we are left with a set of reliable features, and a dense, regularly sampled disparity map. We would like to construct a mesh that contains both 3D and parametric data. We choose to interpolate the parametric information given by the features to construct dense, regularly sampled parametric map corresponding directly to the disparity map.

An interpolation in capture space is not sufficient, as demonstrated in Figures 7 and 8. 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 7). This must be extended from our one-dimensional example to a surface.

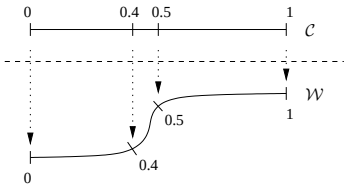

Figure 7: *Example where linear interpolation of parameter values in $\mathcal{C}$ results in distortion of parameters when projected into $\mathcal{W}$.*

This problem is similar in principle to the non-distorted texture mapping problem described by Lévy & Mallet[12] and others. Their technique enforces two primary constraints, perpendicularity and constant spacing of the isoparametric curves traced 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.

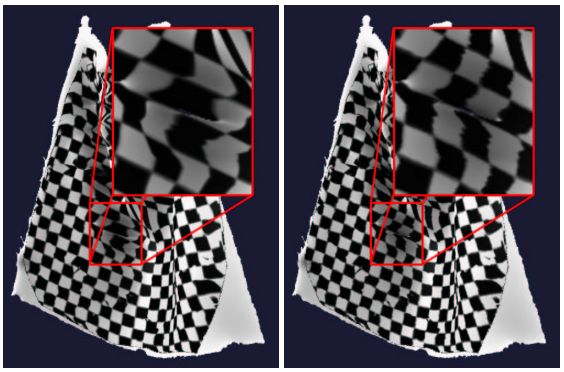

Figure 8: *Left: capture space interpolation. Right: our interpolation method.*

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 we aim, like Lévy & Mallet, 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 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. Rather than 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 6), 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.

6. Results and Discussion

We selected a 63×67 cm cloth with line art images printed on it in a distinct, non-repeating pattern. The SIFT system detects features using edges, and line art provides a natural way of obtaining a high density of edges. We tested our system with several cloth motions. Our 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.

In our experiments, input data was acquired using a Point Grey Digiclops camera. 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, with conservative settings yielding a sparse but reliable disparity map. The stereo mask was kept to a small 7×7 window to avoid excessive "foreground fattening," a standard problem with stereo correspondence algorithms.[14] A mask image of the cloth was constructed by thresholding and combining the intensity and disparity images. Our 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 4 200–6 400 features, with the number of features typically directly proportional to the visible cloth area. We used feature vectors of 128 dimensions, but smaller sizes would also likely be suitable.

The seed-and-grow algorithm accepted matches for 50–60% of the captured features. We allowed stretch and compression of up to 10%. 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 our main dataset, we found that 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 are 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. We also attempted some extremely oblique views, 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 chose not to use them.

The verification algorithm was fairly conservative in its acceptance of features, rejecting 30–40% of the matched features. Table 1 shows the number of accepted features after feature detection, matching, and verification. As can be seen, we typically only accept 37% of the detected features. Despite using a conservative verification, we are still able to track roughly an order of magnitude more features than would be feasible with traditional motion capture or the method by Guskov.[6, 7]

Frame	Visible area	Initial features	Matched features	Verified features
1	271k	6464	3910	2532
6	271k	6458	3891	2575
11	241k	5710	3263	2068
16	207k	4731	2733	1824
20	190k	4249	2306	1516
Average	236k	5567	3240	2103

Table 1: *Number of features found, matched, and verified for selected frames.*

The performance of our system is shown in Table 2. Matching is clearly a bottleneck in our system, and the seeding process is the slowest part of matching. The speed of matching on each frame is highly dependent on the initial success of the growth algorithm.

Frame	Hole filling, smoothing	Feature detection	Matching	Verification & parameterisation
1	3:15	0:14	2:45	0:28
6	2:53	0:15	2:43	0:39
11	2:34	0:14	2:27	0:30
16	2:47	0:14	2:12	0:19
20	2:17	0:16	1:52	0:21
Average	2:46	0:15	2:25	0:26

Table 2: *Performance of our system in selected frames, measured in seconds on a Pentium IV 1.8GHz system.*

Our final results after parameterisation are shown in Figure 9. We use a checkered texture to illustrate the parameterisation of the surface, but clearly any texture could be applied.

We have found that capture of fast-moving cloth is practical. Figure 10 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

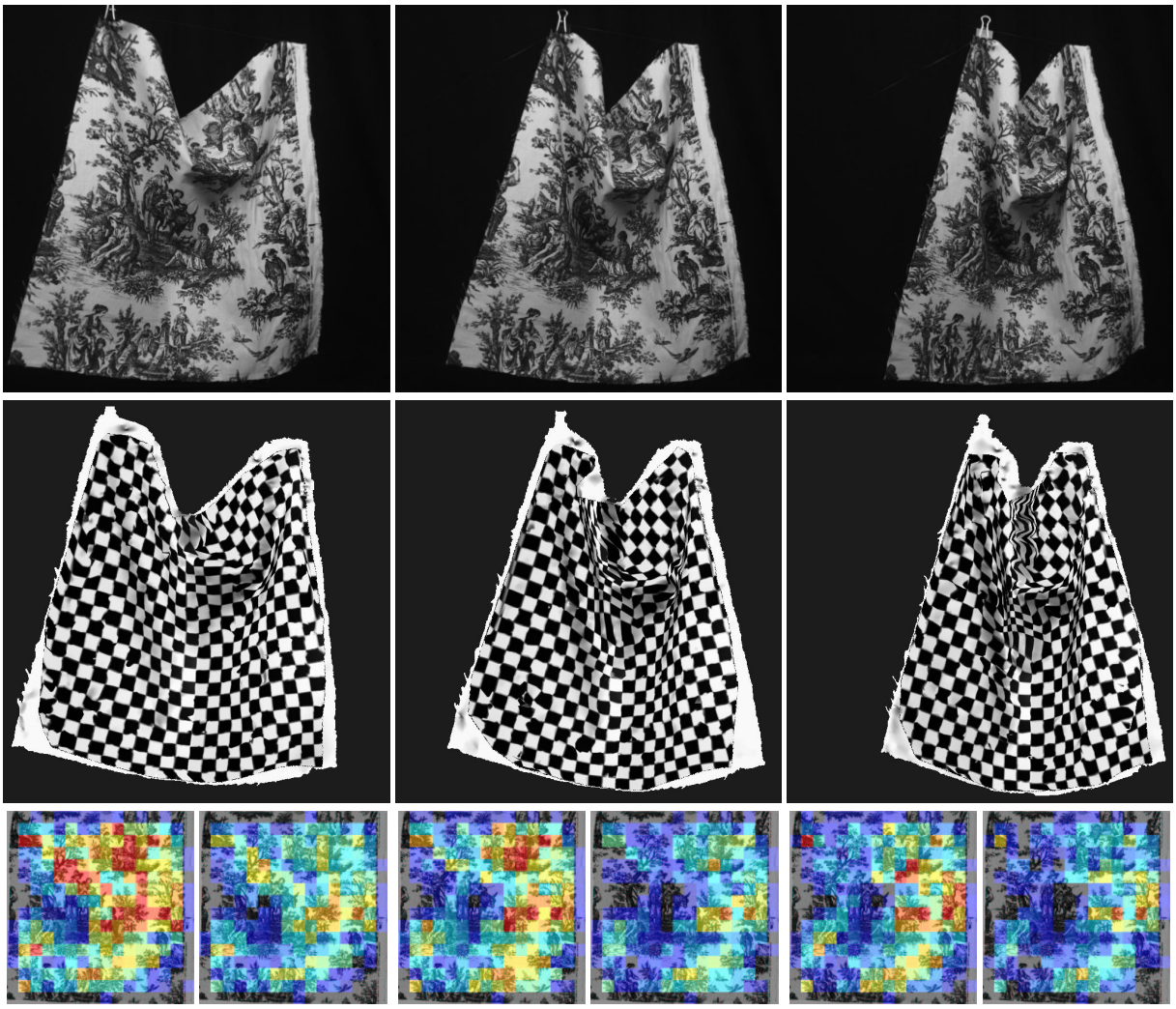

Figure 9: *Top row: input images, frames 6,11,16. Middle row: parameterised geometry with checkered texture. Bottom row: comparison of matched and verified feature density in $\mathcal{R}$ (from blue=1 to red=35, with grey=0)*

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 can still be found in the presence of motion blur. We are unaware of any other tracking technology that could achieve similar results.

7. Conclusions and Future Work

In this paper, we have described an approach for motion capture of cloth. The method is based on multi-baseline stereo algorithms 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.

One of the biggest 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 coherence, 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. This

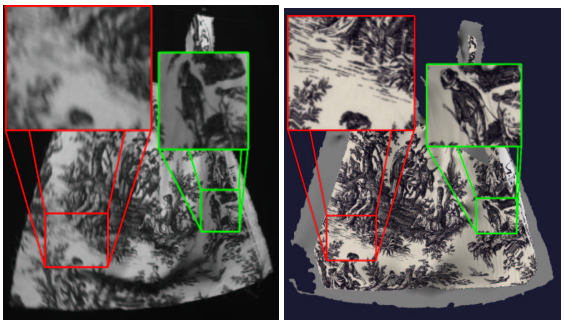

Figure 10: *Left: captured image of fast moving cloth. Right: parameterised geometry. Red inset is moving quickly while green inset is still.*

would still allow tracking of fast moving parts of the cloth, but would also stabilise slow moving and static parts.

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.

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 the Light Stage 2.[5] We plan to investigate this possibility further in the future.

Finally, the captured cloth geometry and parameterisation could be used to solve the *inverse cloth simulation problem*, i.e. the fitting of parameters for a cloth simulation algorithm (such as the algorithms by Baraff & Witkin[1] or Choi & Ko[4]) to a sequence of cloth poses acquired with our approach. This would allow us to re-target the acquired cloth parameters to new animations and even new cloth geometries. We believe this to be a fruitful area for future research.

Acknowledgements

We would like to thank David Lowe for providing source code for the SIFT original method, and for valuable comments on the subject. Many thanks to Point Grey Research for providing the Digiclops stereo camera. This work was supported by an NSERC graduate scholarship, the BC Advanced Systems Institute, and the IRIS network of centres of excellence.

References

1. D. Baraff and A. Witkin. Large steps in cloth simulation. In *Proceedings of ACM SIGGRAPH 98*, pages 43–54, 1998. 1, 2, 9
2. J. Beis and David Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Conference on Computer Vision and Pattern Recognition*, pages 1000–1006, 1997. 5
3. 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, 2002. 1
4. K.-J. Choi and H.-S. Ko. Stable but responsive cloth. *ACM Transactions on Graphics (ACM SIGGRAPH 2002)*, 21(3):604–611, 2002. 1, 2, 9
5. 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, 2002. 9
6. I. Guskov. Efficient tracking of regular patterns on non-rigid geometry. In *16th International Conference on Pattern Recognition*, volume 2, pages 1057–1060, 2002. 2, 7
7. I. Guskov and L. Zhukov. Direct pattern tracking on flexible geometry. In *Winter School of Computer Graphics*, pages 203–208, 2002. 2, 7
8. J. Haddon and D. Forsyth. Shading primitives: finding folds and shallow grooves. In *International Conference on Computer Vision*, pages 236–241, 1998. 2
9. D. House and D. Breen. *Cloth Modeling and Animation*, chapter 3, Particle representation of woven fabrics, pages 55–78. A.K. Peters, 2000. 2
10. N. Jojic and T. Huang. Estimating cloth draping parameters from range data. In *International Workshop on Synthetic-Natural Hybrid Coding and 3-D Imaging*, pages 73–76, 1997. 2
11. D. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, pages 1150–1157, 1999. 1, 2, 3
12. B. Lévy and J.-L. Mallet. Non-distorted texture mapping for sheared triangulated meshes. In *Proceedings of ACM SIGGRAPH 98*, pages 343–352, 1998. 6
13. M. Okutomi and T. Kanade. A multiple-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353–363, 1993. 1
14. D. Scharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *Workshop on Stereo and Multi-Baseline Vision*, pages 131–140, 2001. 7