Capturing and Animating Occluded Cloth

Ryan White†⋆ Keenan Crane⋆ D.A. Forsyth⋆
†University of California, Berkeley ⋆ University of Illinois, Urbana Champaign

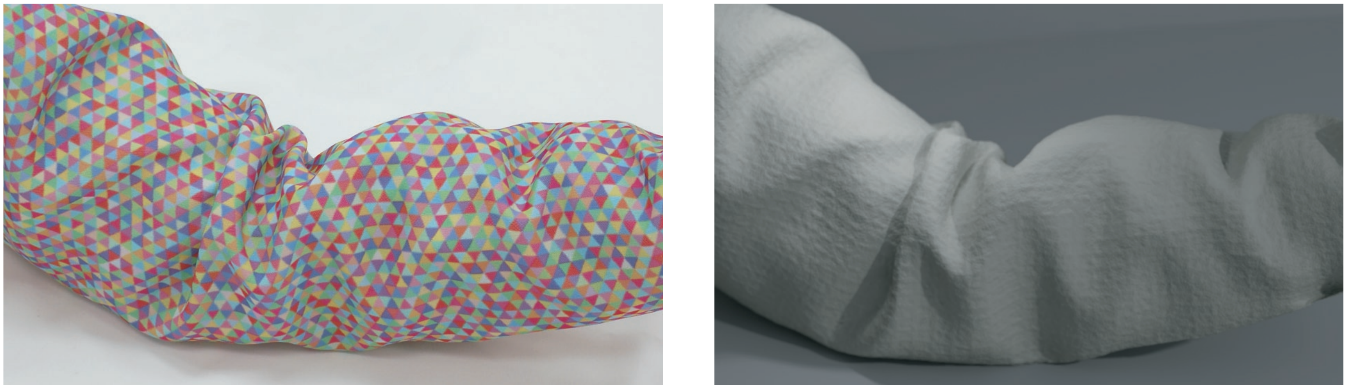

Figure 1: We reconstruct a stationary sleeve using thousands of markers to estimate the geometry (texture added with bump mapping).

Abstract

We capture the shape of moving cloth using a custom set of color markers printed on the surface of the cloth. The output is a sequence of triangle meshes with static connectivity and with detail at the scale of individual markers in both smooth and folded regions. We compute markers' coordinates in space using correspondence across multiple synchronized video cameras. Correspondence is determined from color information in small neighborhoods and refined using a novel strain pruning process. Final correspondence does not require neighborhood information. We use a novel data driven hole-filling technique to fill occluded regions. Our results include several challenging examples: a wrinkled shirt sleeve, a dancing pair of pants, and a rag tossed onto a cup. Finally, we demonstrate that cloth capture is reusable by animating a pair of pants using human motion capture data.

1 Introduction

We capture the motion of cloth using multiple video cameras and specially tailored garments. The resulting surface meshes have an isometric parameterization and maintain static connectivity over time. Over the course of roughly half a dozen papers on cloth capture a prevailing strategy has emerged. First, a pattern is printed on the cloth surface such that small regions of the pattern are unique. Next, correspondence is determined by matching regions across multiple views. The 3D location of a region is determined by intersecting rays through the corresponding observations in the image set (figure 4). Reconstruction is done independently on a frame by frame basis and the resulting data is smoothed and interpolated. Previous work, such as [Scholz et al. 2005], yields pleasing results. Little work has been done to capture garments with folds and scenes with occlusion. In this paper we use **folding** to refer to local phenomena such as wrinkles around a knee and **occlusion** to refer to large scale effects such as one limb blocking the view of another. Folds and occlusion are common, especially when dealing with real garments such as pants where limbs block interior views and cloth collects around joints. Both phenomena are symptoms of the same problem: views of the surface are blocked by other parts of the surface. However, there is a distinction in scale and different methods are required to solve each problem.

When a surface is heavily folded, contiguous visible regions are often small and oddly shaped. In these regions correspondence is essential for detailed reconstruction yet can be challenging to identify. We solve the correspondence problem both by improving the pattern printed on the surface of the cloth and by improving the method used to match regions. Our method gets more information per pixel than previous methods by drawing from the full colorspace instead of a small finite set of colors in the printed pattern. Additionally, because cloth cannot stretch much before ripping, we use strain constraints to eliminate candidates in an iterative search for correspondence. In combination, these two modifications eliminate the need for neighborhood information in the final iteration of our algorithm. As a result, we determine correspondence using regions that are 25 times smaller than in previous work (figure 6).

Many regions on the surface are impossible to observe due to occlusion. We fill these holes using reconstructions of the same surface region taken from other points in time. We found that MeshIK ([Sumner et al. 2005]), a tool originally developed for mesh posing and animation, is appropriate for filling holes in cloth. In fact, MeshIK is well-suited to cloth data and we use it to bind reconstruction of our pants to motion capture data.

We suggest two tools to evaluate marker-based capture systems. The first, *markers per megapixel*, is a measure of efficiency in capture systems. Efficiency is important because camera resolution and bandwidth are expensive: the goal is to get more performance from the same level of equipment. This metric is designed to predict scaling as technology moves from the research lab to the professional studio. The second tool is information theory: we look at the predictive power of different cues in a capture system. By doing simple bit calculations, we direct our design efforts more appropriately.

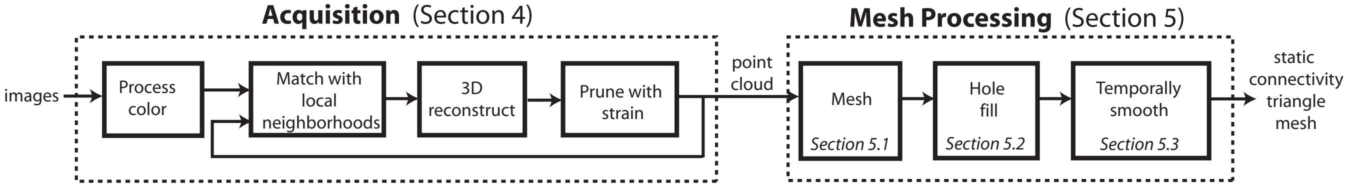

Figure 2: We construct an animated sequence of surface meshes in two stages: acquisition and mesh processing. In acquisition, we convert raw images into a 3D point cloud. In mesh processing, we triangulate the mesh, fill the holes and apply temporal smoothing.

2 Previous Work

Previous work in cloth motion capture has focused on placing high density markers in correspondence between multiple views. The primary challenge is to increase marker density while correctly assigning correspondence between markers. We suggest **markers per megapixel** as an appropriate metric for comparison (figure 3) because it measures the method instead of the equipment. Most high density full frame-rate capture has focused on cloth, however, there has been some recent work enhancing human motion capture [Park and Hodgins 2006]. These methods have far fewer markers per megapixel because they affix individual markers.

When working with cloth, markers are typically painted on the surface. These markers can be broken into three categories: complex surface gradients [Pritchard and Heidrich 2003; Scholz and Magnor 2004; Hasler et al. 2006] (typically detected using SIFT descriptors [Lowe 2004]), intersecting lines [Tanie et al. 2005] and regions of constant color [Guskov and Zhukov 2002; Guskov et al. 2003; Scholz et al. 2005]. Our work falls in the third category: regions of contant color. We evaluate previous work by examining the quality of the reconstructed cloth in still images and video. The most common errors are marker mismatches and are observable in reconstructions by local strain in the reconstructed surface. Overall, we observe that constant color markers perform the best.

[Pritchard and Heidrich 2003] used cloth with unique line drawings as markers. Their work identifies parameterization as one of the key aspects of cloth capture. They use a stereo camera to acquire 3D and SIFT descriptors to establish correspondence. These descriptors are often mismatched and require significant pruning. They introduce a rudimentary strain metric, as measured along the surface, to rule out incorrect matches. While successful, their static reconstructions show numerous correspondence errors.

The real-time system described in [Guskov et al. 2003] introduces markers of constant color, resulting in significantly fewer correspondence errors than in [Pritchard and Heidrich 2003]. This system uses a Kalman smoothing filter and is heavily damped. Additionally, the complexity of the color pattern limits the method to simple geometry.

[Scholz et al. 2005] improve upon [Guskov et al. 2003] by creating a non-repeating grid of color markers. Each marker has five possible colors and all three by three groups are unique. This allows substantially larger sections of cloth and virtually eliminates correspondence errors. Results include a human wearing a shirt and a skirt captured using eight 1K x 1K cameras. However, the range of motion is limited to avoid occlusion (e.g., arms are always held at 90 degrees to the torso). They use thin-plate splines to fill holes.

[White et al. 2005] introduce a combined strain reduction/bundle adjustment that improves the quality of the reconstruction by minimizing strain while reconstructing the 3D location of the points on the surface of the cloth. [White et al. 2006] introduce the use of silhoutte cues to improve reconstruction of difficult to observe regions. While silhouette cues improve reconstruction, hole filling is more effective in many circumstances because it enforces an appropriate prior on the shape of the cloth.

Work	Megapixels	Markers†	**Markers per Megapixel**
Park 2006	48	≤ 350	≤ **7.3**
Tanie 2005	10	407.9	**40**
Guskov 2003	0.9	≤ 136	≤ **148**
Scholz 2005	8	≤ 3500	≤ **434**
Sleeve	15	7557	**504**
Pants	2.5	2405.3	**979**

Figure 3: We suggest **markers per megapixel** as a comparison metric. Because pixels are expensive, efficient use of pixels is necessary. In the pants video, our markers average 56 pixels per camera: the rest of the pixels are consumed by multiple views and background (discussed in section 6.1). †When possible we compare *recovered* markers, however some papers exclusively report total markers.

We make three main contributions: we improve the color pattern and matching procedure to get more information per marker, we introduce strain constraints to simplify correspondence and we create a data driven hole filling technique that splices previously captured cloth into the mesh. As a result, our system is capable of capturing a full range of motion with folding and occlusion.

3 Analyzing Acquisition Methods

To acquire a 3D point cloud of the cloth surface, we print a colored pattern on the cloth, sew it together, and record its motion using multiple synchronized cameras. We then reconstruct the 3D location of surface points by detecting **corresponding** points in multiple views (figure 4).

Our goal is high marker density in the 3D reconstruction – especially in regions with high curvature. To achieve this, we need markers that are both small in scale and highly discriminative. These two goals are in tension: small markers are less discriminative. In addition, we cannot increase camera resolution without bound because camera bandwidth becomes very expensive. As a result, we opt for the smallest markers that we can reliably detect and we *make small markers more distinctive*.

We combine information from three cues to establish correspondence: marker color, neighboring markers and strain constraints in the reconstruction. Marker color and strain constraints are more useful than neighboring markers because they place fewer requirements on local cloth geometry. Specifically, neighboring markers are observed only when the cloth is relatively flat. When the surface is heavily curved only small portions of the surface are visible before the cloth curves out of view. In subsequent sections we adopt the following strategy: maximize information obtained from marker color and *eliminate* the information needed from neighbors.

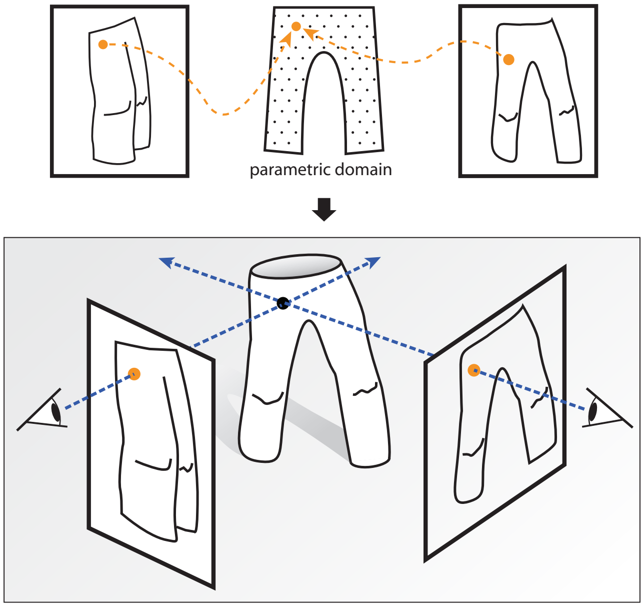

Figure 4: Above: We identify corresponding markers in multiple views in reference to the parametric domain. Below: Once corresponding image points are identified, we intersect eye rays to determine the 3D location.

3.1 Entropy as an Analytical Tool

We optimize our correspondence technique by analyzing the information provided by different cues. In this framework we can accurately minimize the number of neighbors required for correspondence and observe folds better. We can compare our work to previous methods using this framework (figure 6).

It takes $\log_2 M$ bits to determine the identity of each observed marker on a garment with M total markers. Because independent information adds linearly, we can compute the information needed to meet this threshold by adding information from the different cues: color, neighbors and strain. However, structural ambiguities in the pattern subtract information lost to determine which neighbor is which. As a result, we compute our information budget ($\mathscr{I}$) as:

$$\begin{aligned} N &= \text{number of observed neighbors} \\ C &= \text{color information per marker} \\ A &= \text{information lost to structural ambiguities} \\ S &= \text{information gained from strain constraints} \\ \mathscr{I} &= (N+1)*C+S-A \end{aligned}$$

As an example, imagine a rectangular grid of markers and a correspondence method that uses a single immediate neighbor. This neighbor is one of four possible neighbors – thus it takes two bits to specify which neighbor we found ($A = 2$). In this case, the equation reduces to $\mathscr{I} = 2*C-2+S$.

Given almost any structured pattern, we can detect regions by increasing N until $\mathscr{I} > \log_2(M)$ bits. However, larger marker regions have the disadvantage that curvature can cause local occlusions and prevent observation of the entire region. Our best efforts are to improve C – the number of bits from each marker observation. We do this by picking marker color from the full colorspace instead of a small discrete set of colors.

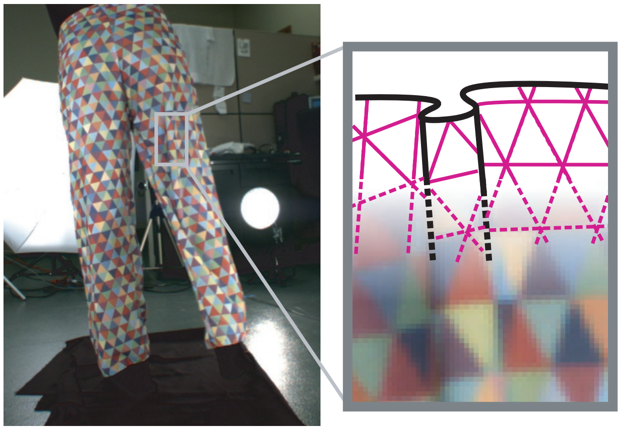

Figure 5: Neighborhood detection methods require that all markers in a fixed geometric pattern in the image neighborhood be neighbors on the cloth. Occluding contours break up neighborhood regions and limit the effectiveness of neighborhood methods in folded regions. We eliminate neighborhood requirements in the final stage of our correspondence algorithm.

3.2 Garment Design and Color Processing

We print a random colored pattern on the surface of cloth in an attempt to maximize the information available per pixel. While our pattern is composed of tesselated triangles (figure 5), any shape that tiles the plane will work (squares and hexagons are also natural choices). To maximize the density of reconstructed points, we print the smallest markers that we can reliably detect. To maximize the information contained in the color of each marker, we print colors that span the gamut of the printer-camera response, then use a gaussian color model (section 4.1).

From a system view, the printer-camera response is a sequence of lossy steps: we generate a color image on a computer, send the image to the printer, pose the cloth, and capture it with a camera. Our experiments suggest that loss is largely attributable to camera response because larger markers produced substantially more information. Illumination is also problematic and takes two forms: direct illumination on a lambertian surface and indirect illumination. To correct for variations in direct illumination, we remove the luminosity component from our color modelling. We do not correct for indirect illumination.

Each marker in the printed pattern has a randomly chosen color, subject to the constraint that neighboring marker colors must be dissimilar. In the recognition stage, we detect markers by comparing colors to a known color. These comparisons must be made in the proper color space: we photograph the surface of the printed cloth with our video cameras to minimize the effect of non-linearities in the printing process.

4 Acquisition

The goal of our acquisition pipeline is to compute correspondence using minimal neighborhoods. We accomplish this through an iterative algorithm where we alternate between computing correspondence and pruning bad matches based on those correspondences. After each iteration we shrink the size of the neighborhood used to match. We start with $N = 3$ and end with $N = 0$. In the final iteration, markers are matched using color and strain alone.

This iterative approach allows us to match without neighborhoods. This is better than label propagation methods. To be successful, propagation methods [Guskov et al. 2003; Scholz et al. 2005; Lin

	1st iteration △ △▽△	2nd iteration △▽△	4th iteration △	[Scholz 2005] ○○○ ○○○ ○○○
Relative Area	15.8	11.8	4.0	100
Color (C)	≥ 5	≥ 5	≥ 5	1.93
Neighbors (N)	3	2	0	8
Strain (S)	0	~ 7	~ 9	–
Ambiguities (A)	1.6	1.6	0	3
Total bits ($\mathscr{I}$)	**18.4**	**20.4**	**14**	**14.4**

Figure 6: Our correspondence algorithm iterates from large to small regions. At each stage, the number of recovered bits must stay above the marker complexity (11.6 bits for our pants). We are able to obtain significantly more information per unit cloth surface area than previous work. See section 3.1 for the entropy equation and appendix B for detailed analysis.

and Liu 2006] require large sections of unoccluded cloth and must stop at occluding contours. As shown in figure 5, occluding contours are both common and difficult to detect. In contrast, our iterative approach relies on strain constraints – which require computing the distance between a point and a line, and color detection – which requires averaging color within a marker. Both of these computations are easier than detecting occluding contours.

We describe our acquisition pipeline, shown in figure 2, below.

Color Processing: We compare observed colors with stored values using a gaussian noise model. Our gaussian noise model has a single free parameter, the variance, which must be computed empirically for each recording setup. This variance determines the color response for the entire setup — smaller variances mean more bits from color. At this stage, we compute color information for each marker and eliminate hypothetical correspondences from further consideration that have large color differences.

Neighborhood Matching: At each iteration, we match highly distinctive neighborhoods by combining information across cues. The size of the neighborhood is chosen so that we get more than enough bits to meet our information budget ($\log_2 M$ bits – typically 11 to 13). The analysis in figure 6 shows that we can set $N = 3$ at the start and continue until $N = 0$. Because the identity of the marker is overspecified, there are few mistakes.

This approach works from flat regions in the first iteration to foldy regions in the later iterations. In the first iteration, we require three neighbors to make a match. In heavily folded regions, often neighboring markers on the image do not neighbor on the surface of the cloth. As such, these regions are not going to match. In contrast, in the last iteration, no neighbors are necessary. Occluding contours, which are common in heavily folded regions, no longer disrupt the matching procedure.

3D Reconstruction: Markers that are observed in multiple views (at least 2) are reconstructed in 3D using textbook methods [Hartley and Zisserman 2000]. We use reprojection error to prune bad matches (reprojection errors average 0.3 pixels and we discard points with errors larger than 2 pixels).

Pruning with Strain: We do two separate strain pruning steps: one on reconstructed 3D points and one on marker observations in each image. The first discards reconstructed points that cause physically unrealistic strain on the surface of the mesh and the second constrains our search for correspondence. Our strain constraint is based on the work of [Provot 1995] who noted that strain in cloth does not exceed 20% in practice. Relaxing the constraint to distances in 3D (surface distance is always more than the distance in 3D), we can use strain to exclude possible correspondences. Strain naturally fits in to our information theory framework: if strain excludes 87.5% of the possible correspondences, then strain has added 3 bits (because $\log_2 (1 - 0.875) = -3$). The strain cue is described in figure 7.

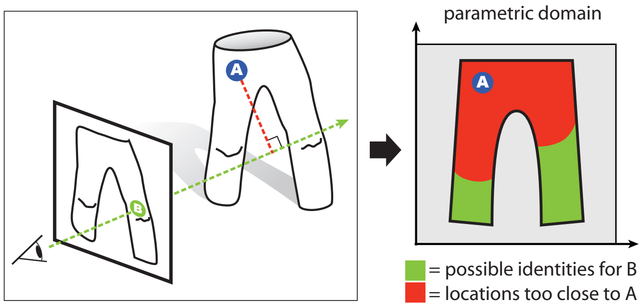

Figure 7: Top: we compute the shortest distance between a known point A and the eye ray through unidentified image point B. **Bottom**: in the parametric domain, this distance restricts the possible identities of B to the green region. The distance from A to B along the surface can be no shorter than the shortest distance in 3D.

4.1 Representation

To find correspondence, we match each image marker to a marker in the parametric domain. To do this, we define affinities $a_{i,j}$ between image marker i and parametric marker j. Each affinity is a product over different cues. We write $c_{i,j} \in [0,1]$ for the color affinity, $d(C_i, C_j)$ for the color distance between i and j, $s_{i,j} \in \{0,1\}$ for the strain constraint, n_i for the image neighbors of marker i and N_j for the parametric neighbors of marker j:

$$\begin{aligned} a_{i,j} &= c_{i,j}\, s_{i,j} \prod_{l \in N_j} \max_{k \in n_i} c_{k,l} \\ c_{i,j} &= \exp\left(-\frac{d(C_i,C_j)^2}{2\,\sigma^2}\right) \\ s_{i,j} &= \begin{cases} 0 & \text{if a strain constraint is violated} \\ 1 & \text{if not} \end{cases} \end{aligned}$$

When only one affinity for image marker i is above a theshold, then we declare a correspondence. Initially, we learned this threshold from labelled data, but we found that changing it by several orders of magnitude had little effect on our results. Subsequently, we use the value $10^{-5(N+1)}$ where N is the number of neighbors.

5 Mesh Processing

In the acquisition process, occlusion inevitably creates holes in the reconstructed mesh (figure 8). One would like to fill these holes with real cloth. One of our major contributions is a data driven approach to hole filling: we fill holes with previously observed sections of cloth. Our work differs from [Anguelov et al. 2005] because our hole filling procedure does not assume a skeleton that drives the surface and our procedure estimates a single coefficient per example.

This hole filling procedure has a number of requirements: the missing section needs to be replaced by a section with the same topology; the new section needs to obey a number of point constraints around the edge of the hole, and the splicing method should respect properties of cloth (specifically strain). We select a reconstruction technique based on deformation gradients [Sumner and Popovic 2004]. In this approach, we fit deformation gradients for the missing section to a combination of deformation gradients in

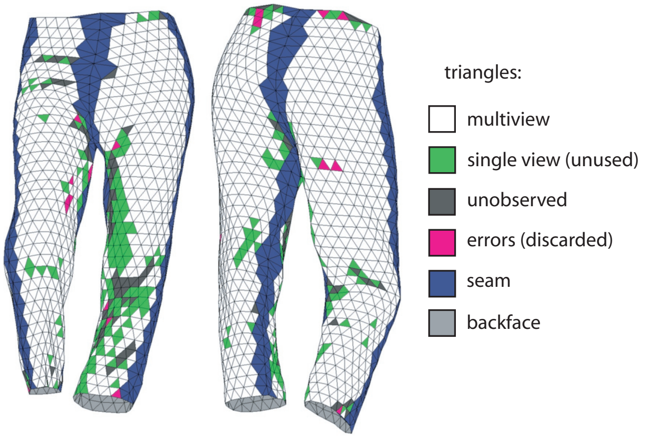

Figure 8: Occlusion is inveitable when capturing highly articulated objects. In this reconstruction, the inner thigh region of the left leg is difficult to observe because the right leg shields it from view. Regions that contain errors or that are seen in two or fewer views must be filled afterwards. Errors are detected using reprojection error and strain.

other observed sections. Then, we reconstruct the point locations from the deformation gradients.

This procedure has a number of advantages. First, deformation gradients naturally yield cloth like properties. Deformation gradients are the transformation matrix between triangles in two poses of the mesh. By penalizing elements that deviate in this matrix, we have a fairly direct penalty on large changes in scale or strain. In contrast, methods based on the Laplacian of the mesh ([Sorkine et al. 2004]) do little to penalize these strains and can show many artifacts around the edge of the mesh. Second, deformation gradients can be converted into vertex locations by inverting a linear system, allowing us to specify vertex locations as constraints. Methods such as [Lipman et al. 2005] don't allow vertex constraints.

Our subsequent discussion is divided into three sections: constructing a mesh from the point cloud, filling the holes in the mesh using deformation gradients, and temporally smoothing the results.

5.1 Meshing and Seams

We produce a mesh by forming equilateral triangles for sections of cloth that are printed with a contiguous pattern by referencing the triangle stucture of markers on the cloth. Our recovered markers are at the center of each triangle – so we average points to get out the vertices and subsequently the original mesh. We insert artificial points where two pieces of fabric come together. These points are created once per garment by hand clicking on photos of the each seam. The 3D locations of these points are recreated in each frame by averaging points near the seam.

5.2 Hole Filling

We use occlusion free meshes from other frames to automatically interpolate holes. For each hole in each frame, we cut out the missing region plus a ring of two triangles around the region. We select a set of examples of the enlarged region, then use MeshIK ([Sumner et al. 2005]) to reconstruct the surface. MeshIK works by choosing a combination of deformation gradients from the examples and then solving for the missing point locations. We use the points from the ring of known triangles around the hole as constraints in MeshIK.

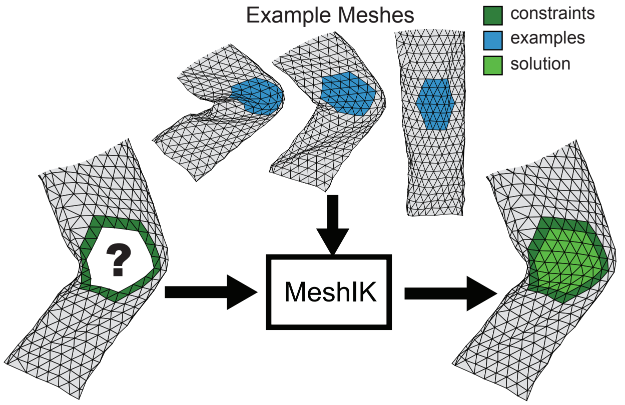

Figure 9: Holes are filled with a combination of cloth sections observed in other frames. (In reality, we use a ring of two triangles as constraints)

The most restrictive aspect of MeshIK is that it requires example meshes without holes. In practice, we *never* observe complete example meshes – each mesh is missing some triangles. These holes appear in different places in different meshes and we create complete meshes in an iterative method. First, we fill all holes with a naive linear algorithm (specifically, we triangulate across gaps and use barycentric coordinates to place the missing points – this gets the job done, but works poorly). Then, we do another pass through all the data, where we replace the linear sections with sections created using MeshIK on the linearly filled data. To downweight the linear data, we select the examples with the highest percentage of viewed points in the missing section. These frames are then used as examples in MeshIK to hole fill in the rest of the sequence.

For the pants capture, we iteratively refine a set of 27 extreme poses which were captured specifically for filling holes. The advantage of this apporach is that the example poses are chosen to capture the relevant degrees of freedom – yielding better results. For the cloth toss sequence, we chose the simpler approach: iteratively refine the entire sequence.

5.3 Smoothing

We introduce flexibility preserving smoothing – a method similar to anisotropic diffusion [Perona and Malik 1990] that smoothes near-rigid movement without effecting flexible deformation. Typical temporal smoothing is dangerous because fast non-rigid movements can easily become physically implausible when blurred over time. However, because fast non-rigid regions of the cloth are complex, small temporal errors are often difficult to notice. In contrast, small errors in regions of the cloth that move rigidly are typically easy to observe. As a result we use flexibility preserving smoothing, a procedure that smoothes rigid movement more heavily than non-rigid movement. To do this, we take a local region around each vertex in the mesh (typically 25 points) and compute a rigid transformation to previous and subsequent frames. Aligning the regions with this transformation, we compute the movement of the vertices in this reference frame as a proxy for rigidity. Large variations in location indicate non-rigid movement and consequently receive little smoothing. Smaller variations indicates rigid movement and benefit from more substantial smoothing. We use a size adjusted gaussian to smooth in this reference frame.

6 Results and Applications

Our video sequences were taken with synchronized firewire cameras (Foculus FO214C) with a capture resolution of 640 x 480 and a capture rate of 24 frames per second. Our still captures were taken using a digital SLR camera and then downsampled to approximate

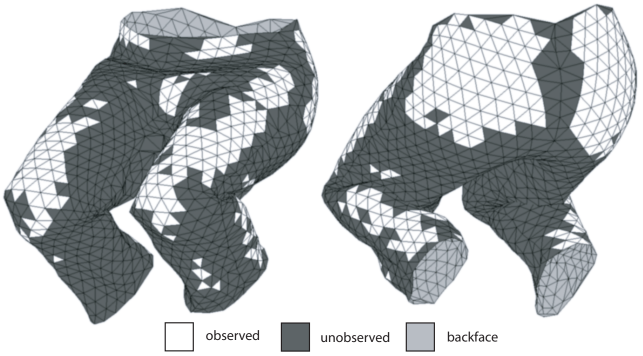

Figure 10: Our hole filling works in extreme circumstances. In this frame, 73% of the mesh is unobserved and inserted using MeshIK based hole filling. This frame is unusal: only 22% of the surface is unobserved in an average frame.

available video resolutions. We use the automated calibration technique in [White and Forsyth 2005], but any standard calibration will work ([Zhang 2002] and [Bouguet 2005] are good choices). In the pants sequences, we used seven lights totalling 1550 Watts to illuminate the scene. Adequate lighting is critical: from our experience fewer lights degrade performance due to harsh shadows and dim lighting causes motion blur through slower shutter speeds. Our cloth was printed by a digital mail order fabric printing service. On a P4 2.4 GHz machine, acquisition takes roughly 6 minutes and mesh processing 2 minutes per frame. Code is written in MATLAB.

6.1 Capture Results

Our capture results are best evaluated by looking at our video and figures 1,12,13. However, to compare against other capture techniques, it is also necessary to evaluate on several numerical criteria for each capture session:

	cloth drop	pants dance	table cloth	sleeve†
# cameras	6	8	18	10
resolution	640x480	640x480	900x600	1500x1000
total markers	853	3060	4793	13465
recovered	819	2405	4361	7557
percentage	96%	79%	91%	56%
bits needed	9.7	11.6	12.2	13.7
color bits	6.1	5.1	6.4	4.5
strain bits	9.1	9.4	11.4	~ 6.6

†The sleeve example is unique because it was one of the first items we captured. Much of the cloth is in contact with the floor and unobservable – yielding fewer bits of strain. In addition, the camera images were not output in a linear color space, reducing the number of color bits. As a result, we terminated the correspondence algorithm at $N = 2$.

Our pants animation is by far the most challenging, and we analyze some of the details a little more closely. With an average of 2405 observed markers, there were 979 3D markers per megapixel. If we factor out the pixels lost to background, we get 3500 3D markers per foreground megapixel or 282 foreground pixels per recovered 3D marker. Our marker observations average 56 pixels per marker per image. There are several reasons for the discrepancy: markers must be observed multiple times (approx 44% of 3D markers are observed in 3 or more views), some markers are observed but not reconstructed (due to errors or missing correspondence), and many pixels are not considered part of a marker: they lie in heavy shadow or occupy the edge between two markers (approx 35% of pixels).

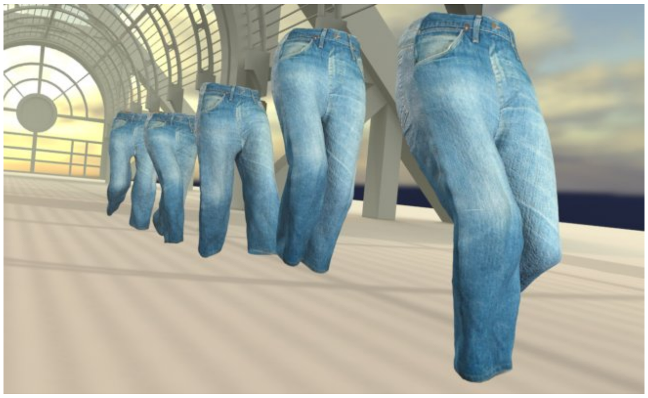

Figure 11: We use MeshIK to bind captured cloth to human motion capture data using 6 joints in the mocap data.

6.2 Retargeting Animations

We use a small set of captured frames (the previous basis of the 27 examples) in combination with MeshIK to skin skeletal human motion capture data (figure 11). This approach covers a reasonably large range of motion, but ignores cloth dynamics.

The largest challenge is that captured cloth meshes contain only points on the cloth surface, so we do not know joint locations. Instead, we insert proxy points for knee and hip joints in each of our basis meshes. These points are then connected to a small set of nearby triangles in the original mesh. For each frame of animation we set the proxy points' locations according to joint angles in the skeletal mocap data. The resulting transformed joints are used as constraint points in MeshIK, which produces the final output meshes. Using our MATLAB implementation of MeshIK, this process takes around 5-10 seconds per frame.

We use the same 27 bases poses for MeshIK based reconstruction. In order for a small basis to adequately express a full range of motion, each basis pose must be an *extreme* configuration. For simple objects such as a cylinder, a small bend (for example) is sufficient to extrapolate to a larger bend [Sumner et al. 2005]. However, for pants the relationship is more complex: the fact that no folding occurs in a small bend does not imply that folding will be absent in a larger bend. Conversely, if a decent amount of folding occurs in a small bend, we do not expect extreme folds in a corresponding larger bend. As a result, MeshIK is most useful when a basis is carefully chosen to prevent extrapolation artifacts.

One drawback to our approach is the loss of secondary kinematic motion, such as the sway of loose cloth. Because MeshIK does not use velocity information, the resulting animation appears damped.

7 Discussion

We have brought cloth capture from constrained laboratory examples to real settings by providing robust methods for dealing with occlusion and folding. Like human motion capture, this tool requires significant engineering effort. Camera setup and calibration are time consuming and the equipment is costly. However, once these obstacles have been overcome, capturing large amounts of data is relatively easy. So that other researchers can benefit from our work, we are releasing our capture data at *http://www.ryanmwhite.com/data*. In our video, we show some of the uses of this data, including editing using [Kircher and Garland 2006] and posing using [Sumner et al. 2005].

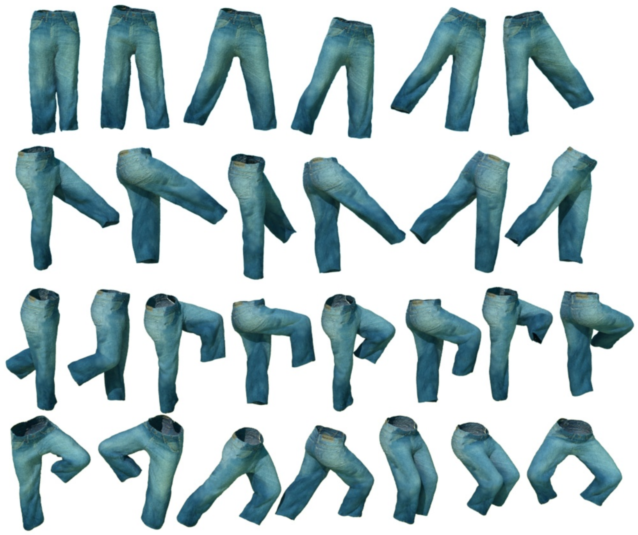

Figure 12: We collected twenty seven basis poses covering the major modes of deformation in the pants. These poses were used in creating the motion capture sequence and for hole filling in the captured sequences.

Future work in cloth capture should involve more cameras, higher resolution (leading to smaller denser markers), different garments and different materials. We plan to pursue more tools to edit and repurpose captured data.

Finally, we would like to conclude with a discussion about cloth capture in the context of other cloth animation techniques. Simulation and image based rendering both provide methods to generate animation of cloth (a limited simulation list includes [House and Breen 2000; Terzopoulos et al. 1987; Choi and Ko 2002; Bridson et al. 2003; Baraff et al. 2003] and a limited image based rendering list includes [Bradley et al. 2005; White and Forsyth 2006; Lin and Liu 2006; Scholz and Magnor 2006]). These methods have several advantages: simulation gives significant user control and produces higher resolution meshes while image based rendering techniques produce more accurate illumination. However, capturing large amounts of data is far easier than simulating large amounts of data and provides more control than image based rendering. Common simulation complaints include long computation times, significant parameter tweaking and tangling. In contrast, capture is relatively quick (our code is 8 minutes per frame in MATLAB); parameters are set by selecting the type of cloth [Bhat et al. 2003] and tangling is relatively uncommon. Cloth capture makes it easy to capture large amounts of cloth, including fast light cloths that create instabilities in simulation. An added attraction of cloth capture is that complex interaction between the cloth and the body is recorded without complicated human models.

Acknowledgements

We thank Jai Vasanth and Anthony Lobay for early support of this project, Scott Kircher and Robert Sumner for providing mesh editing binaries, and Sony Computer Entertainment America for supplying human motion capture data. This work was supported in part by a Department of Homeland Security Fellowship and an ATI Graduate Research Fellowship.

References

ANGUELOV, D., SRINIVASAN, P., KOLLER, D., THRUN, S., RODGERS, J., AND DAVIS, J. 2005. Scape: shape completion and animation of people. In *SIGGRAPH*.

BARAFF, D., WITKIN, A., AND KASS, M. 2003. Untangling cloth. In *SIGGRAPH*.

BHAT, K., TWIGG, C., HODGINS, J. K., KHOSLA, P., POPOVIC, Z., AND SEITZ, S. 2003. Estimating cloth simulation parameters from video. In *SCA*.

BOUGUET, J.-Y., 2005. Camera calibration toolbox for matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/.

BRADLEY, D., ROTH, G., AND BOSE, P. 2005. Augmented clothing. In *Graphics Interface*.

BRIDSON, R., MARINO, S., AND FEDKIW, R. 2003. Simulation of clothing with folds and wrinkles. In *SCA*.

CHOI, K.-J., AND KO, H.-S. 2002. Stable but responsive cloth. In *SIGGRAPH*.

COVER, T. M., AND THOMAS, J. A. 1991. *Information Theory*. John Wiley and Sons.

FORSYTH, D., AND PONCE, J. 2002. *Computer Vision: a modern approach*. Prentice-Hall.

GUSKOV, I., AND ZHUKOV, L. 2002. Direct pattern tracking on flexible geometry. In *WSCG*.

GUSKOV, I., KLIBANOV, S., AND BRYANT, B. 2003. Trackable surfaces. In *SCA*.

HARTLEY, R., AND ZISSERMAN, A. 2000. *Multiple View Geometry*. Cambridge University Press.

HASLER, N., ASBACH, M., ROSENHAHN, B., OHM, J.-R., AND SEIDEL, H.-P. 2006. Physically based tracking of cloth. In *VMV*.

HOUSE, D., AND BREEN, D., Eds. 2000. *Cloth Modelling and Animation*. A.K. Peters.

KIRCHER, S., AND GARLAND, M. 2006. Editing arbitrarily deforming surface animations. In *SIGGRAPH*.

LIN, W.-C., AND LIU, Y. 2006. Tracking dynamic near-regular textures under occlusion and rapid movements. In *ECCV*.

LIPMAN, Y., SORKINE, O., LEVIN, D., AND COHEN-OR, D. 2005. Linear rotation-invariant coordinates for meshes. In *SIGGRAPH*.

LOWE, D. 2004. Distinctive image features from scale-invariant keypoints. *IJCV*.

PARK, S. I., AND HODGINS, J. K. 2006. Capturing and animating skin deformation in human motion. In *SIGGRAPH*.

PERONA, P., AND MALIK, J. 1990. Scale-space and edge detection using anisotropic diffusion. *PAMI*.

PRITCHARD, D., AND HEIDRICH, W. 2003. Cloth motion capture. *Eurographics*.

PROVOT, X. 1995. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Graphics Interface*.

SCHOLZ, V., AND MAGNOR, M. A. 2004. Cloth motion from optical flow. In *VMV*.

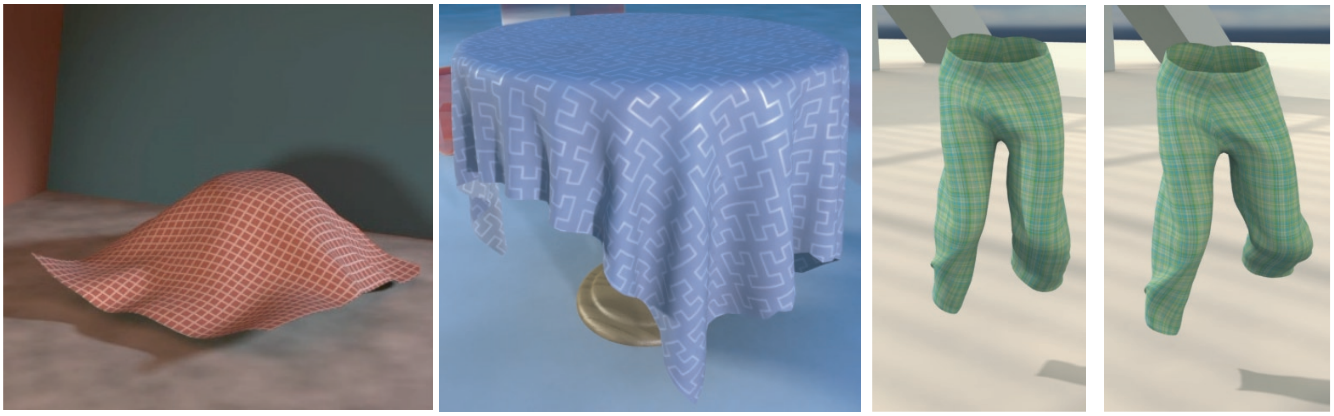

Figure 13: We reconstruct cloth being tossed over a cup, a tablecloth and a pair of pants (shown in the middle of a jump). See the video for a better view of the results.

SCHOLZ, V., AND MAGNOR, M. 2006. Texture replacement of garments in monocular video sequences. In *Rendering Techniques*.

SCHOLZ, V., STICH, T., KECKEISEN, M., WACKER, M., AND MAGNOR, M. 2005. Garment motion capture using color-coded patterns. In *Eurographics*.

SORKINE, O., COHEN-OR, D., LIPMAN, Y., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. 2004. Laplacian surface editing. In *Symposium of Geometry Processing*.

SUMNER, R. W., AND POPOVIC, J. 2004. Deformation transfer for triangle meshes. In *SIGGRAPH*.

SUMNER, R. W., ZWICKER, M., GOTSMAN, C., AND POPOVIC, J. 2005. Mesh-based inverse kinematics. In *SIGGRAPH*.

TANIE, H., YAMANE, K., AND NAKAMURA, Y. 2005. High marker density motion capture by retroreflective mesh suit. In *ICRA*.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically deformable models. In *SIGGRAPH*.

WHITE, R., AND FORSYTH, D., 2005. Deforming objects provide better camera calibration. UC Berkeley Technical Report.

WHITE, R., AND FORSYTH, D. 2006. Retexturing single views using texture and shading. In *ECCV*.

WHITE, R., LOBAY, A., AND FORSYTH, D., 2005. Cloth capture. UC Berkeley Technical Report.

WHITE, R., FORSYTH, D., AND VASANTH, J., 2006. Capturing real folds in cloth. UC Berkeley Technical Report.

ZHANG, Z. 2002. A flexible new technique for camera calibration. *PAMI*.

A Image Processing

We do some pre-processing to get marker locations and connectivity from raw images. We recommend readers unfamiliar with these techniques refer to [Forsyth and Ponce 2002]. We start by converting each image to HSV, disregarding the luminosity (V) and using polar coordinates to compute distances in hue and saturation. To detect markers, our code looks for uniformly colored blobs in two stages: first regions are built by growing neighborhoods based on similarity between pixels. This method is sensitive to image noise and can produce oversized regions when the color boundaries are smoothed. The second stage takes the center of mass of each blob from the first stage, computes the mean color and grows a region based on distance to the mean color (it is computationally intractable to use this as the first stage of the blob detection). The process is iterated for increasing thresholds on the affinity value in the first stage, using the portions of the image where detection failed in previous stages. Finally, blobs are thresholded based on size.

Next, we need to determine the neighborhood relationships. For each marker, we construct a covariate neighborhood (a fitted ellipse) and vote for links to the three closest markers with similar covariate neighborhoods. This measures distances appropriately in parts of the scene where the cloth is receding from view and discourages links between markers with wildly different tilts. All links that receive two votes (one from either side) are kept while the rest are discarded. Links that bridge markers with conflicting color information are also discarded (typically on internal silhouettes).

B Entropy Comparison

For more reading on information theory, consult [Cover and Thomas 1991]. Our analysis is based on the information entropy definition: $H(X) = -\sum_{i=1}^{n} p(x_i) \cdot \log_2 x_i$.

For [Scholz et al. 2005], the equation in section 3.1 is reduced to $\mathscr{I} = 9 * C - A$ because they use 8 neighbors and no strain constraints. They use 5 colors which, without errors, yields $C = \log_2 5$ bits per marker. They cite an error rate of five to ten percent. As a result, they recover anywhere from 1.65 to 2.04 bits per marker. In our comparison, we use $C = 1.93$ bits for color information from their method (five percent error, with equal probabilities for all remaining choices). Note that this is effectively less than four colors! Second, we compute structural ambiguities in their method which account for uncertainty in observations. The orientation of the surface is unknown, yielding four possible directions, or two bits of structural ambiguity. Second, in their paper, they say that oblique views cause another bit of uncertainty. As a result $A = 3$ bits.

For our work, C is an empirical observation. Depending on the lighting and camera configuration, we get anywhere from 5 to 7 bits. We use the conservative estimate of $C = 5$ bits per marker. Second, our mesh is triangular and there are three possible neighborhood rotations, yielding $A = \log_2 3 = 1.59$ bits of structural ambiguity. When neighborhoods are not used, there is no structural ambiguity. Strain information is difficult to compute and depends on the geometry of the surface and the orientation of the camera. In most cases, we observe more than 9 bits of strain information.