Cloth Motion from Optical Flow

Volker Scholz, Marcus A. Magnor

Max-Planck-Institut für Informatik
Graphics - Optics - Vision
Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany
Email: {vscholz,magnor}@mpi-sb.mpg.de

Abstract

This paper presents an algorithm for capturing the motion of deformable surfaces, in particular textured cloth. In a calibrated multi-camera setup, the optical flow between consecutive video frames is determined and 3D scene flow is computed. We use a deformable surface model with constraints for vertex distances and curvature to increase the robustness of the optical flow measurements. Tracking errors in long video sequences are corrected by a silhouette matching procedure. We present results for synthetic cloth simulations and discuss how they can be extended to real-world footage.

1 Introduction

Cloth simulation is a well-known subject in computer graphics. A good survey of the available methods is given in [1]. State-of-the-art simulations use particle systems with non-physical parameters which have to be tuned manually in order to achieve real-world material characteristics. Recent work [2] focuses on the preservation of folds and wrinkles during collision detection, showing that current simulations are still not physically accurate. The physical models used in simulation are rather complex and the results should be compared to real-world data. The acquisition of real cloth motion could be helpful for this purpose.
We are also not aware of realistic simulations of the interaction of cloth and turbulent wind flow. Flag-flapping is one of the most complex phenomena in hydrodynamics [3]. Motion capture circumvents the problem to simulate this type of motion.

2 Related Work

A general image-based approach to cloth motion was introduced by Pritchard and Heidrich [4]. Cloth geometry is obtained with stereo cameras with a low frame rate (10 frames per second, fps). The surface parameterisation with texture coordinates is performed using a feature matching technique. The results they present are based on a short frame sequence. The authors mention that the animation lacks frame-to-frame coherence because reconstruction is performed for every frame separately. This is one way to cope with rapid movements if only a low frame rate is available. Our method can handle rapid motion with higher frame rates and achieves frame-to-frame coherence.
Another approach is to estimate the parameters of a cloth simulation by adjusting the simulation results to real-world footage [5]. This is useful to avoid parameter-tuning by hand but captures motion only in a qualitative way.
The rest of the paper is structured as follows. In Section 3, we give a short overview of our algorithm. Sections 4–6 describe the components of the algorithm in detail. Results are presented in Section 7, before we conclude in Section 8.

3 Algorithm Overview

We propose a new approach and use *optical flow* as the main component in our reconstruction algorithm. A prerequisite for the use of optical flow is a richly detailed cloth texture. If the initial position of the cloth is known, vertex motion can be tracked from frame to frame using optical flow information. Given a high frame rate, optical flow between consecutive frames is suitable to track rapid motion. Optical flow is a well-known problem and the quality of the available algorithms is sufficient

for practical applications.
Optical flow is not well-defined in poorly textured regions so we have to interpolate these regions. We employ a deformable cloth model for this purpose. The model also makes the algorithm more robust against optical flow errors. Adding the interframe flow vectors over a long frame sequence is not feasible because flow errors accumulate and tracking errors are introduced. In order to address this problem, the cloth silhouette is determined in the video images and the boundary vertices of the model are matched to the silhouette in a correction step.

4 Optical Flow

Optical flow is the apparent motion of brightness patterns between two frames of an image sequence. In [6] and [7] several optical flow algorithms are evaluated. The method by Lucas and Kanade [8] shows the best accuracy and noise tolerance. It was originally a stereo matching technique but now is mainly used for optical flow. We give a short description of the algorithm. The basic assumptions of the algorithm are:

- the brightness of the image pixels remains constant between successive video frames.
- the motion can be described by a pure translation in the image plane.

This can be summarized as

$$I(\mathbf{x}, t) = I(\mathbf{x} + \mathbf{u}, t + dt) \tag{1}$$

where I denotes image brightness, $\mathbf{x}$ the pixel location, $\mathbf{u}$ the pixel translation and t the time. The first order Taylor series expansion of I is given by

$$I(\mathbf{x}+\mathbf{u}, t + dt) = I(\mathbf{x}, t)+\nabla I \cdot \mathbf{u}+\frac{\partial I}{\partial t} \cdot dt \tag{2}$$

where ∇I denotes the spatial image gradient. Eqs. (1) and (2) lead to the optical flow constraint equation:

$$\nabla I \cdot \mathbf{u}+\frac{\partial I}{\partial t} \cdot dt = 0 \tag{3}$$

The Lucas-Kanade algorithm minimizes the left hand side of Eq. (3) in a window W around a pixel with respect to $\mathbf{u}$:

$$\sum_{\mathbf{x} \in W} w^2(\mathbf{x})(\nabla I \cdot \mathbf{u}+\frac{\partial I}{\partial t} \cdot dt)^2 \tag{4}$$

$w : W \rightarrow \mathbb{R}$ denotes a Gaussian weight function. This function is minimized with the Newton-Raphson method. The summation window W increases the robustness of the method, as the pixel translation $\mathbf{u}$ is assumed to be constant inside the window. The algorithm can find pixel displacements in the subpixel range. Larger pixel displacements are handled by a multiresolution scheme on a Gaussian image pyramid. It is also a strategy to find the global minimum of the multi-modal cost function. The implementation details can be found in [9].
In order to make the flow computation more reliable, we compute the flow in a projected rectangular patch around the vertex positions of our model and apply a median filter to the flow vectors in the patch for outlier removal. The size of the window W is 5x5 pixels.
Two-dimensional optical flow is a projection of three-dimensional scene flow (vertex motion) to the image plane [10]. If the camera calibration and the initial vertex positions of the surface are known, the scene flow can be determined from several camera views. Vertex motion is optimized together with a deformable model which we describe in the next section.

5 Deformable Model

Deformable models have successfully been applied to motion tracking problems [11, 12]. The goal is to provide additional constraints which make the motion estimation more robust. In [11], the number of degrees of freedom of a FEM model is reduced by an analysis of the vibration modes. This approach is suitable for simple shapes. Ref. [12] introduces superquadrics for deformable surfaces, a model which is targeted towards closed surfaces.
Our cloth model consists of a rectangular grid of vertices. Cloth deformation can be described in terms of three basic deformations: stretching, bending and shearing [1]. Stretching is almost negligible for non-elastic materials and can be used as a constraint for the vertex positions. We propose the following energy function which penalises compression and stretching of horizontally or vertically adjacent vertices $\mathbf{p}_i$, $\mathbf{p}_j$, where d_0 denotes the initial vertex distance (Fig. 1):

$$E_{stretch} = \sum_{i,j} (\frac{\|\mathbf{p}_i - \mathbf{p}_j\| - d_0}{d_0})^2 \tag{5}$$

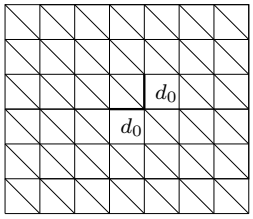

Figure 1: Uniform triangulation with distance constraint for adjacent vertices.

The diagonal mesh edges are not considered here as this would constrain shearing deformations. It is also reasonable to assume a smooth cloth surface. We use a discrete version of thin plate spline energy [13]

$$E_{curv} = \sum_{t_a,t_b} l_e \cdot (\mathbf{n}_{t_a} - \mathbf{n}_{t_b})^2 \qquad (6)$$

where $\mathbf{n}_{t_a}$ and $\mathbf{n}_{t_b}$ are the normalised normals [1] of adjacent triangles t_a and t_b and l_e is the length of their common edge (Fig. 2). The summation is done over all triangle pairs with an adjacent edge. $E_{stretch}$ and E_{curv} are similar to the energy terms

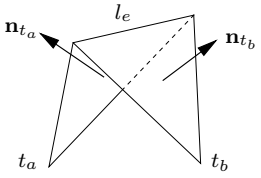

Figure 2: Bending constraint for adjacent triangles

used in cloth simulation [1].
The vertex error of the model at time t with respect to optical flow to the next video frame at time $t+1$ is given as

$$\begin{aligned} \epsilon_i^c &= \sqrt{(\frac{\mathbf{m}_0^c \cdot \mathbf{p}_i}{\mathbf{m}_3^c \cdot \mathbf{p}_i} - u_i)^2 + (\frac{\mathbf{m}_1^c \cdot \mathbf{p}_i}{\mathbf{m}_3^c \cdot \mathbf{p}_i} - v_i)^2} \\ E_{flow} &= \sum_{c \in C} \sum_i \epsilon_i^c \qquad (7) \end{aligned}$$

where $\mathbf{m}_0^c - \mathbf{m}_3^c$ are the rows of the 4x4 calibration matrix $\mathcal{M}^c$ of camera $c \in C$ and u_i, v_i the desired vertex projections resulting from the optical flow algorithm. The fractions in Eq. (7) represent perspective projection from vertex to image coordinates. The inner sum is computed for all visible vertices in a camera view. Visibility is determined with a z-buffer test.
All energy terms are combined with weighting factors into one energy function

$$E = E_{flow} + \lambda \cdot E_{stretch} + \mu \cdot E_{curv} \qquad (8)$$

and optimized with the Polak-Ribière conjugate gradient method [14]. The energy function gradient can be computed analytically. If the interframe differences are small (high frame rate), conjugate gradient minimization is suitable because we are already near the optimum.

6 Silhouette Matching

The optical flow errors accumulate over longer frame sequences and the 3D model has to be readjusted to the images. For this purpose, we determine the cloth silhouettes in the input images by a simple contour following algorithm, Fig. 3 [15]. With our synthetic test data, the contour can be directly determined from the input images. Video data would require background subtraction as a preprocessing step. For every vertex on the mesh boundary, the nearest contour point is determined. Its position corresponds to the new vertex position u_i, v_i in Eq. (7), i.e. the contour generates *flow* vectors for the boundary vertices. The inner vertices do not contribute to E_{flow}, they are only constrained by $E_{stretch}$ and E_{curv}. The mesh is adjusted by optimizing the energy function from Eq. (8). Observe that the contour matching procedure is only correct if the boundary vertices stay on the cloth silhouette during the whole animation sequence.

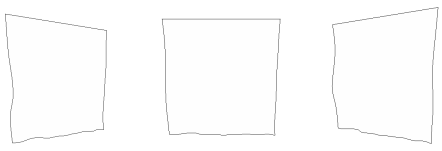

Figure 3: Cloth silhouettes in three camera views.

[1] We omit normalisation in our implementation as the gradient would get too complex. This approximation assumes constant triangle areas, i.e. small shear but yields smooth results.

7 Results

Experiments with our current video cameras with a resolution of 320x240 pixels have shown that a higher resolution is necessary for an accurate surface reconstruction from optical flow. For this reason, we use synthetic data generated with a cloth simulator [16]. We are confident that the method can be applied to video data from new high-resolution cameras.

The sequence consists of 300 frames, recorded by three camera views which are separated by rotations of 30 degrees from each other with a resolution of 640x480 pixels. We simulate a frame rate of 30 fps. Fig. 5 shows the test sequence, a piece of cloth flapping in a breeze. Only two camera views are shown, the third camera is located in the middle of the other two cameras. The cloth texture was acquired with a still image digital camera and a directional light source was added to the scene. The resulting shading effects are challenging for the optical flow algorithm because the brightness constancy constraint is violated.

The triangle mesh used for reconstruction has a resolution of 33x33 vertices, i.e. the optimization problem has n=3267 variables. The parameters λ and μ in Eq. (8) are tuned manually. Values near the optimum are sufficient for a satisfactory result. We choose μ so that the bending deformation of the cloth is preserved but high-frequent noise is removed.

The average computation time for one frame of the sequence is 53 seconds on a Pentium IV 2.4 GHz. The different stages of the algorithm have the following average time requirements:

- optical flow computation: 18 s
- vertex flow optimization: 17 s
- contour matching: 5 s
- correction step optimization: 13 s

Fig. 6 shows the computed optical flow vector fields and Fig. 7 the reconstructed surface. In Fig. 8 the tracking accuracy of the algorithm over the whole sequence is depicted. The object silhouettes are preserved by the contour matching algorithm, although the matching is ambiguous near the cloth corners. In Fig. 9, a reconstructed frame and the difference to the corresponding input image are shown. The difference image shows a pixel displacement in the range of 2-3 pixels between reconstruction and input image. The error is concentrated in the edges of the image.

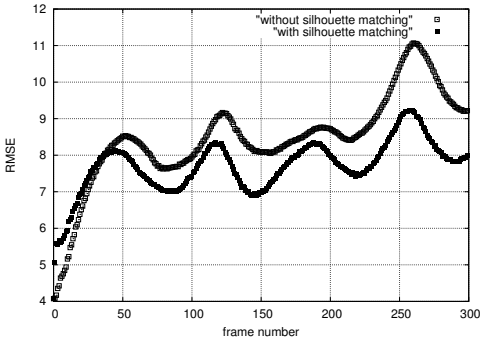

Figure 4: Per-pixel error with and without silhouette matching. Pixel values range from 0 to 255.

In Fig. 4 the root mean squared error (RMSE) between the input images and the reconstructed images over the whole frame sequence is depicted. The mean is computed over all camera views and colour channels. We have added the upper curve showing the behaviour without silhouette matching. The average error is higher in this case. The curve for our method shows several local minima corresponding to frames where the cloth motion is minimal (the test sequence contains periodic motion). This shows that the deformable model constraints are able to reduce the error at these points (E_{flow} is negligible in this case). The average error is constant for about 200 frames and grows towards the end of the sequence. This result is confirmed by the visual quality of the reconstruction. A result movie can be downloaded from our webpage. [2]

8 Conclusion and Future Work

We have presented a method that is capable of reconstructing rapid wide-range cloth motion for synthetic test data. A combination of optical flow and a deformable model is used to track motion robustly. We obtain photo-realistic results and can track motion over a long frame sequence. Frame-to-frame coherence is achieved by our incremental approach using optical flow, adding to the realism of the captured motion. The approach is applicable to cloth with richly detailed texture.

In our current implementation we assume a flat ini-

[2] http://www.mpi-sb.mpg.de/~vscholz/vmv04/result.mpg

tial cloth position. We plan to determine this position by matching our textured model to the corresponding video frames. Our results are based on synthetic test data, so the next step is the application to video data from high-resolution cameras for sequences with turbulent wind flow. This requires background subtraction for the silhouette matching step. Other areas to investigate are the exploitation of temporal coherence of the data and of shape-from-texture techniques to increase the accuracy of the method.

References

[1] D. H. House and D. E. Breen (Eds.), "*Cloth Modeling and Animation*", AK Peters, Ltd., Natick, MA, 2000.

[2] R. Bridson, S. Marino and R. Fedkiw, "*Simulation of Clothing with Folds and Wrinkles*", Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2003, pp. 28–36.

[3] J. Zhang, S. Childress, A. Libchaber, M. Shelley, "*Flexible filaments in a flowing soap film as a model for one-dimensional flags in a two-dimensional wind*", Nature 408, pp. 835–839, 14 December 2000.

[4] D. Pritchard and W. Heidrich, "*Cloth Motion Capture*", Computer Graphics Forum (Proceedings of Eurographics), 22(3):263–271, Sep. 2003.

[5] K. S. Bhat, C. D. Twigg, J. K. Hodgins, P. K. Khosla, Z. Popovič and S. M. Seitz, "*Estimating Cloth Simulation Parameters From Video*", Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2003, pp. 37–51.

[6] J.L. Barron, D.J. Fleet, and S. Beauchemin, "*Performance of optical flow techniques*", International Journal of Computer Vision, 12(1):43–77, 1994.

[7] B. Galvin, B. McCane, K. Novins, D. Mason and S. Mills, "*Recovering Motion Fields: An Evaluation of Eight Optical Flow Algorithms*", Proceedings of the British Machine Vision Conference, Southampton, England, Sep. 1998.

[8] B. D. Lucas and T. Kanade, "*An Iterative Image Registration Technique with an Application to Stereo Vision.*", Proceedings of the Seventh International Joint Conference on Artifical Intelligence, pp. 674–679, Vancouver, Canada, Aug. 1981.

[9] J.Y. Bouguet, "*Pyramidal Implementation of the Lucas Kanade Feature Tracker*", OpenCV Documentation, Microprocessor Research Labs, Intel Corp., 2000.

[10] S. Vedula, S. Baker, P. Rander, R. Collins and T. Kanade, "*Three-Dimensional Scene Flow*", Proceedings of the 7th International Conference on Computer Vision, Vol. 2, pp. 722–729, Corfu, Greece, Sep. 1999.

[11] A. Pentland and B. Horowitz, "*Recovery of Nonrigid Motion and Structure*", IEEE Transactions on Pattern Analysis and Machine Intelligence, 13(7):730–742, 1991.

[12] D. Terzopoulos and D. Metaxas, "*Dynamic 3D Models with Local and Global Deformations: Deformable Superquadrics*", IEEE Transactions on Pattern Analysis and Machine Intelligence, 13(7):703–714, 1991.

[13] S. Malassiotis and M.G. Strintzis, "*Model-Based Joint Motion and Structure Estimation from Stereo Images*", Computer Vision and Image Understanding, 65(1):79–94, 1997.

[14] W. H. Press, B. P. Flannery, S. A. Teukolsky and W. T. Vetterling, "*Numerical Recipes in C*", 2nd edition, Cambridge University Press, 1992.

[15] S. Suzuki and K. Abe, "*Topological Structural Analysis of Digital Images by Border Following*", CVGIP, 30(1):32–46, 1985.

[16] D.H. Eberly, "*Game Physics*", Morgan Kaufmann Publishers, 2003.

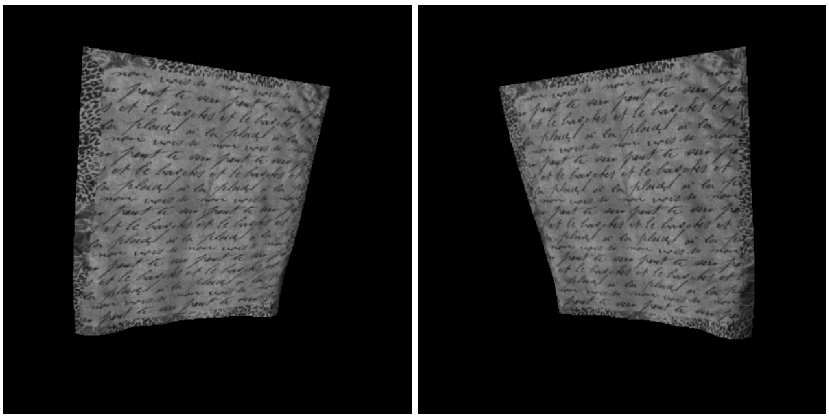

Figure 5: Input frames from two of three camera views.

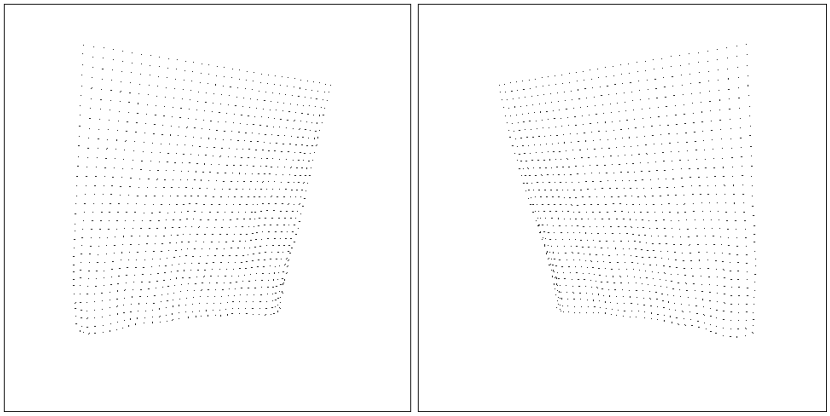

Figure 6: Optical flow vector fields between consecutive input frames.

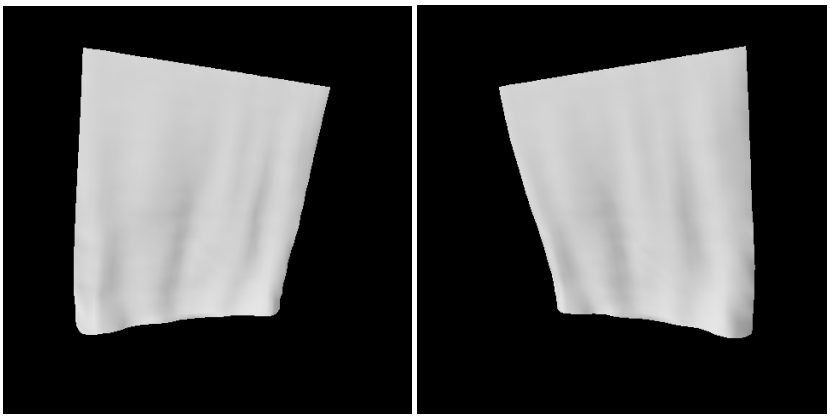

Figure 7: Reconstructed surface. Notice the cloth folds our method is able to find.

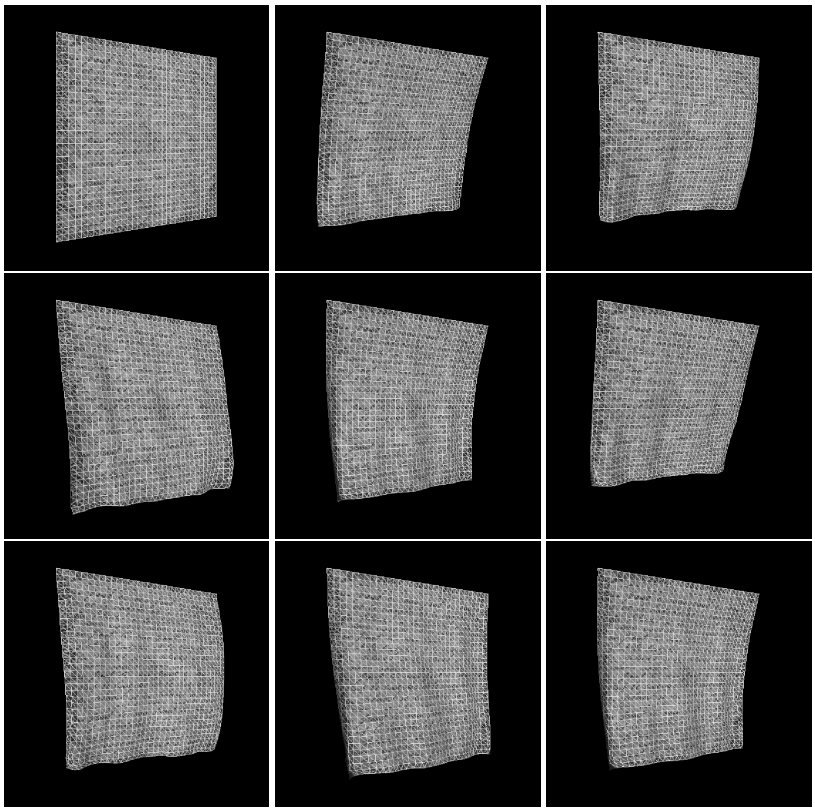

Figure 8: Wireframe models overlayed on the input images show the tracking accuracy of the algorithm.

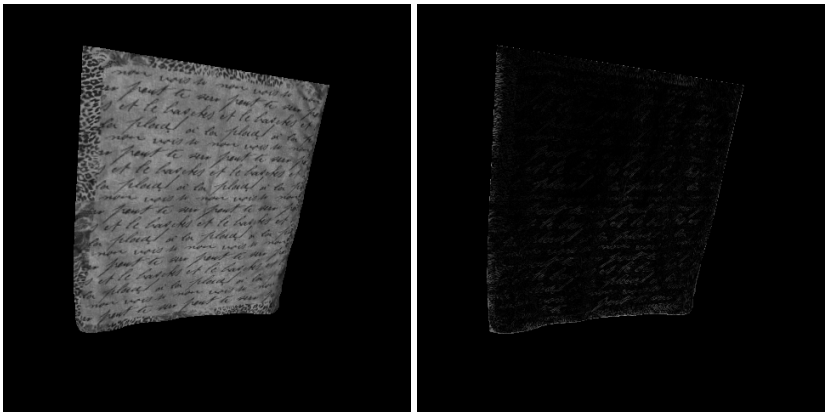

Figure 9: Reconstructed frame and the difference to the left input image in Fig. 5.