Eurographics Symposium on Rendering (2006)
Tomas Akenine-Möller and Wolfgang Heidrich (Editors)

Texture Replacement of Garments in Monocular Video Sequences

Volker Scholz[1] and Marcus Magnor[2]

[1]MPI Informatik [2]TU Braunschweig

Abstract

In this paper, we present a video processing algorithm for texture replacement of moving garments in monocular video recordings. We use a color-coded pattern which encodes texture coordinates within a local neighborhood in order to determine the geometric deformation of the texture. A time-coherent texture interpolation is obtained by the use of 3D radial basis functions. Shading maps are determined with a surface reconstruction technique and applied to new textures which replace the color pattern in the video sequence. Our method enables exchanging fabric pattern designs of garments worn by actors as a video post-processing step.

Categories and Subject Descriptors (according to ACM CCS): I.4.1 [Image Processing and Computer Vision]: Digitization and Image Capture I.3.7 [Computer Graphics]: Animation

1. Introduction

Movie production has always consisted of elaborate post-recording work. Movie cutting has long been the only means to alter movie content after recording. Even so, the cut has great influence on the perception (and success) of a movie and probably constitutes the artistically demanding challenge to any movie director. Today's digital image processing tools have greatly advanced movie editing capabilities. However, considerable, time-consuming manual interaction is still necessary if the content of a recorded scene is to be altered long after all actors have gone home.

Our method addresses a specific problem of movie post-production. We propose to realistically alter garment appearance and limit manual interaction to segmentation of the garment region. Our method could also be useful for virtual fashion presentation in e-commerce. Current rotoscoping software allows tracking edges or single features in videos for tasks like matting of CGI objects, selective filtering and creating cartoon animation from video [AHSS04]. For our purpose, however, an automatic approach is needed which can track several hundred texture features in parallel while handling occlusions automatically. Manual texture editing is in this case almost infeasible. We propose such a system to enable texture replacement with correct texture deformation and lighting.

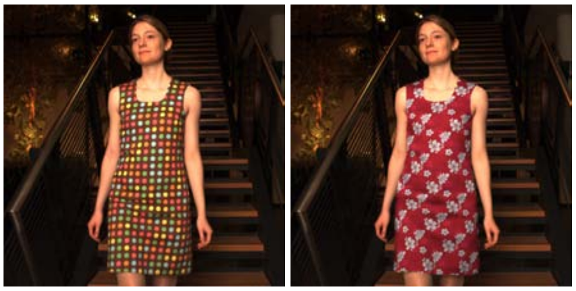
Figure 1: *Input frame (left) and texture replacement result (right). Notice how shading adds an important visual cue.*

Our paper is organized as follows. In Section 2 we describe related work in this area. Section 3 gives an overview of our system. In Section 4-7 we detail our proposed method. Section 8 presents results. We end in Section 9 by drawing conclusions on our work and we mention ideas for future work.

2. Related Work

Several authors have worked on texture replacement in still images. [TLR01] propose to replace near-regular texture patterns in a plane by learning a statistical texture model and lighting distributions from a sample image. [OCDD01] use texture replacement in their image editing system. Depth information is used to generate foreshortening distortions of the texture, and lighting changes are also extracted. Image Analogy [HJO*01] and Image Quilting [EF01] show texture transfer effects which preserve local appearance of the texture but do not model texture distortion and lighting effects explicitly. [LLH04] present an approach which builds on user-assisted lattice extraction for near-regular texture (e.g. a brick wall). A PCA analysis of the obtained geometric and lighting deformation fields allows to control texture regularity. Textureshop [FH04a] introduces the idea of using shape-from-shading to recover a rough set of normals for a non-textured surface in the image and using these normals to introduce distortion in the texture synthesis process. User interaction is required to fix normal recovery errors. [ZFGH05] present a faster system with improved object selection, texture synthesis and shape-from-shading algorithms. [LF04] use a shape-from-texture algorithm to recover the shape and irradiance map for textured cloth to replace the texture.

Our work is also related to cloth motion capture where the goal is to capture 3D motion. Approaches for general textures [PH03, SM04] and for color-coded patterns [GKB03, SSK*05] have been proposed. While periodic and general patterns can cause serious correspondence problems, color-coded patterns are able to avoid this problem. These methods, however, all rely on synchronized multi-video footage of the garment. Multi-camera systems are mainly used in research labs so a monocular method would open up a variety of new applications (films, TV etc.). A monocular cloth capture method is described in [TB02, TH04]. They obtain reconstructions of non-rigid surfaces by tracking sparse feature sets. While the results are impressive for single camera reconstruction, the features are too sparse for a detailed representation of cloth folds.

The major difficulty of replacing texture in video streams consists of achieving temporal coherence. A single-frame method would inevitably lead to flickering artifacts. [PLF05b] have proposed an algorithm for real-time non-rigid surface detection for arbitrary textures which detects a surface by per frame feature matching in conjunction with a deformable mesh model. Being a single frame method, however, temporal coherence is not considered. They extend this work in [PLF05a] by taking shading effects into account. [BR04] augment cloth and paper with texture and interpolated lighting by using augmented reality square markers. Recently [WF06] have retextured special clothing with color patterns and natural clothing with a limited number of colors. Their irradiance estimation exploits the property that pixels can be classified into few color classes. Texture replacement for video data maintaining temporal coherence has been attempted only recently [Lin05, LL06]. The method is based on user-assisted lattice extraction for near-regular texture on cloth. The lattice structure is modeled by a Markov Random Field and tracked with an affine Lucas-Kanade algorithm. Temporal coherence of the texture deformation and shading maps is achieved by spatiotemporal smoothing as a post-processing step.

Determining reflectance and shading at each scene point is also referred to as the intrinsic image problem. The goal is to decompose an input image into two images, one containing the shading information and the other the reflectance information. [OCDD01] make the simplifying assumption that large-scale luminance variations are due to the lighting, while small-scale variations are due to texture. The texture features are blurred with an adaptive bilateral filter. A texture image with uniform lighting is obtained by dividing the initial image by the blurred image. The computer vision literature contains several algorithms to solve the general intrinsic image problem. [TFA05] use machine learning for classifying image pixels while [FDL04] rely on a projection of color onto gray images minimizing image entropy. In [FDB92] the shading field is recovered by removing reflectance changes in the gradient image. Integrating the manipulated gradient field by solving a Poisson equation leads to the shading image.

Our approach is most closely related to the work by [Lin05]. Their method is based on near-regular textures while we recover garment shape information by using a special pattern printed on the fabric. Our method is robust to deformations, lighting changes and feature occlusions and is reinitialized at every frame. Tracking is fully automated and does not require any user interaction as in [Lin05]. Knowing the garment's texture, we have more a-priori information available to avoid ambiguous correspondence matches. This enables to robustly cope also with fast motion. Seam detection and interpolation are fully automated. By encoding texture coordinates in our color code, our method can also deal with folding topologies where parts of the fabric are hidden by self-occlusion. Due to ambiguous correspondences this is difficult for methods based on periodic textures [Lin05].

In the following we

- propose a novel processing pipeline suitable for monocular video footage
- apply 3D spatiotemporal RBF approximation as a general approach to ensure temporal coherence, and we
- determine shading maps with thin-plate interpolation.

As the result, we are able to manipulate and change garment texture as a post-processing step.

3. Overview and Segmentation

Figure 2 shows an overview of our system. For proper texture replacement, we need a segmentation of the images into garment and background sections. For this purpose we use the rotoscoping software by [AHSS04] for contour tracking which requires the user to specify contour curves at key frames. In general, any other video segmentation method

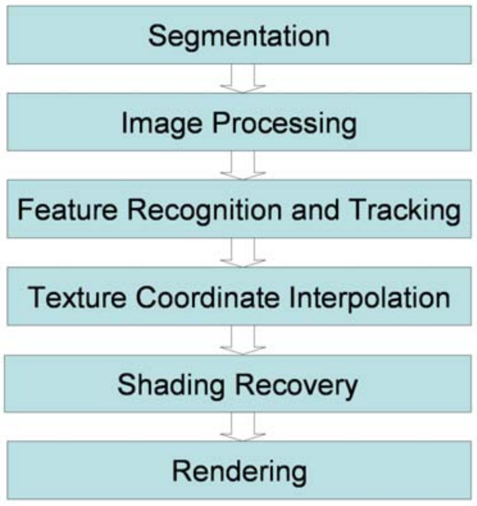
Segmentation
Image Processing
Feature Recognition and Tracking
Texture Coordinate Interpolation
Shading Recovery
Rendering

Figure 2: *Overview of the processing steps of our method.*

[LSS05, WBC*05] which delivers alpha mattes could also be used. These methods all require some amount of user interaction. This preprocessing step yields boundary curves which are converted into a binary mask for the foreground where all further processing is done. Next, we perform image processing for feature classification. We use garments with a custom-designed color-coded pattern. The pattern uses five different colors and is constructed in a way that allows to identify each dot in the pattern by its $3x3$ neighborhood. We use the single-frame method from [SSK*05] which identifies the dots by their local neighborhood with a region-growing approach. Texture coordinates can then automatically be assigned to each dot. The results of the labeling algorithm are complemented by a feature tracker and fed into a texture interpolation algorithm which determines a time-coherent image texture from the feature positions. For realistic texture shading we determine the shading image. The new texture is rendered into each video image by multiplying texture color with the corresponding shading image.

4. Image Processing

We convert the input video images into HSV color space in order to increase color recognition robustness against illumination changes. For color classification, we only use hue and learn the five color classes from an example image. The feature pixels are identified with an adaptive thresholding algorithm [GW02] in the luminance image. From these pixel positions we collect the hue values and fit a Gaussian distribution to each color class (Gaussian mixture model GMM) with a statistical technique [FH04b]. First, we run k-means with random initial centers on the input data and apply the EM algorithm for determining GMMs [HTF01]. As the EM algorithm can stagnate in local minima this procedure is restarted 10 times and the result with the best log-likelihood is kept as the final result. After this step, the pixels segmented by the adaptive thresholding method can be classified into five color classes. For this purpose we compute maximum-likelihood decision boundaries from the Gaussian parameters μ_i, σ_i of each color class. The color-classified image is labeled with a connected component algorithm for every color separately [HS92]. The obtained features are filtered by an upper and lower bound for their area. Finally, for every feature the center of mass is calculated. We now have 2D image coordinates of a number of color dots on the garment.

5. Feature Recognition and Tracking

We use two different pieces of apparel for our experiments, a dress and a skirt. A priori known is the pattern matrix M which contains a color label for each dot in the pattern. We identify the outline of the individual cloth panels (three for the dress and two for the skirt, Fig. 3) in the pattern matrix manually and identify also the boundary dots adjacent to the seams. Panel boundaries are interactively identified only once per garment. The algorithm proposed in [SSK*05] labels the features obtained in the image processing step with their indices i, j in the pattern matrix M. In a first step, a seed dot with its $3x3$ neighborhood and adjacent dots are found by region growing. The direction of search is directed by the local principal directions u,v of the pattern lattice. The obtained indices i, j yield the texture coordinates for the feature dots. Details of the algorithm and the construction of the color code can be found in [SSK*05]. The original feature labeling algorithm is a robust single-frame method which does not use tracking history. Its performance deteriorates at oblique surface angles and requires also that a seed with a $3x3$ neighborhood can be identified for each connected texture component in the image. In order to increase the number of recognized features we apply a feature tracker to fill in missing features after labeling [LK81, Int01]. We track the features known from labeling with image patches and set the patch size to the mean distance of neighboring features. In order to handle feature occlusions between a pair of images, we run the tracker forward in time and track the obtained position backward. As occlusion test, we compute the deviation from the original feature position. If it is below some threshold (1 pixel in our experiments), the feature was tracked successfully and is added to the list of recognized features. Feature tracking is applied to the whole video sequence forward and backward in time in order to take feature occlusions and disocclusions into account.

6. Texture Coordinate Interpolation

Our garments consist of several panels (Fig. 3). Texture interpolation is done separately for every panel so our method is also applicable to several pieces of clothing at the same time. In the following we assume that we have no tears in the

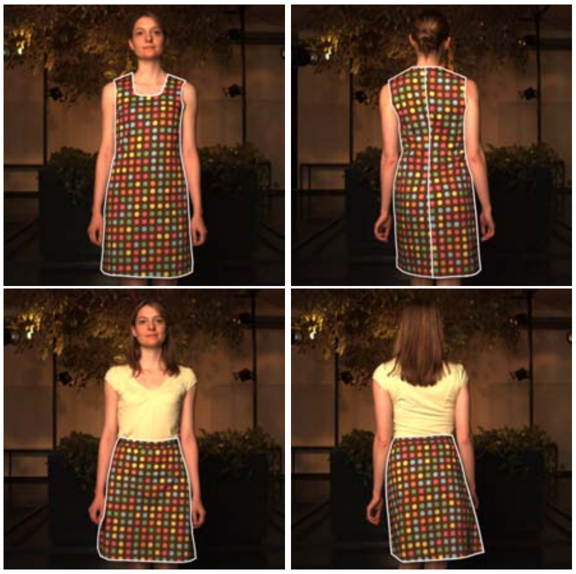
Figure 3: *Our dress has three panels: one front panel and two back panels (top). The skirt has a front and a back panel (bottom).*

fabric, i.e. the panels are continuous. In views where several panels are visible in the image, we have to find the seams between the panels in order to determine the panel segments. We determine the visible seams in the image by identifying dots at the panel boundaries which are a priori known from the pattern matrix. As additional information we know which boundary dots of different panels are adjacent to each other at the seams. The boundary dots lie inside the panel, not on the seam and do not define a smooth boundary due to the discrete nature of the pattern. A smooth boundary polyline is obtained by interpolating a new seam point between each pair of adjacent boundary dots. We use membrane interpolation $\Delta f = 0$ where Δ is the Laplacian operator and fix the position of the boundary dots. In order to obtain a smooth polyline, we use a weighted Laplacian stencil in the corresponding linear system which assigns a higher weight to neighboring seam points. The seam points define an estimate of the panel seam which cannot be determined from the images directly (Fig. 4). The seam polylines are used to cut out a mask for each visible panel from the foreground mask.

Our goal is a temporally smooth parametrization of the garment region with texture coordinates. Near the silhouettes, the feature trajectories are not stable enough (due to failure of detection and occlusions). Smoothing individual trajectories would not be helpful in this case. Therefore we integrate the smoothing into the interpolation function. Radial Basis Functions (RBF) are commonly used for scattered data interpolation problems like reconstructing surfaces from point clouds [CBC*01]. A trivariate scalar RBF

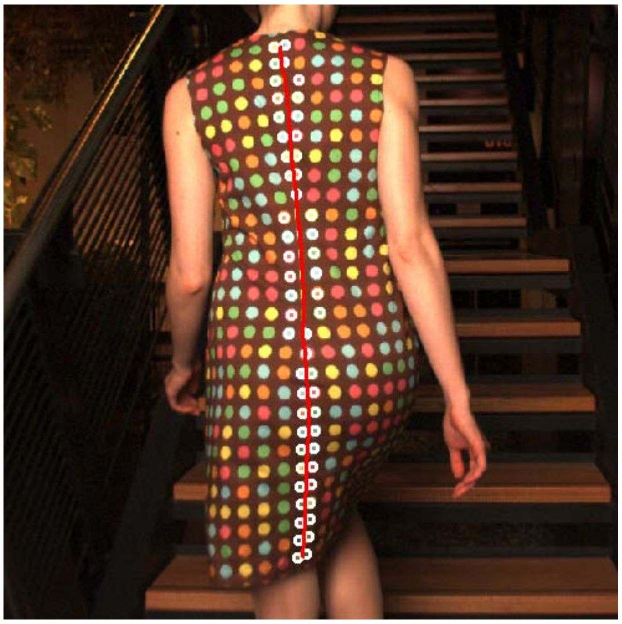
Figure 4: *Boundary dots (white) and interpolated seam boundary (red) necessary for texture replacement of separate garment panels.*

is defined by a set of centers $\mathbf{c}_i \in \mathbb{R}^3$ and weights $w_i \in \mathbb{R}$ as [CBC*01]

$$f(\mathbf{x}) = p(\mathbf{x}) + \sum_i w_i \cdot \phi(\mathbf{x} - \mathbf{c_i}) \qquad (1)$$

where ϕ is the basis function and $p(\mathbf{x})$ is a polynomial of low degree. Since basis functions with local support do not provide the same degree of extrapolation and hole filling capabilities as functions of global support [CBC*01], we use the global basis function $\phi(\mathbf{x}) = \|\mathbf{x}\|$ where $\|\,\boldsymbol{.}\,\|$ is the Euclidean norm, and a linear polynomial p. The resulting surface is a biharmonic thin-plate spline. For interpolating texture coordinates $(u\,v)^T \in \mathbb{R}^2$ we use a vector-valued RBF

$$\mathbf{f}(\mathbf{x}) = \mathbf{p}(\mathbf{x}) + \sum_i \mathbf{w_i} \cdot \phi(\mathbf{x} - \mathbf{c_i}) \qquad (2)$$

with $\mathbf{f} : \mathbb{R}^3 \to \mathbb{R}^2$ and $\mathbf{w_i} \in \mathbb{R}^2$, $\mathbf{p} \in \mathbb{R}^2$ are vectors. $\mathbf{f}$ is defined in spatiotemporal 3D space (x, y, t) for temporally smooth texture interpolation. This means we have to add a time coordinate to the obtained feature positions x, y. The difference $t_{n+1} - t_n$ of adjacent video frames is set to the mean distance of neighboring features in frame n in order to make the method adaptive to feature scale. We now use RBF approximation (also known as spline smoothing [Wah90, CBC*01]) by solving

$$\begin{pmatrix} \Phi - 8N\pi\rho I & P \\ P^T & 0 \end{pmatrix} \begin{pmatrix} \mathbf{w}_i \\ \mathbf{q}_i \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ 0 \end{pmatrix} \qquad (3)$$

where $\Phi_{ij} = \phi(\mathbf{c_i} - \mathbf{c_j})$, $P_{ij} = p_j(\mathbf{c_i})$ for the polynomial basis $\{p_1, p_2, p_3\} = \{1, x, y\}$. The $\mathbf{q}_i$ are polynomial coefficients,

N is the number of centers and I is the identity matrix. ρ is a parameter that determines the trade-off between smoothness of the surface and fidelity to the data. This parameter is found empirically. We use $\rho = 0.005$ for all examples (the smallest amount of smoothing which leads to reasonable results). The resulting matrix is dense due to the global nature of ϕ and can be solved directly for our problem size of $N \leq 1000$ centers. RBF approximation is used for overlapping time windows of three video frames to ensure temporal smoothness. The texture coordinates are interpolated for every pixel in the foreground mask of the middle frame. Larger time windows do not improve the results significantly.

7. Shading Maps

The goal of the shading algorithm is to remove the reflectance contribution of the color dots from the luminance images I while preserving shading effects. We interpolate the dot regions with smooth thin-plate splines in order to get a homogeneous shading map. The dot pixels identified by adaptive thresholding from Section 4 are used as input for shading map computation. We assume here that the dot edges have a higher contrast than shadow edges on the garment which is the case in practice. The detected dots are dilated with a circle-shaped morphological structure element [GW02] in order to remove the dots reliably (Fig. 5b). The inverse image yields a mask which is multiplied with the image I. We interpolate the deleted dot regions (Fig. 5c) by using a surface reconstruction method for height fields [Ter88]. An approximating thin-plate surface is fitted to the luminance values of the dark garment background.

This is done my minimizing the energy functional

$$E = \iint_{\Omega} \frac{\alpha(x,y)}{2}(I-J)^2 + (J_{xx}^2 + 2J_{xy}^2 + J_{yy}^2)\,dx\,dy \qquad (4)$$

where J is the thin-plate surface interpolant, Ω is a bounding box of the segmented garment region and $\alpha(x,y)$ a weight term for the interpolation constraint. We set $\alpha = 0$ in dot regions (interpolation) and $\alpha = 0.1$ for the remaining pixels (approximation) which results in homogeneous shading maps. The corresponding Euler-Lagrange equation is

$$\alpha(I-J) + \Delta^2 J = 0 \qquad (5)$$

where Δ^2 denotes the Bilaplacian operator. Eq. 5 is discretized with finite elements on the pixel grid and uses a 5x5 stencil for the Bilaplacian. A bounding box of the foreground mask is computed and Eq. 5 is solved on this rectangle. At the bounding box borders not all 24 neighbors might exist so we recompute the stencil for the existing neighbors [Ter88]. This leads to a sparse linear system $Ax = b$ where the number of variables equals the number of reconstructed pixels. We solve it in MATLAB with a Cholesky-based solver for systems up to $n = 400.000$ variables. The single-frame shading maps show temporal fluctuations. Therefore we filter the shading maps with a temporal Gaussian filter per pixel (window size 3-5 frames). In order to get an accurate result for fast image motion, we build pixel correspondences between different frames by using the feature correspondences obtained during feature tracking. The features deliver a sparse set of flow vectors (we compute forward flow to the next frame and backward flow to the previous frame). This set is interpolated per pixel by fitting a 2D thin-plate smoothing spline [Wah90] with $\alpha = 10$. The obtained flow fields are used as spatial offsets during temporal filtering.

The obtained shading maps are applied to the new texture during rendering by multiplication per-texel. For the texture lookup we use bilinear interpolation. We adapt the maps to a higher albedo by rescaling. The maps are rescaled by dividing with a reference white value which is obtained by recording a reference image with maximum brightness of the fabric. Note that this is correct for fabric with Lambertian reflectance only. Fortunately, our fabric is close to Lambertian. Our tracking method does not handle non-Lambertian fabrics very well as specularities would compromise the color recognition algorithm. As final step we apply a gamma correction to the rendered images.

8. Results

We record our sequences with an Imperx 1004C vision camera (1004x1004 pixels) at 25 frames per second with raw output in order to avoid compression artifacts. We put the camera on a tripod. However, our method is not limited to static camera position and works just as well for hand-held camera sequences. The camera is color-calibrated with a reference color checker and a linear regression model in order to obtain a good separation of the garments' dot colors for feature classification. For the garments we use a cotton fabric with a custom-printed color pattern using a medium gray tone as background. A high-brightness contrast between the color dots and background is needed for robust feature recognition (adaptive thresholding), whereas the shading algorithm needs a reasonably bright fabric, so we meet both requirements with a medium gray tone. The dot spacing is 3.2 cm and the diameter is 2.1 cm which is a compromise between high sampling rate of the surface and sufficient dot size in the image when capturing a whole person. Digital printing makes it easy to design such a pattern and send it to a company specialized on fabric printing. The garments are manufactured by a professional tailor for the recorded subject.

All experiments are performed on a Pentium IV 3.2 GHz with 2 GB RAM. The average computation time for the automatic processing steps (Fig. 2) in our unoptimized MATLAB implementation for a 1004x1004 video frame is 60 seconds (45 seconds for the shading map). Selected algorithms are implemented in C++: labeling, RBF evaluation, bilinear texture lookup and optical flow. Fig. 6 shows the accuracy of our texture interpolation algorithm. The corners of the overlayed checkerboard texture lie on the geometric centers of the color dots, although we need to smooth the texture

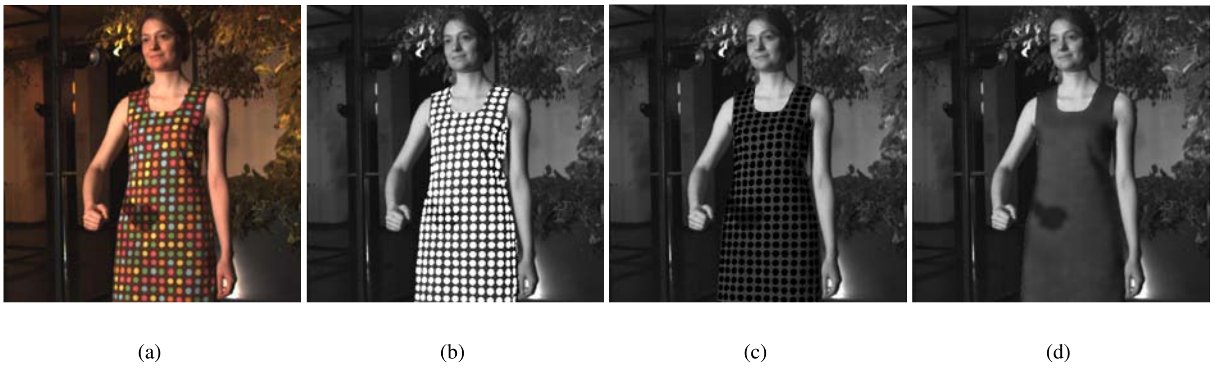

Figure 5: *Input image (a), detected dots (b), removed dots (c), and shading map (d). Although the input contains strong shadow edges, the interpolation results are satisfactorily close to the input frame (a). The video reveals a faint shadow from a secondary light source.*

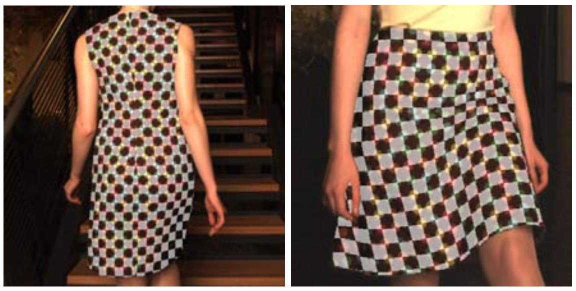

Figure 6: *Accuracy of texture distortion visualized as overlayed checkerboard texture.*

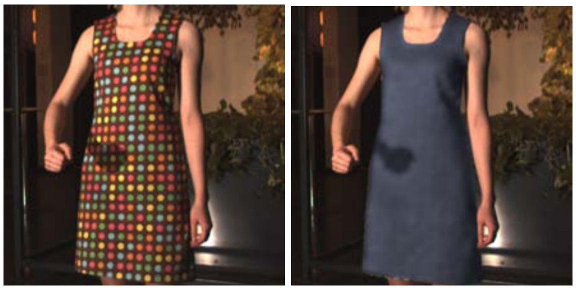

Figure 7: *Shadows are preserved in our renderings. They appear softer because we regularize the solution. The shadow contrast is higher than in Fig. 5d because the shading map is rescaled during rendering.*

maps. At the garment borders, the parameterization is less accurate because fewer features are detected (Fig. 6, right example). The temporal smoothness of the interpolation can be assessed in the accompanying video. We obtain realistic shading maps which preserve shadows and the shading of cloth folds (Fig. 5, 7, 8, 9, 10, 11). While our pattern cannot capture fine folds, the shading maps contain this information. The catwalk sequence shows also the robustness of feature recognition against lighting changes (the garment is rather dark in the beginning of the sequence). Two sequences show fast jumping motion to validate the feature tracking ability.

One limitation of our method is that video segmentation still requires user interaction. This is a notoriously difficult problem (e.g. due to shadows adjacent to the garment borders) where most automatic approaches require manual correction for an accurate result. Segmentation is not the main focus of our work. Our RBF model handles discontinuities at self-occlusions only in an approximate way (the discontinuities are smoothed). For very loose garments the results might not be visually satisfactory in this case. In our experiments self-occlusion due to folding is barely observable because the dots are quite far apart. Self-occlusions between different garment panels however (e.g. between two legs for trousers) are not a problem as the RBF fitting is done separately for every panel. In our experience the texture compression effect at discontinuities is not very noticeable. Our texture maps require spatiotemporal smoothing at the garment borders because feature detection is affected by foreshortening, especially when the overall feature size is small (full person capture). This is an inevitable drawback of a monocular method. Still, our proposed method is able to replace the fabric texture with realistic deformation and lighting for a wide range of real-world scenes. It can robustly deal with deformation, fast motion, lighting changes and feature occlusion.

9. Conclusion and Future Work

We have presented a system for automatic texture replacement of color-coded garments. Visually convincing replace-

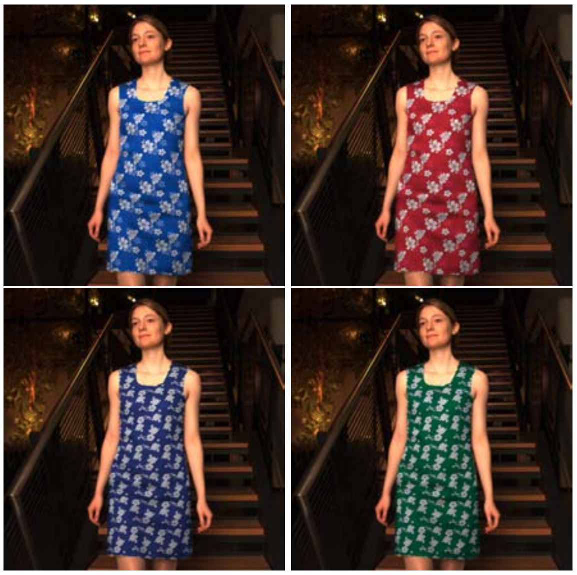

Figure 8: *Replacement results with different patterns.*

ment results are obtained by using 3D spatiotemporal RBF approximation. We also show that our shading maps can capture small details at cloth folds. A single-view method is more challenging than a multi-view approach but opens up new applications. Currently, the performance bottleneck in our implementation is the computation of the shading maps. Using faster solvers (e.g. multigrid) would improve our system's time and memory consumption. Changing the reflectance properties of the fabric and relighting are also interesting topics for future research.

Acknowledgments

We would like to thank our fashion model Julia Luxenburger for making this project possible, especially for her patience during the tailoring and recording process. Thanks to Aseem Agarwala for making the rotoscoping code available. This work is supported by the EC within FP6 under Grant 511568 with the acronym 3DTV.

References

[AHSS04] AGARWALA A., HERTZMANN A., SALESIN D., SEITZ S. M.: Keyframe-based tracking for rotoscoping and animation. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH 2004) 23*, 3 (2004), 584–591.

[BR04] BRADLEY D., ROTH G.: *Augmenting Non-Rigid Objects with Realistic Lighting*. Tech. Rep. NRC 47398, National Research Council Canada, Institute for Information Technology, 2004.

[CBC*01] CARR J. C., BEATSON R. K., CHERRIE J. B., MITCHELL T. J., FRIGHT W. R., MCCALLUM B. C., EVANS T. R.: Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of ACM SIGGRAPH 2001* (2001), pp. 433–442.

[EF01] EFROS A. A., FREEMAN W. T.: Image quilting for texture synthesis and transfer. In *Proceedings of ACM SIGGRAPH 2001* (2001), pp. 341–346.

[FDB92] FUNT B. V., DREW M. S., BROCKINGTON M.: Recovering shading from color images. In *Proc. ECCV* (1992), pp. 124–132.

[FDL04] FINLAYSON G. D., DREW M. S., LU C.: Intrinsic images by entropy minimization. In *Proc. ECCV* (2004), pp. 582–595.

[FH04a] FANG H., HART J. C.: Textureshop: texture synthesis as a photograph editing tool. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH 2004) 23*, 3 (2004), 354–359.

[FH04b] FRANC V., HLAVÁČ V.: *Statistical Pattern Recognition Toolbox for MATLAB*. Tech. Rep. CTU–CMP–2004–08, Center for Machine Perception, Czech Technical University, 2004.

[GKB03] GUSKOV I., KLIBANOV S., BRYANT B.: Trackable surfaces. In *Proc. of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)* (2003), pp. 251–257.

[GW02] GONZALEZ R. C., WOODS R. E.: *Digital Image Processing*. Prentice Hall, 2002.

[HJO*01] HERTZMANN A., JACOBS C. E., OLIVER N., CURLESS B., SALESIN D.: Image analogies. In *Proceedings of ACM SIGGRAPH 2001* (2001), pp. 327–340.

[HS92] HARALICK R. M., SHAPIRO L. G.: *Computer and Robot Vision Volume I*. Addison Wesley, 1992.

[HTF01] HASTIE T., TIBSHIRANI R., FRIEDMAN J.: *The Elements of Statistical Learning*. Springer, 2001.

[Int01] INTEL CORPORATION: *Open Source Computer Vision Library*, 2001.

[LF04] LOBAY A., FORSYTH D.: Recovering shape and irradiance maps from rich dense texton fields. In *Proc. CVPR (1)* (2004), pp. 400–406.

[Lin05] LIN W.-C.: *A Lattice-based MRF Model for Dynamic Near-regular Texture Tracking and Manipulation*. PhD thesis, Carnegie Mellon University, 2005.

[LK81] LUCAS B., KANADE T.: An iterative image registration technique with an application to stereo vision. In *Proc. Seventh International Joint Conference on Artificial Intelligence* (1981), pp. 674–679.

[LL06] LIN W.-C., LIU Y.: Tracking dynamic near-regular textures under occlusion and rapid movements. In *Proc. ECCV* (May 2006).

[LLH04] LIU Y., LIN W.-C., HAYS J.: Near-regular texture analysis and manipulation. *ACM Transactions on*

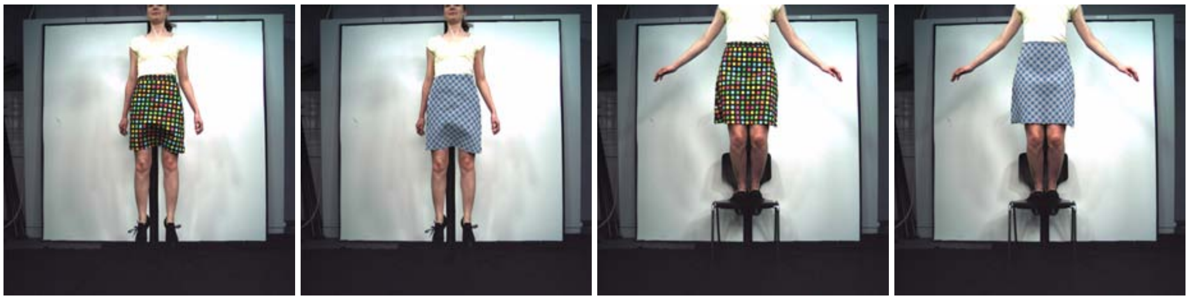

Figure 9: *Our method can track fast motion because it can re-initialize at every video frame (see video).*

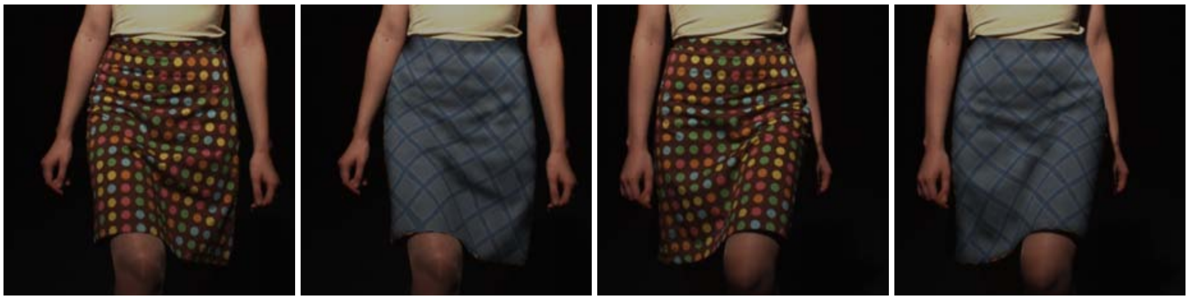

Figure 10: *Shading effects near wrinkles and cloth folds. The shading maps faithfully represent the main cloth folds.*

Graphics (Proc. of ACM SIGGRAPH 2004) 23, 3 (2004), 368–376.

[LSS05] LI Y., SUN J., SHUM H.-Y.: Video object cut and paste. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH 2005) 24*, 3 (2005), 595–600.

[OCDD01] OH B. M., CHEN M., DORSEY J., DURAND F.: Image-based modeling and photo editing. In *Proceedings of ACM SIGGRAPH 2001* (2001), pp. 433–442.

[PH03] PRITCHARD D., HEIDRICH W.: Cloth motion capture. *Computer Graphics Forum (Proc. of Eurographics EG '03) 22*, 3 (2003), 263–272.

[PLF05a] PILET J., LEPETIT V., FUA P.: Augmenting deformable objects in real-time. In *International Symposium on Mixed and Augmented Reality* (2005).

[PLF05b] PILET J., LEPETIT V., FUA P.: Real-time non-rigid surface detection. In *Proc. CVPR* (2005), pp. 822–828.

[SM04] SCHOLZ V., MAGNOR M.: Cloth motion from optical flow. In *Proc. Vision, Modeling and Visualization (VMV)* (2004), pp. 117–124.

[SSK*05] SCHOLZ V., STICH T., KECKEISEN M., WACKER M., MAGNOR M.: Garment motion capture using color-coded patterns. *Computer Graphics Forum (Proc. Eurographics EG '05) 24*, 3 (2005), 439–448.

[TB02] TORRESANI L., BREGLER C.: Space-time tracking. In *Proc. ECCV (1)* (2002), pp. 801–812.

[Ter88] TERZOPOULOS D.: The computation of visible-surface representations. *IEEE Trans. Pattern Anal. Mach. Intell. 10*, 4 (1988), 417–438.

[TFA05] TAPPEN M. F., FREEMAN W. T., ADELSON E. H.: Recovering intrinsic images from a single image. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 9 (2005), 1459–1472.

[TH04] TORRESANI L., HERTZMANN A.: Automatic non-rigid 3d modeling from video. In *Proc. ECCV (2)* (2004), pp. 299–312.

[TLR01] TSIN Y., LIU Y., RAMESH V.: Texture replacement in real images. In *Proc. CVPR (2)* (2001), pp. 539–544.

[Wah90] WAHBA G.: *Spline Models for Observational Data*. SIAM, 1990.

[WBC*05] WANG J., BHAT P., COLBURN A., AGRAWALA M., COHEN M. F.: Interactive video cutout. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH 2005) 24*, 3 (2005), 585–594.

[WF06] WHITE R., FORSYTH D.: Retexturing single views using texture and shading. In *European Conference on Computer Vision* (2006), vol. LNCS 3954, Springer, pp. 70–81.

[ZFGH05] ZELINKA S., FANG H., GARLAND M., HART J. C.: Interactive material replacement in photographs. In *Proc. Graphics Interface* (2005), pp. 227–232.