Enhancing Silhouette-based Human Motion Capture with 3D Motion Fields

Christian Theobalt Joel Carranza Marcus A. Magnor
Hans-Peter Seidel
Max-Planck-Institut für Informatik
Saarbrücken, Germany
{theobalt|carranza|magnor|hpseidel}@mpi-sb.mpg.de

Abstract

High-quality non-intrusive human motion capture is necessary for acquistion of model-based free-viewpoint video of human actors. Silhouette-based approaches have demonstrated that they are able to accurately recover a large range of human motion from multi-view video. However, they fail to make use of all available information, specifically that of texture information. This paper presents an algorithm that uses motion fields constructed from optical flow in multi-view video sequences.

The use of motion fields augments the silhoutte-based method by incorporating texture-information into the tracking process. The algorithm is a key-component in a larger free-viewpoint video system of human actors. Our results demonstrate that our method accurately estimates pose parameters and allows for realistic texture generation in 3D video sequences.

1. Introduction

The synthesis of realistic images of humans in motion is a challenging problem in Computer Graphics. Two aspects of this problem are the creation of natural human motion and the accurate rendering of a person's physical appearance. Combining the small scale details of skin, muscle, and cloth movement with the large scale motions of the body into a realistic image has required the development of new techniques which rely on the strengths of both Computer Vision and Computer Graphics. Many conventional methods for estimating motion parameters are intrusive, requiring optical markers or complex mechanical setups, and thus require a separation of the generation of realistic motion from the generation of realistic physical appearance. However, in the field of Computer Vision, numerous techniques have been developed for non-intrustive motion parameter estimation. Incorporating some of these techniques into Computer Graphics allows us to capture body appearance and motion at the same time, vastly simplifying the problem of novel image synthesis.

We have developed a method which non-intrusively estimates motion parameters using silhouette information. This method employs the use of a detailed geometric body model which, when combined with image-based rendering techniques, generates highly realistic images of a body in motion. Silhouette-based techniques have demonstated their power in capturing a broad range of complex body motion. However, portions of the body with small scale details (such as features of the face) are often not accurately represented. To be maximally effective, an algorithm should make use of all possible information. We propose to use texture information to augment the silhouette-fitting process.

Optical flow is a computer vision technique that employs texture information to compute a 2D motion field in the image plane. Making use of optical flow calculations from multiple input views and an a-priori body model, it becomes possible to construct a 3D motion field which estimates body motion. We present a method for extracting heirarchial rigid body transformations from these motion fields and show that it is best used in conjunction with, and not in place of, silhouette-based tracking. At each time instant, the generic body model is first fit to a set of input-camera silhouettes. That pose is then updated to conform with estimates from the computed motion field. The use of motion fields in conjunction with silhouette-based fitting methods generates good results, precisely because each method excels in aspects where the other does not. Non-intrusive motion parameter estimation is a component in our larger free-viewpoint video system. We demonstrate that this new hybrid method improves motion parameter estimation and consequently has a significant impact on the quality of generated free-viewpoint video sequences [4].

The paper proceeds with a discussion of previous work in Section 2, and a general overview of the proposed motion capture method in the context of our free-viewpoint video system is given in Section 3. Our environment for acquiring multi-view video streams is described in Section 4. The employed body model and multi-view texture generation are presented in Section 5 and Section 7 respectively, and the silhouette fitting step of the motion capture system is outlined in Section 6. Preliminaries about optical flow and the reconstruction of 3D motion fields from 2D flows are presented in Section 8. The algorithmic details on how to compute differential pose update parameters from 3D flow fields are explained in Section 9. Results with our algorithm are presented in Section 10 and the the paper concludes in Section 11.

2. Previous Work

In the Computer Vision literature, a variety of non-intrusive optical human motion capture techniques from video have been proposed (see [8] for a review). Some methods work on a single 2D image and apply, for example, frame differencing [14] or image skeletonization [10] to fit simple body models to human motion. 3D human motion capture approaches typically employ an explicit human body model consisting of a joint structure and some form of surface representation. Simple shape primitives, such as cylinders [11, 22] or superquadrics [9], are commonly used to represent limbs. The body models are fitted to the motion by aligning their projection with features in the image plane, such as image discontinuities. The application of silhouette images for human motion capture has also been considered. In [6] a force field exerted by multiple image silhouettes aligns a 3D body model. In [20] a combination of stereo and silhouette fitting is used to fit a human body model, and in [4] a silhouette-based motion estimation method is described that exploits graphics hardware to maximize model and silhouette overlap. Recently, the application of reconstructed volumetric models (visual hulls) from silhouettes of a moving person for motion capture has also been considered. Ellipsoidal body models [5] , kinematic skeletons [17], or skeleton models with attached volume samples [24] are fitted to the volume data.

In 3D video, dynamic models of scenes that were recorded from several camera perspectives are reconstructed for re-rendering from novel viewpoints. The methods applied involve shape-from silhouette-like approaches, such as visual hull [18, 29] or stereo-based approaches [19]. The application of a generic human body model and a non-intrusive human motion capture method for free-viewpoint video was also considered [4].

None of the previously mentioned approaches explicitly uses optical flow or computed 3D velocity fields for motion parameter estimation or scene reconstruction. The optical flow is the observed 2D motion field in the image plane of a camera resulting from the projection of the 3D velocity field of the recorded moving scene (see [1] for a comparison of optical flow techniques). The application of 2D optical flow has been investigated in model-based video coding for deriving facial animation parameters of a generic head model [7] or for recovering motion parameters of a body model in a teleconferencing scenario [15]. Using optical flow in one or more camera views for full body human motion estimation is presented in [3]. In their work, the authors use a twist parameterization for rigid body transformations to solve for the body pose parameters from 2D information directly by solving a linear system. The algorithm computes pose updates and performs image warping in an iterative procedure. None of these methods explicitly reconstruct a 3D motion field. In [27], an algorithm for computing such a 3D motion field from optical flows in multiple camera views is presented. It has been used to improve voxel-based scene reconstruction [28] and to compute models of intermediate time steps in a sequence of shape-from-silhouette representations [26]. Unfortunately, the methods employing optical flow exhibit robustness problems if the typical optical flow assumptions, such as brightness constancy over time and flow similarity in a spatial neighborhood, are not fulfilled. This can happen very easily if the motion in the scene is quite large and effects such as self-shadowing come into play. In contrast, we propose a method that uses a 3D motion field reconstructed from optical flow to update pose parameters computed via a silhouette-based model fitting procedure. The algorithm is an extension of our work presented in [4]. We combine the strengths of silhouette-based fitting for robust acquisition of a large range of motions with that of motion estimation using texture information. Using texture information, pose updates can be recovered on a small scale. For these small corrections, a linear approximation of the motion of the body parts is valid, hence iteration towards a solution is not necessary.

3. System Overview

In Figure 1, an overview of the proposed motion capture algorithm within our free-viewpoint video system is shown. The system takes synchronized multi-view video streams as inputs and separates the person from the background. In an initialization step, the employed body model is adapted to the physical shape of the recorded person. For every time step, the system computes multi-view textures for the body model using image-based techniques. Starting with a body pose recovered for time step t, the system first computes an estimate of the pose parameters for time $t+1$ by optimizing the overlap between the projected model and the silhouette images at time $t+1$. In a second step, this pose estimate for time $t+1$ is augmented by computing a 3D corrective motion field from optical flows. The optical flows are computed between images generated by a model textured with the texture at time t and the input views at time $t+1$.

The reconstructed motion field gives an estimate of differential pose updates that are necessary to correct slight pose inaccuracies in the result of the silhouette step. Least-squares pose updates are computed and added to the pose parameter estimate for time $t + 1$. With the new pose parameters the algorithm iterates to the next time step. In principle, the described motion capture algorithm is a two-step predictor corrector scheme.

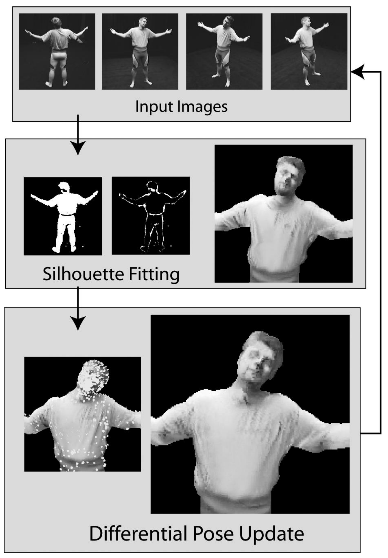

Figure 1. Overview of the motion capture algorithm within our free-viewpoint Video System.

4. Acquisition

The video sequences used as input to our system are recorded in a multi-view camera studio [23]. IEEE1394 cameras are placed in a convergent setup around the center of the scene. The video sequences used for this paper are recorded from 8 static viewing positions arranged at approximately equal angles and distances around the center of the room. The cameras are synchronized via an external trigger. Video frames are recorded at a resolution of 320x240 at 15 fps. The frame rate is fundamentally limited to 15 fps by the external trigger. Using Tsai's algorithm [25] the cameras' intrinsic and extrinsic parameters are determined, calibrating every camera into a common global coordinate system. The lighting conditions are controlled and the all cameras are color-calibrated.

In each video frame, the person in the foreground is segmented via background subtraction. The algorithm used employs per-pixel color-statistics to generate silhouettes (see [5] for details). Shadow regions that might lead to an incorrect classification of background pixels as foreground are eliminated via an additional angular threshold on pixel hue values.

5. Body Model

The body model used throughout the system is a generic model consisting of a hierarchic arrangement of 16 body segments (head, upper arm, torso etc.), each of which is represented by a closed triangle mesh. The model's kinematics are defined via an underlying skeleton consisting of 17 joints connecting bone segments. Rigid transformations at each of these joint locations define a specific body pose for the model. These transformations are constrainted to imitate the actual motions of the body. Shoulder and hip joints are represented by 3 degree-of-freedom (DOF) ball joints and elbow and knee joints are represented by 1-DOF hinge joints. Assuming the actor stands in a specific initialization pose, the generic body model shape is conformed to that of the person through a silhouette-based fitting process [4]. From this point on, bone lengths and segment geometry is fixed and motion parameter estimation is performed using a combination of silhouette and motion field information.

6. Silhouette Fitting

In a hierarchical non-linear optimization, a set of pose parameters for the body model is computed that maximizes the overlap between the model projected to each image plane and the input silhouettes at time $t + 1$. The optimization uses the pose parameters from time step t as an input. The error function that drives motion capture can be efficiently computed in graphics hardware. The method is described in detail in [4]. It is relevant to this paper only in that it is used as robust prediction scheme for pose parameters.

7. Texture Generation

With any body pose, it becomes possible to generate a textured body model by projecting the input camera views onto the body model surface. The degree to which a specific input view is visible at a given surface location is variable. Per-vertex blending weights are computed based on visibilty and the angular difference between the vertex normal and the input view vector. [4] addresses texture generation for the purpose of novel viewpoint generation. We make use of texture generation for both rendering and motion parameter estimation. This requires a slight modification. When rendering the textured model for motion field computation, the texture coordinates generated from the previous time step are used.

8. Motion Field Preliminaries

This sections briefly reviews the mathematical preliminaries of optical flow and the reconstruction of 3D motion fields.

8.1. 2D Optical Flow

The optical flow is the projection of the 3D velocity field of a moving scene into the 2D image plane of a recording camera. The determination of the 2D optical flow from spatio-temporal intensity variations in images has been an issue in Computer Vision for many years [1].

A number of simplifying assumptions are typically made to compute the optical flow from the pixel intensities of two subsequent images. First, it is assumed that the change in image intensity is due to translation in the image plane only (intensity constancy constraint)

$$I(x,t) = I(x - \mathbf{u}t, 0) \tag{1}$$

where $\mathbf{u} = (u,v)^T$ is the optical flow at image point x, I being the image intensity at x at time t. From a Taylor expansion of eq. 1 the *optical flow constraint equation* is derived

$$\nabla I(x,t) \cdot \mathbf{u} + I_t(x,t) = 0 \tag{2}$$

where $I_t(x,t)$ is the temporal derivative. This is an equation in two unknowns which can not be solved at a single image plane location without additional assumptions. To make the problem tractable, an assumption is typically made about the smoothness of optical flow in a local spatial neighborhood.

In the optical flow technique by Lukas and Kanade [16], a weighted least-squares fit of the local first-order constraints (eq. 2) is computed by minimizing the fuctional

$$\sum_{x \in W} W^2(x)[I(x,t) \cdot \mathbf{u} + I_t(x,t)]^2 \tag{3}$$

where $W(x)$ is a Gaussian neighborhood around the current position x in the image plane. In our system, we use this technique to compute the optical flows.

8.2. 3D Motion Fields

The optical flow observed in a camera is only a 2D-projection of the real world 3D motion field. The goal of motion capture is the recovery of the parameters of three-dimensional motion. A reconstructed 3D motion field from optical flows in multiple camera views can be used to compute these parameters. The reconstruction of the 3D motion field, also know as the *scene flow*, from the 2D optical flows is possible using a technique described in [27].

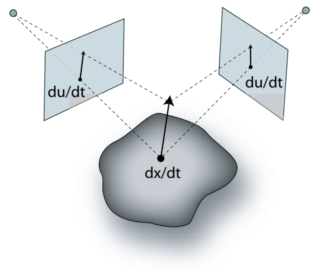

Figure 2. 3D motion (scene flow) of a surface point and the corresponding observed optical flows in two camera views.

If correspondences in the image plane are known, i.e. it is known to which locations 3D points project in each camera view, the scene flow can be reconstructed by solving a linear system. In our system, the correspondences are known for each vertex because we have an explicit body model and since, during calibration, the projection matrices $\mathbf{P}_i$ for each recording camera i were determined. The projection matrices describe the relationship between a 3D position of a vertex and its projection into the image plane of the camera, $\mathbf{u}_i$.

The differential relationship between the vertex $\mathbf{x}$ with coordinates (x,y,z) and $\mathbf{u}_i$ is described by the 2×3 Jacobian matrix$J_i = \frac{\partial \mathbf{u}_i}{\partial \mathbf{x}_i}$.

$$\frac{d\mathbf{u}_i}{dt} = J_i \frac{d\mathbf{x}}{dt} \tag{4}$$

In other words, the Jacobian describes the relationship between a small change in 3D position of a vertex, and the change of its projected image in camera i. The term $\frac{d\mathbf{u}_i}{dt}$ is the optical flow observed in camera i, $\frac{d\mathbf{x}}{dt}$ is the corresponding scene flow of the vertex (Figure 2). Having a mathematical camera model, the Jacobian can be computed analytically (see [27]).

If a vertex is visible from at least 3 camera views, an equation system, $\mathbf{B}\frac{d\mathbf{x}}{dt} = \mathbf{U}$, can be formulated to solve for the scene flow of this vertex given the optical flows in all camera views, where

$$\mathbf{B} = \begin{bmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} & \frac{\partial u_1}{\partial z} \\ \frac{\partial v_1}{\partial x} & \frac{\partial v_1}{\partial y} & \frac{\partial v_1}{\partial z} \\ . & . & . \\ . & . & . \\ \frac{\partial u_N}{\partial x} & \frac{\partial u_N}{\partial y} & \frac{\partial u_N}{\partial z} \\ \frac{\partial v_N}{\partial x} & \frac{\partial v_N}{\partial y} & \frac{\partial v_N}{\partial z} \end{bmatrix}, \mathbf{U} = \begin{bmatrix} \frac{\partial u_1}{\partial t} \\ \frac{\partial v_1}{\partial t} \\ . \\ . \\ \frac{\partial u_N}{\partial t} \\ \frac{\partial v_N}{\partial t} \end{bmatrix} \tag{5}$$

and N is the number of camera views. A least-squares solution to this equation system can be found via singular value decomposition (SVD) [21].

9. Differential Pose Update

Optical flow operates under the assumption that the projection of the underlying motion is purely translational. This is simply not a reasonable approximation for fast or complex motion. We submit that a purely motion-field based tracking system would be suitable for a slow moving subject. However, by combining optical flow and silhouette information, it becomes possible to bypass some of the limitations of optical flow and capture complex, fast motions of the body. Whereas a motion field describes the motion of a scene between two time instants, our *corrective motion field* describes the motion between an intermediary textured model generated from silhouette based tracking and a time instant. These motions are small translations and rotations properly aligning texture information and consequentially suitable for approximation by a linear model.

We apply the previously described scene flow reconstruction algorithm to compute a corrective motion field at each time step. Let $I_{j,t}$ the j-th input camera view at time t, and P_t be the model pose at time t. The algorithm then proceeds as follows:

- With P_t as a starting point, use silhouette fitting to compute P'_{t+1}, an estimated pose for time $t+1$
- Generate $I'_{j,t+1}$ by rendering model from camera j in pose P'_{t+1} with textures from time t.
- **Computation of corrective motion field** D: For each model vertex
  - Determine the projection of the vertex into each camera image plane.
  - Determine vertex visibility in all cameras by comparing the projected z-coordinate to the OpenGL z-buffer value.
  - If a vertex is visible from camera j, compute optical flow from images $I'_{j,t+1}$ and $I_{j,t+1}$.
  - If a vertex is visible in at least three camera views, compute a least squares solution by applying a SVD as described in Section 8.2.
- Update P'_{t+1} to conform with motion field to yield P_{t+1}

The computed corrective 3D motion field D describes vertex position updates that correct slight inaccuracies in the result of the silhouette step. Figure 6 shows examples of corrective scene flow fields. The remainder of the section describes the derivation of the differential pose updates from P'_{t+1} to P_{t+1} using D.

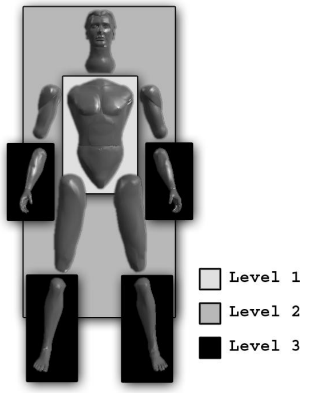

Figure 3. Body model with separated hierarchy levels.

9.1 Differential Pose Update

The corrective motion field D can be used to compute differential pose parameter updates for each limb of the body model. For the root which is located in the torso segment, 3 differential rotation and 3 differential translation parameters are computed. All the joints apart from the root are purely rotational. This includes 3-DOF rotations for the shoulders, hips, and neck, and a 1-DOF rotation for the elbows and knees. The wrist and ankle joints are currently not considered.

By adding each vector in D to the current 3D position of its corresponding vertex, a set of goal positions is defined for each model vertex. The goal is to find the set of differential joint parameters of the body model that best aligns the vertices with these positions. The idea is to compute the differential pose parameter updates for every joint only from the goal positions of the vertices of the attached body segment, e.g. using the upper arm goal positions to find the shoulder parameters.

Both our artificial body model and the real human body are hierarchical kinematic chains. This implies that transformations of joints lower in the hierarchy involve all transformation of preceding joints too. Taking this into account, we solve for the differential model parameters one hierarchy level of the model at a time, proceeding from top to bottom (level 1 being the highest level, see Figure 3). After the pose updates for a higher level are found, the model parameters on this level are updated, leaving all lower levels unchanged. The algorithm proceeds to the next lower level. Through this method it is assured that the computed differential update corresponds only to a joint transformation on this level.

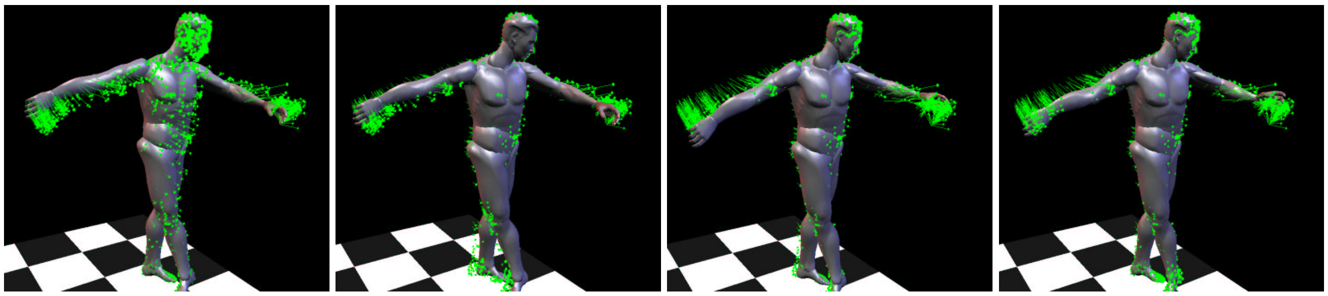

Figure 4. The pictures from left to right show the original model pose with 3D motion field, the model after correction on the first hierarchy level, the second and then the third level.

9.1.1 Registration Method for Pose Update

Finding a pose update for a joint corresponds to finding a coordinate system transformation between two point sets, a problem know as the *absolute orientation problem* in photogrammetry [12]. For each joint, one point set consists of the current 3D vertex positions of the attached body segment. The second point set defines the goal locations for each vertex in 3D space.

In [13], Horn describes a closed form solution to the absolute orientation problem , henceforth referred to as the registration method. In his work, Horn uses quaternions to parameterize rotations. All transformations are computed with respect to the centers of gravity of both point sets. Let $x_{1,i}$ and $x_{2,i}$, $i = \{1, \ldots, N\}$ be corresponding points from two point sets, then the solution to the absolute orientation problem in the least-squares sense are the rotation R and translation c that minimize the error function

$$\sum_i^N \| x_{2,i} - Rx_{1,i} - c \|^2 \tag{6}$$

It is shown in [13] that the optimal translation c is defined by the difference between the centroid of set 2 and the rotated centroid of set 1. To find the optimal rotation, the coordinates of the points in both point sets are defined relative to their center of gravity, respectively. It can be shown that the optimal rotation in the sense of (6) can be found by maximizing

$$\sum_i^N x_{2,i} \cdot Rx_{1,i} \tag{7}$$

The maximal solution to (7) can efficiently be computed in closed-form using a quaternion parameterization q of the rotation. A quaternion can be regarded as a complex number with one real component and three imaginary components, $q = q_0 + q_x i_x + q_y i_y + q_z i_z$, and can be represented by a 4-component vector. Rotations can be represented by unit quaternions. A detailed description of quaternions is beyond the scope of this paper, hence we would like to refer the reader to the paper by Horn [13].

Using quaternions, the sum (7) can be transformed into the form

$$q^T N q \tag{8}$$

The matrix N contains entries that are purely made up of products of coordinates of corresponding points in the two point sets that need to be registered (see Appendix). The rotation q that maximizes this sum is the eigenvector that corresponds to the largest eigenvalue of the symmetric 4x4-matrix N. The solution q is a unit vector in the same direction as the eigenvector.

We apply the registration method to compute differential pose updates as follows. The adjustment starts at hierarchy level 1 with the root of the model. To find the corrective model update of the root joint, a differential rotation and translation is computed using the torso segment start and destination positions computed from D. The rotation component is computed by applying the previously described registration method. The corrective translation is simply the optimal translation of the registration method transformed into the global coordinate system.

On the second level of the hierarchy, only differential rotation parameters for 3-DOF shoulder, hip, and head joints need to be computed. The rotations are to be performed around the center of each joint, not around the center of gravity of the vertex positions. However, it is valid to simply use the start and goal vertex coordinates, $x_{1,i}$ and $x_{2,i}$, defined with respect to the local joint coordinate system instead of relative to the centers of gravity. The same algorithm for finding the optimal rotation as it is part of the registration method still applies. The least-squares rotation for the joint is found as the rotation R that minimizes

$$\sum_i^N \| x_{2,i} - Rx_{1,i} \|^2 \tag{9}$$

This energy term can be expanded into

$$\sum_{i=1}^{N} \parallel x_{2,i} \parallel^2 - 2\sum_{i=1}^{N} x_{2,i} \cdot Rx_{1,i} + \sum_{i=1}^{N} \parallel x_{1,i} \parallel^2 \quad (10)$$

which is minimized by maximizing the middle sum. This sum can be maximized by the same quaternion-based eigenvector decomposition method as previously described.

On hierarchy level 3, there are 4 1-DOF joints (the elbows and the knees). The body model is designed in such a way that the rotation axis for each of these joints coincides with the x-axis of the local coordinate system. The optimal rotations are found using the same procedure as on hierarchy level 2. The 1-DOF constraint is incorporated by simply projecting the start and goal vertex positions into the local yz-planes.

In Figure 4 the different steps of the pose parameter update computation are illustrated using an exaggerated flow field for better visualization.

10. Results

The performance of our system was tested on two multi-view video sequences that were recorded with 8 cameras at a resolution of 320x240 pixels. The sequences show simple gestures that exhibit a large amount of head motion which is difficult to accurately recover for the silhouette step only. The test machine used is a 1.8 GHz Pentium IV Xeon with 512 MB of RAM. For the different sub-components of the algorithm, we obtained the following timing results. For the two sequences, the silhouette fitting takes between 3 and 5s for each time step. The Lukas Kanade optical flow algorithm takes 45s on 8 input views if 4 levels of an image pyramid and a 20x20 Gaussian window are used. The algorithm is configured to compute a motion field vector for each model vertex. The runtime of the optical flow computation strongly depends on the chosen parameters. For only 1 level in the pyramid and a 10x10-neighborhood the optical flows in 8 camera views can be computed in 8s. Our system is flexible enough to include any other optical flow method. The reconstruction of the three-dimensional motion field from the 2D optical flow takes on average 0.34s for each time step.

The results obtained with both sequences show that the motion field update step can noticeably improve the quality of the reconstructed motion and therefore also the reconstructed 3D video. In Figure 5 screenshots of the 3D video system with and without motion field correction are depicted side by side. The most obvious improvements are visible in the face and at the torso. The silhouette step can often not exactly recover the head orientation. The additional use of the texture information can correct for such small errors. Slight changes in torso orientation are also discovered more robustly if the motion field correction step is applied.

	Difference Avg.	Max. Difference
sequence1	0.33 dB	0.81 dB
sequence2	0.35 dB	0.93 dB

Table 1. PSNR measurements

To measure the improvements quantitatively, we computed the peak signal-to-noise-ratio (PSNR) [2] in the luminance channel for both the uncorrected and corrected reconstructed scene with respect to the original segmented input images. In our case, the PSNR is a measure of reconstruction quality. We obtained positive differences between the average PSNRs for both sequences, quantifying the improvement caused by the motion field correction step. The difference in PSNR at one time instant can be even more significant. Results are summarized in Table 10.

It is interesting to observe that, after only small differences at the beginning, at later points in both sequences, the PSNR differences are larger. This validates the assumption that the correction step improves the model fitting over time. Result movies can be downloaded from http://www.mpi-sb.mpg.de/∼theobalt/SceneFlowFitting/index.html.

11. Conclusion

We have presented a new approach for non-intrusively estimating motion parameters which makes use of both silhouette and texture information. This hybrid fitting approach combines the robust fitting of silhouette based tracking with the small-scale accuracy of optical flow methods. Accurate estimation of motion parameters in conjunction with a realistic body model allows for the use of projective texturing to synthesize realistic images of a body in motion. We have demonstrated both empirically and quantitatively that the incorporation of corrective motion fields into the fitting process yields significant improvements, and thus is a worthwhile enhancement to the process of capturing free-viewpoint video.

12. Acknowledgements

The authors would like to thank Ming Li from the Max-Planck-Institut für Informatik for his assistance in recording the multi-view video sequences.

References

[1] J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. *IJCV*, 12:1:43–77, 1994.

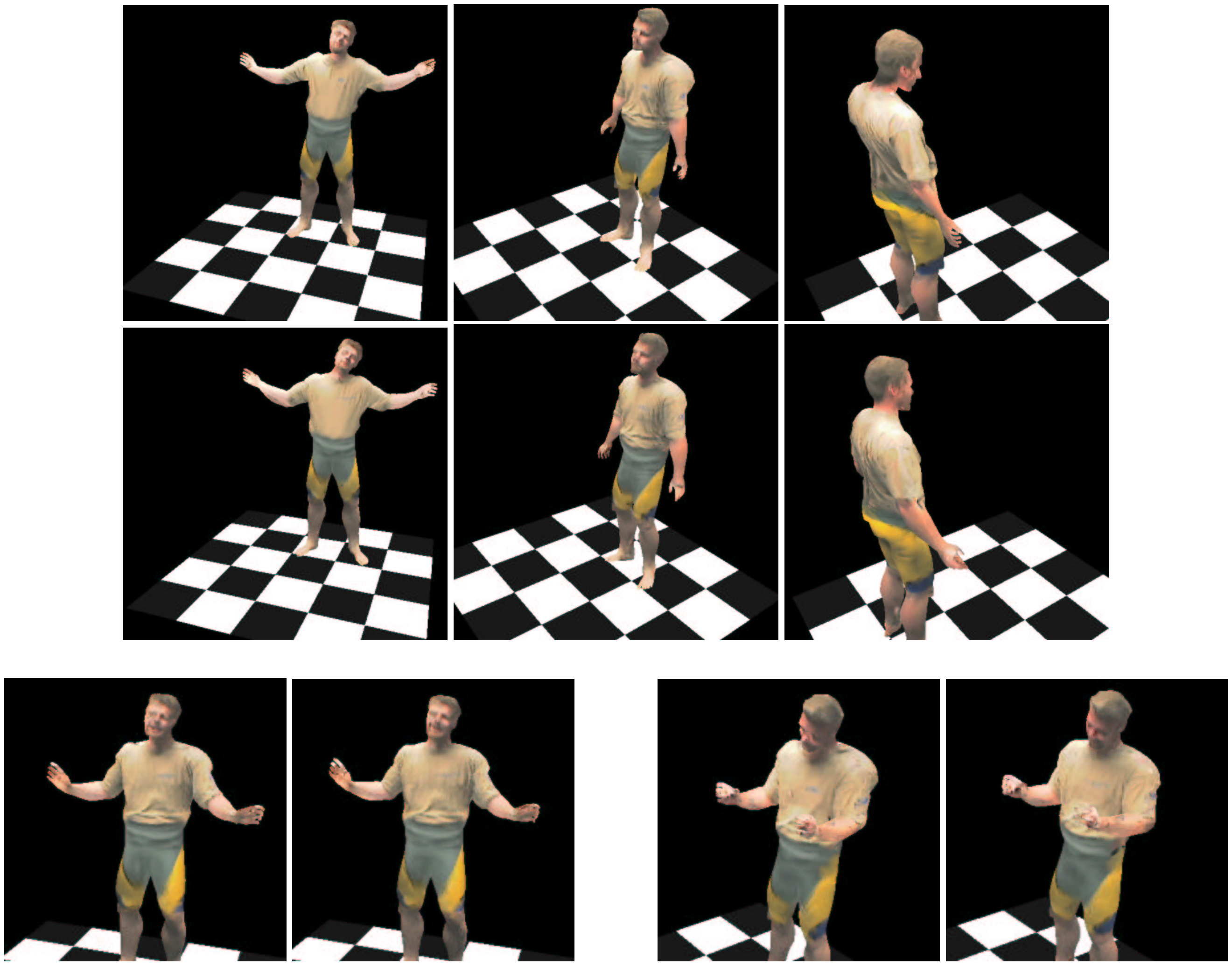

Figure 5. Top row: 3 rendered body poses reconstructed with silhouette-fitting only. Second row : Corresponding body poses with motion field correction from same camera positions. Third row: Two pairs of closeups of the rendered model (left: only silhouette fit, right: with motion field correction). The improvements obtained with the correction step are most obvious in the face since the head pose is more accurately recovered. The torso orientation is improved as well.

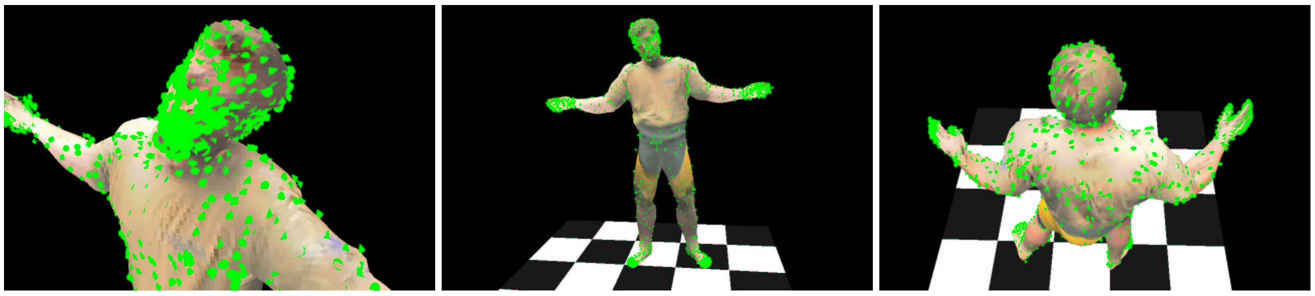

Figure 6. Textured body model with reconstructed corrective 3D motion field rendered as green arrows.

[2] V. Bhaskaran and K. Konstantinidis. *Image and Video Compression Standards*. Kluwer, 1999.

[3] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proc. of CVPR 98*, pages 8–15, 1998.

[4] J. Carranza, C. Theobalt, M. A. Magnor, and H.-P. Seidel. Free-viewpoint video of human actors. In *Proceedings of SIGGRAPH2003*, to appear, page nn, San Diego, USA, 2003. Association of Computing Machinery (ACM), ACM.

[5] K. Cheung, T. Kanade, J.-Y. Bouguet, and M. Holler. A real time system for robust 3D voxel reconstruction of human motions. In *Proc. of CVPR*, volume 2, pages 714 – 720, June 2000.

[6] Q. Delamarre and O. Faugeras. 3D articulated models and multi-view tracking with silhouettes. In *Proc. of ICCV 99*, pages 716–721, 1999.

[7] P. Eisert, W. T., and B. Girod. Model-aided coding : A new approach to incorporate facial animation into motion-compensated video coding. *Transactions on Circuits and Systems for Video Technology*, 10(3):244–258, 2000.

[8] D. Gavrila. The visual analysis of human movement. *CVIU*, 73(1):82–98, January 1999.

[9] D. Gavrila and L. Davis. 3D model-based tracking of humans in action: A multi-view approach. In *CVPR 96*, pages 73–80, 1996.

[10] Y. Guo, G. Xu, and S. Tsuji. Tracking human body motion based on a stick-figure model. *Journal of Visual Communication and Image Representation*, 5(1):1–9, 1994.

[11] D. Hogg. Model-based vision : a program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.

[12] B. Horn. *Robot Vision*. MIT Press, 1986.

[13] B. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Sociey of America*, 4(4):629–642, 1987.

[14] Y. Kameda, M. Minoh, and K. Ikeda. Three dimensional motion estimation of a human body using a difference image sequence. In *Proceedings of the Asian Conference On Computer Vision '95*, pages 181–185, 1995.

[15] R. Koch. Dynamic 3D scene analysis through synthesis feedback control. *PAMI*, 15(6):556–568, 1993.

[16] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. DARPA IU Workshop*, pages 121–130, 1981.

[17] J. Luck and D. Small. Real-time markerless motion tracking using linked kinematic chains. In *Proc. of CVPRIP02*, 2002.

[18] W. Matusik, C. Buehler, and L. McMillan. Polyhedral visual hulls for real-time rendering. In *Proceedings of 12th Eurographics Workshop on Rendering*, pages 116–126, 2001.

[19] P. Narayanan, P. Rander, and T. Kanade. Constructing virtual worlds using dense stereo. In *Proc. of ICCV 98*, pages 3 – 10, January 1998.

[20] R. Plaenkers and P. Fua. Tracking and modeling people in video sequences. *CVIU*, 81(3):285–302, March 2001.

[21] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes*. Cambridge University Press, 1992.

[22] K. Rohr. Incremental recognition of pedestrians from image sequences. In *Proc. of CVPR 93*, pages 8–13, 1993.

[23] C. Theobalt, M. Li, M. Magnor, and H.-P. Seidel. A flexible and versatile studio for synchronized multi-view video recording 2003. In *Proceedings of Vision, Video and Graphics 2003*, page NN, to appear.

[24] C. Theobalt, M. Magnor, P. Schueler, and H.-P. Seidel. Combining 2D feature tracking and volume reconstruction for online video-based human motion capture. In *Proceedings of Pacific Graphics 2002*, pages 96–103, 2002.

[25] R. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'86)*, pages 364–374, June 1986.

[26] S. Vedula, S. Baker, and T. Kanade. Spatio-temporal view interpolation. In *Proceedings of the 13th ACM Eurographics Workshop on Rendering*, pages 65–75, June 2002.

[27] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. In *Proceedings of the 7th IEEE International Conference on Computer Vision (ICCV-99)* [6], pages 722–729.

[28] S. Vedula, S. Baker, S. Seitz, and T. Kanade. Shape and motion carving in 6D, 2000.

[29] S. Wuermlin, E. Lamboray, O. Staadt, and M. Gross. 3D video recorder. In *Proceedings of Pacific Graphics 2002, IEEE Computer Society Press*, pages 325–334, 2002.

Appendix

Structure of Matrix N

The matrix N needed to compute the optimal rotation in a joint is defined as follows : Let $\mathbf{x_1}$ and $\mathbf{x_2}$ be two point sets of size n, each point defined via coordinates (x, y, z), then

$$M = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix},$$

where

$$S_{xy} = \sum_{i=1}^{n} x_{1,i} y_{2,i} \quad ,$$

The entries in N are built via arithmetic operations on elements of M.

$$N = \begin{bmatrix} N_1 & N_2 & N_3 & N_4 \end{bmatrix}$$

$$N_1 = \begin{bmatrix} (S_{xx} + S_{yy} + S_{zz}) \\ S_{yz} - S_{zy} \\ S_{zx} - S_{xz} \\ S_{xy} - S_{yx} \end{bmatrix} \qquad N_2 = \begin{bmatrix} S_{yz} - S_{zy} \\ (S_{xx} + S_{yy} + S_{zz}) \\ S_{xy} + S_{yx} \\ S_{zx} + S_{xz} \end{bmatrix}$$

$$N_3 = \begin{bmatrix} S_{zx} - S_{xz} \\ S_{xy} + S_{yx} \\ (-S_{xx} + S_{yy} - S_{zz}) \\ S_{yz} + S_{zy} \end{bmatrix} \quad N_4 = \begin{bmatrix} S_{xy} - S_{yx} \\ S_{zx} + S_{xz} \\ S_{yz} + S_{zy} \\ (-S_{xx} - S_{yy} + S_{zz}) \end{bmatrix}$$