ARTICULATED 3D MODEL TRACKING WITH ON-THE-FLY TEXTURING

Philipp Fechteler[1,2], *Wolfgang Paier*[1], *Peter Eisert*[1,2]

[1]Fraunhofer HHI, [2]Humboldt Universität zu Berlin, Germany

ABSTRACT

In this paper, we present a framework for capturing and tracking humans based on RGBD input data. The two contributions of our approach are: (a) a method for robustly and accurately fitting an articulated computer graphics model to captured depth-images and (b) on-the-fly texturing of the geometry based on the sensed RGB data. Such a representation is especially useful in the context of 3D tele-presence applications since model-parameter and texture updates require only low bandwidth. Additionally, this rigged model can be controlled through interpretable parameters and allows automatic generation of naturally appearing animations. Our experimental results demonstrate the high quality of this model-based rendering.

Index Terms— Texture synthesis, Pose estimation.

1. INTRODUCTION

The recent increase in virtual and augmented reality applications demands for efficient approaches to capture and render realistic virtual humans, in order to let the users participate in virtual worlds or related applications. With the availability of the Kinect sensor and its API, a broad range of real-time applications have emerged. The motivation for this work is to increase the output quality of such RGBD sensors, which, despite of their popularity, provide only a limited level of quality.

Several approaches for the capturing and modeling of real people exist in literature. For example in [1], realistic 3D models are created from sparse multi-view video. The authors of [2, 3] use a dense array of cameras to capture dynamic scenes. However, despite the high quality of the reconstructed scenes, such methods are not suitable for our purpose because the resulting models cannot be animated and lack a low dimensional representation. In [4] the authors combine visual hull, sparse feature matching and dense reconstruction to create a visually pleasing model from a wide baseline setup of 8 HD-cameras. The resulting model can then be animated using a collection of previously captured motions. The authors of [5, 6] describe a model-based approach to recover the user's geometry, which is useful for tele-presence applications since a low dimensional parameter vector is sufficient to describe the captured content. In [7] the authors combine image-based rendering techniques with an articulated geometric model in order to achieve photo-realistic cloth animations of human motions.

In this paper, we present a framework to process captured color+depth data, that allows for photo-realistic rendering of the subject from arbitrary 3D viewpoints. This approach consists of two steps: (a) fitting a rigged geometric 3D template model to the captured depth map and (b) updating/incrementing the texture map in accordance to the captured subject for photo-realistic rendering. The usage of an underling template model provides increased temporal consistency on a high level of detail. Additionally, an embedded skeleton control structure allows automatic generation of anima-

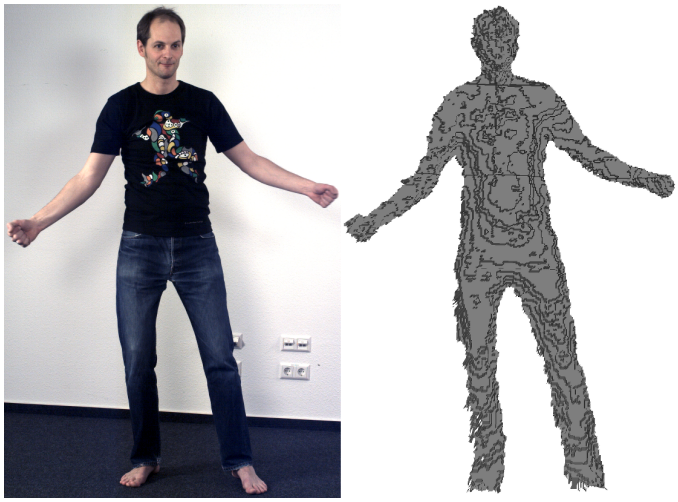

Fig. 1. Typical input: Captured color image and depth mesh.

Fig. 2. Resulting textured models from captured RGBD frames.

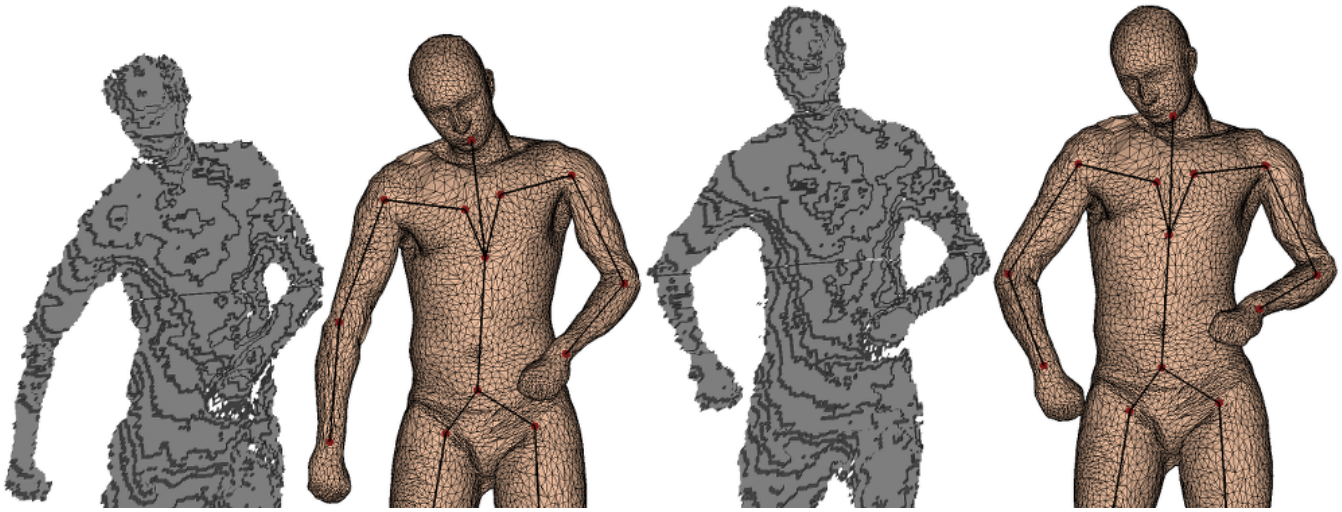

Fig. 3. Kinematic fitting results of consecutive RGBD input frames.

tions or adaptation of captured sequences. In combination with the online generation of a subject specific texture map this approach provides 3D photo-realistic rendering that can be used efficiently for tele-immersion applications.

2. CAPTURING AND PREPROCESSING RGBD DATA

We use a Kinect sensor for depth sensing in combination with an additional RGB video camera to obtain better texture quality compared to the builtin camera. Both sensors are at fixed positions and are tilted vertically to align the aspect ratio with the captured subject. A background model is trained using depth frames of the empty scene to perform a foreground extraction in the depth image which is necessary for the template fitting. Finally, the extracted foreground depth image is converted to a 3D point cloud using the intrinsic calibration of the IR camera. We are using a non-skewed pinhole camera model [8] considering the principle point and focal lengths. In order to allow for mesh based calculations, e.g. comparison of normals, we compute a set of triangles by connecting adjacent depth pixels, if their depth value does not exceed a pre-defined threshold. See Fig. 1 for such an extracted depth mesh.

3. KINEMATIC TEMPLATE FITTING

In order to enhance the quality of the resulting mesh and to allow for pose modifications, we fit an articulated template model to each input frame using an enhanced version of the kinematic template fitting presented in [9]. There are several approaches available to generate kinematic template models with different characteristics and their specific parameters (rotation centers of joints, skinning weights, template vertices) [10, 11, 12, 13]. Our kinematic template model is based on the SCAPE database [14]. For realistic modeling of articulated motion, a combination of Linear Blend Skinning (LBS) and Spherical Linear Interpolation (SLERP) is used as presented in [11]. This skinning approach nicely compensates for the deficiencies (Candy-wrapper, bulging artifacts etc.) of both methods by decomposing the joint rotations. SLERP is used for rotations around the outward bone axis while LBS models bending. The skinning equation $\mathcal{S}$ to transform a template vertex $\mathbf{v}_t$ along the kinematic chain from its rest pose to the desired target pose $\mathbf{v}'_t$ is given as:

$$\begin{aligned}
0 \text{ joints: } \mathbf{v}'_t &= \mathcal{S}(\mathbf{v}_t; s_0, \mathbf{t}_0, \mathbf{R}_0) \\
&= s_0\, \mathbf{R}_0\, \mathbf{v}_t + \mathbf{t}_0 \\
1 \text{ joint: } \mathbf{v}'_t &= \mathcal{S}(\mathbf{v}_t; s_0, \mathbf{t}_0, \mathbf{R}_0, \mathbf{R}_1) \\
&= s_0\, \mathbf{R}_0\, \mathcal{B}(\mathbf{v}_t; \mathbf{R}_1) + \mathbf{t}_0 \\
2 \text{ joints: } \mathbf{v}'_t &= \mathcal{S}(\mathbf{v}_t; s_0, \mathbf{t}_0, \mathbf{R}_0, \mathbf{R}_1, \mathbf{R}_2) \\
&= s_0\, \mathbf{R}_0\, \mathbf{R}_1 \big[\mathcal{B}(\mathbf{v}_t; \mathbf{R}_2) - \mathbf{t}_1\big] + \mathbf{t}_1 + \mathbf{t}_0
\end{aligned} \tag{1}$$

and so forth, using the combined LBS/SLERP blending function $\mathcal{B}$:

$$\begin{aligned}
\mathcal{B}(\mathbf{v}_t; \mathbf{R}_i) = (w^l \mathbf{R}_i^{\text{LBS}} + \tilde{w}^l \mathbf{1}) \\
\big[(w^s \mathbf{R}_i^{\text{SLERP}} + \tilde{w}^s \mathbf{1})(\mathbf{v}_t - \mathbf{t}'_i) + \mathbf{t}'_i - \mathbf{t}_i\big] + \mathbf{t}_i
\end{aligned} \tag{2}$$

with $\mathbf{1}$ being the unity matrix, the pose specific variables for global similarity $s_0, \mathbf{t}_0, \mathbf{R}_0$ as well as the joint rotations $\mathbf{R}_i = \mathbf{R}_i^{\text{LBS}} \mathbf{R}_i^{\text{SLERP}}$. This rotation decomposition can be calculated efficiently using [15]. The template specific fixed parameters are the joint rotation centers $\mathbf{t}_i$ and the per vertex skinning weights w^l, w^s for LBS and SLERP. The LBS weights are constrained to $0 \leq w^l \leq 1$ and $\tilde{w}^l = 1 - w^l$. The SLERP weights are non-linear functions of the rotation angle θ of $\mathbf{R}_1^{\text{SLERP}}$ and the per vertex parameter t: $w^s = \frac{\sin(t \cdot \theta)}{\sin(\theta)}$ and $\tilde{w}^s = \frac{\sin((1-t) \cdot \theta)}{\sin(\theta)}$ with $0 \leq t \leq 1$. A derivation of the SLERP equation, as well as an alternative quaternion formulation can be found in [16]. As already presented in [17], a suitable rotation center handling is critical for SLERP interpolation. Therefore, the SLERP rotation is performed using a vertex specific rotation center $\mathbf{t}'_i$, which is the orthogonal projection of $\mathbf{v}_t$ onto the rotation axis of $\mathbf{R}_i^{\text{SLERP}}$. As a result of using this adapted rotation center $\mathbf{t}'_i$, SLERP interpolates along a cylinder instead of a sphere, which is desirable for our case.

To align the template model with a captured RGBD frame, the pose parameters of the skinning equation $\mathcal{S}$ need to be recovered. Inspired by the efficiency of the Coherent Point Drift algorithm [18], we model the objective function as a probabilistic one-to-many matching problem, with S scan vertices $\mathbf{v}_s$ and T template vertices $\mathbf{v}_t$:

$$Q = \sum_{\substack{s=1\ldots S,\\ t=1\ldots T}} p_{s,t} \big\| \mathbf{v}_s - \mathcal{S}(\mathbf{v}_t; s_0, \mathbf{t}_0, \mathbf{R}_0, \mathbf{R}_1, \ldots) \big\|^2 \tag{3}$$

The critical component in this equation is $p_{s,t}$, the probability of correspondence between template vertex $\mathbf{v}_t$ and scan vertex $\mathbf{v}_s$. An advantage of this one-to-many matching approach is its support for indirect sub-triangle matching and its inherent robustness against noise and outliers.

In order to model Q to have its minimum at the desired template configuration, we embed all available cues into the probability-like weight $p_{s,t}$:

$$p_{s,t} = \frac{\max\left(0, \mathbf{n}(\mathbf{v}_s) \cdot \mathbf{n}(\mathbf{v}'_t)\right)}{\left\|\mathbf{v}_s - \mathbf{v}'_t\right\|^2 + \left\|\text{rgb}(\mathbf{v}_s) - \text{rgb}(\mathbf{v}'_t)\right\|^2 + \epsilon} \quad (4)$$

with $\text{rgb}(\mathbf{v}_s)$ being the corresponding RGB vector of the current scan vertex, $\text{rgb}(\mathbf{v}_t)$ being the corresponding RGB vector of the current texture map (see sec. 4), $\mathbf{n}(\cdot)$ being the normal vector of its argument vertex (calculated by averaging the normals of all connected triangles), and $\epsilon > 0$ being a small constant for numerical stability. Note, that $p_{s,t}$ are just probability-like weights, which do not add to one; this would require additional normalization.

The numerator in (4) enforces stronger influence for correspondences with small angles between their normals, while angles bigger than 90° are completely omitted. This ensures agreement of orientations of corresponding shape regions. The rgb-term causes similarity of captured colors between the scan and the reconstructed texture map generated from previous frames, thereby achieving temporal consistency in the photometric domain.

The actual template fitting is formulated as a minimization problem with costs Q and the template parameters $s_0, \mathbf{t}_0, \mathbf{R}_0, \mathbf{R}_1, \ldots$ as variables. This minimization is achieved by employing the Expectation Maximization algorithm. The E-step consists of updating the $p_{s,t}$ table based on the current template parameter values. In the M-step the template pose parameters are updated, based on the $p_{s,t}$ values calculated in the E-step. This is achieved by using the Gauss-Newton algorithm. Therefore, Δ-perturbations of the template parameters are introduced into the skinning equation $\mathcal{S}$: $s_0 \to \Delta s_0 + s_0, t_0 \to \Delta t_0 + t_0, R_0 \to \Delta R_0 R_0, R_1 \to \Delta R_1 R_1$. By using the following approximation of rotation matrices for small angles

$$\Delta\mathbf{R} \approx \begin{bmatrix} 1 & \Delta r_z & -\Delta r_y \\ -\Delta r_z & 1 & \Delta r_x \\ \Delta r_y & -\Delta r_x & 1 \end{bmatrix} \quad (5)$$

as well as the omission of cross terms in Δ (which corresponds to the omission of higher order terms in Taylor approximation), the skinning equation $\mathcal{S}$ can be rearranged into a linear system of Δ-perturbations having the form $v' = \mathbf{A}\Delta\mathbf{p} + \mathbf{a}$. $\Delta\mathbf{p}$ is the vector of Δ-perturbations and matrix $\mathbf{A}$ and vector $\mathbf{a}$ are the rearranged elements of the costs Q of the objective function (3). The resulting equation system can be solved efficiently in a linear least squares sense. In order to prevent divergence in ambiguous cases, only small parameter updates from frame to frame are allowed. This is achieved by adding a Tikhonov regularization term to the linear least squares equation system. Results of this kinematic model fitting are shown in Fig. 3.

This kinematic fitting approach relies on suitable initialization values for the parameters. For the first frame, we assume the subject to be in a certain pre-defined pose in order to achieve a sufficient prealignment with the Iterative Closest Point algorithm. Alternatively, a more generic approach, like Shape Similarity Trees [19], could be adopted. Since the rgb-term in (4) depends on previously processed frames, it is omitted during initialization. In all successive frames, the estimated parameters of the previous frame are used as an initial guess for the current frame. Although this strategy turns out to be sufficient, better prediction schemes, e.g. Kalman- or Particle filter, are straightforward to integrate in order to handle more challenging scenarios like fast and/or complex movement.

4. TEXTURE GENERATION

Our texture generation builds upon the kinematic template fitting and uses the resulting mesh as scene representation. We assume that each vertex of the template mesh has constant uv coordinates, that have been created once by a semiautomatic approach for mesh unwrapping.

Since the RGB camera only provides texture information for the currently visible part of the template mesh, a set of previously captured images is maintained to provide enough information for a complete texture map. Redundant and old images are discarded to keep memory and CPU load low.

Not all visible parts of the mesh can be textured equally well from one color image, since visual details depend on the distance, occlusions, and viewing direction $\mathbf{v}$. Therefore, a quality measure $\mathcal{W}(f_i, l_i)$ is employed to select the best source image I_{l_i} for each triangle f_i of the template mesh. Since the distance between object and camera does not vary considerably in our setup, we use the following simple quality measure:

$$\mathcal{W}(f_i, l_i) = \max(0, -\mathbf{n}_{f_i}^T \mathbf{v}), \quad (6)$$

with $\mathbf{n}$ being the surface normal direction. More sophisticated variants of $\mathcal{W}$ can be found in [20, 21, 22, 23]. The texture is constructed by sampling each texel in its corresponding source image. To avoid blur or ghosting caused by inaccurate geometry or tracking errors, only one source image is used per triangle.

Treating all triangles independently leads to a non-smooth solution that results in a high number of visible edges between adjacent triangles (see Fig. 4). Therefore, a smoothness term $\mathcal{V}_{i,j}$ is added for each pair of adjacent triangles (f_i,f_j), resulting in the following discrete optimization problem:

$$\mathcal{E}(L) = \sum_i^n \mathcal{D}_i(f_i, l_i) + \lambda \sum_{i,j \in \mathcal{N}} \mathcal{V}_{i,j}(l_i, l_j). \quad (7)$$

$L = \{l_1, ..., l_n\}$ is a label vector that assigns each triangle f_i one image I_{l_i} as texture source and $\mathcal{D}_i$ is an error term that corresponds to the quality measure of image I_{l_i} for triangle f_i.

$$\mathcal{D}(f_i, l_i) = \begin{cases} 1 - \mathcal{W}(f_i, l_i) & f_i \text{ is visible} \\ \infty & f_i \text{ is occluded} \end{cases} \quad (8)$$

To reduce the visibility of seams, $\mathcal{V}_{i,j}$ adds costs $\Pi_{e_{i,j}}$ depending on the sum of color differences along the common edge $e_{i,j}$ of f_i and f_j.

$$\Pi_{e_{i,j}} = \int_{e_{i,j}} \left\| I_{l_i}(x) - I_{l_j}(x) \right\| dx \quad (9)$$

$$\mathcal{V}_{i,j}(l_i, l_j) = \begin{cases} 0 & l_i = l_j \\ \Pi_{e_{i,j}} & l_i \neq l_j \end{cases} \quad (10)$$

This way, seams are more likely to occur in regions where adjacent triangles look similar, thus hiding them. To reduce the visibility of remaining seams, we use seam levelling [23] for the final construction of a texture. This technique adds piece-wise continous functions to the constructed texture mosaic that remove strong edges at seams caused by different brightness or white balance of the source images.

Fig. 4. Results without (left) and with smoothness term (right).

Minimizing (7) yields a label vector $L = \{l_1, ..., l_n\}$ that contains the optimal source image for each triangle in terms of spatial resolution and visibility of seams. Since (7) is an NP-hard problem, we calculate an efficient but approximate solution by transforming the original multi-label problem into a series of binary label problems that can be solved optimally using s-t graph cuts as explained in [24].

Since rapidly moving seams draw attention, we also modify $\mathcal{V}_{i,j}$ to smooth their position over time. This can be implemented by augmenting $\mathcal{V}_{i,j}$ with $\Psi_{i,j}$ which adds costs corresponding to the distance of the seam (edge $e_{i,j}$) to previous seams $e'_{k,l}$. $\alpha \in [0, 1]$ is a scalar weight to control the influence of $\Psi_{i,j}$ and $dst(e1, e2)$ defines the shortest distance between two edges $e1$ and $e2$.

$$\Psi_{i,j} = \min_{k,l} dst(e_{i,j}, e'_{k,l}) \tag{11}$$

$$\mathcal{V}_{i,j}(l_i, l_j) = \begin{cases} 0 & l_i = l_j \\ (1-\alpha)\Pi_{e_{i,j}} + \alpha\Psi_{i,j} & l_i \neq l_j \end{cases} \tag{12}$$

To keep the texture up to date, the lifetime of every source image must be considered. Therefore, $D_i(f_i, l_i)$ has to account for the lifetime t_i:

$$t'_i = \min(t_{max}, t_i) \tag{13}$$

$$\mathcal{D}(f_i, l_i) = \begin{cases} \beta\big(1 - \mathcal{W}(f_i, l_i)\big) + (1-\beta)\frac{t'_i}{t_{max}} & f_i \text{ is visible} \\ \infty & f_i \text{ is occluded} \end{cases} \tag{14}$$

t'_i is a truncated version of t_i for simple normalization and $\beta \in [0, 1]$ is a scalar weight to control the influence of visual quality and actuality.

To keep the number of source images low, old images should be removed if they do not contribute a significant part to the final texture, that cannot be provided by any other source image at comparable quality. Therefore a coarse histogram $\mathbf{h}_i$ of $\mathcal{W}(f_i, l_j)$ for each triangle f_i can be used. Starting with the oldest image, I_{l_j} is discarded if no triangle f_i exists for which image I_{l_j} provides the only entry in the best non-empty bin of $\mathbf{h}_i$.

5. EXPERIMENTAL RESULTS

In our experiments, we capture the depth with a Kinect depth sensor at a 480x640 resolution (tilted vertically to align the aspect ratio with the capturing target). To improve the visual quality, we use a Basler acA2040-25gc at a resolution of 1600x2032 instead of the integrated Kinect color sensor. Both sensors are driven at 30Hz.

The discussed tracking and texturing system is developed as enhancement of a networked 3D tele-conferencing system [25]. The aim is to provide several participants with a visual face-to-face experience during remote communication. Therefore, photo-realistic 3D visualizations of the participants are captured and rendered (as presented in this paper) into a shared 3D virtual world by streaming model parameters and texture updates. In Fig. 1 the typical input to our system, consisting of a captured RGB frame and a depth mesh is shown. Figure 2 displays some output of the presented framework. The texture was created using several frames of the input RGBD-stream. The template model was animated by modifying the low dimensional parameter vector to obtain new poses.

Our template model is based on the publicly available SCAPE database [14] and consists of approx. 12500 vertices and 25000 triangles. To allow for realistic animations, we have embedded 17 joints, resulting in 58 parameters (including 7 parameters for global similarity).

Using the one-to-many matching approach for our kinematic template fitting significantly increased the tracking stability in comparison to the usual one-to-one matching. However, manual correction was still required in cases of complex, fast movements.

The computational complexity of the kinematic template fitting is quite high because of the one-to-many matching approach. Our implementation of the model fitting requires several minutes per frame. In order to achieve real-time performance, a requirement for interactivity, we are currently focusing on mainly two optimizations:

- limitation of the number of the objective function equations, e.g. by filtering small $p_{s,t}$, and
- exploiting the inherent potential for parallelism by developing a fast GPU implementation, e.g. the equation system can be setup independently per vertex and fused afterwards.

The kinematic template fitting converges in our experiments on average within 15 iterations. The multi-label optimization converges quickly, since each new color image affects only some parts of the texture and the previous solution is used as initial label vector.

6. CONCLUSION

In this paper, we have presented a model-based approach for capturing and tracking of humans with incremental texture map generation. For increased accuracy and robustness, a one-to-many matching approach, based on depth cues as well as photometric consistency, is used. The on-the-fly generation of texture maps achieves up-to-date and visually pleasing results using a graph-cut based mosaicing approach and seam-levelling. A set of source images is maintained during runtime to provide enough information for a complete texture without using too much resources. For performance evaluation purposes a calibrated depth sensor and a RGB camera setup are used as input devices.

7. ACKNOWLEDGMENTS

The research that lead to this paper was supported in part by the European Commission under the Contract FP7-ICT-287723 REVERIE.

8. REFERENCES

[1] E.d. Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun, "Performance Capture from Sparse Multi-view Video," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 98:1–98:10, Aug. 2008.

[2] Y. Liu, Q. Dai, and W. Xu, "A Point-Cloud-Based Multiview Stereo Algorithm for Free-Viewpoint Video," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 3, pp. 407–418, May 2010.

[3] K. Li, Q. Dai, W. Xu, and J. Yang, "Temporal-Dense Dynamic 3-D Reconstruction With Low Frame Rate Cameras," *J. Sel. Topics Signal Processing*, vol. 6, no. 5, pp. 447–459, 2012.

[4] J. Starck and A. Hilton, "Surface Capture for Performance-Based Animation," *IEEE Comput. Graph. Appl.*, vol. 27, no. 3, pp. 21–31, May 2007.

[5] C. Theobalt, N. Ahmed, H. Lensch, M. Magnor, and H.-P. Seidel, "Seeing People in Different Light-Joint Shape, Motion, and Reflectance Capture," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 4, pp. 663–674, July 2007.

[6] J. Carranza, C. Theobalt, M.A. Magnor, and H.-P. Seidel, "Free-viewpoint Video of Human Actors," in *ACM SIGGRAPH 2003 Papers*, New York, NY, USA, 2003, SIGGRAPH '03, pp. 569–577, ACM.

[7] A. Hilsmann, P. Fechteler, and P. Eisert, "Pose Space Image Based Rendering," *Comput. Graph. Forum (Proc. Eurographics)*, vol. 32, no. 2, pp. 265–274, May 2013.

[8] R.I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[9] P. Fechteler, A. Hilsmann, and P. Eisert, "Kinematic ICP for Articulated Template Fitting," in *Proceedings of the 17th International Workshop on Vision, Modeling and Visualization (VMV2012)*, Magdeburg, Germany, 12-14th November 2012.

[10] A. Mohr and M. Gleicher, "Building Effcient, Accurate Character Skins from Examples," *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, vol. 22, no. 3, pp. 562–568, July 2003.

[11] L. Kavan and O. Sorkine, "Elasticity-Inspired Deformers for Character Articulation," *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA)*, vol. 31, no. 6, pp. 196:1–196:8, 2012.

[12] B. Levy, X. Tong, and KK. Yin, "A Data-Driven Approach to Efficient Character Articulation," *Submitted to Pacific Graphics*, vol. 32, no. 7, 2013.

[13] B. Huy Le and Z. Deng, "Smooth Skinning Decomposition with Rigid Bones," *ACM Transactions on Graphics (SIGGRAPH2012)*, vol. 31, no. 6, December 2012.

[14] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, "SCAPE: Shape Completion and Animation of People," *ACM Trans. Graph*, vol. 24, pp. 408–416, 2005.

[15] G. Piovan and F. Bullo, "On Coordinate-Free Rotation Decomposition: Euler Angles about Arbitrary Axes," *IEEE Transactions on Robotics*, vol. 28, no. 3, pp. 728–733, 2012.

[16] E.B. Dam, M. Koch, and M. Lillholm, "Quaternions, Interpolation and Animation," Tech. Rep., University of Copenhagen, Denmark, 1998.

[17] L. Kavan, S. Collins, J. Žára, and C. O'Sullivan, "Geometric Skinning with Approximate Dual Quaternion Blending," *ACM Transactions on Graphics*, vol. 27, no. 4, pp. 105:1–105:23, Nov. 2008.

[18] A. Myronenko and X. Song, "Point Set Registration: Coherent Point Drift," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 2262–2275, 2010.

[19] C. Budd, P. Huang, and A. Hilton, "Hierarchical Shape Matching for Temporally Consistent 3D Video," in *Proceedings of Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, Hangzhou, China, 16-19 May 2011, pp. 172–179.

[20] Z. Janko and J.-P. Pons, "Spatio-Temporal Image-Based Texture Atlases for Dynamic 3-D Models," in *IEEE International Workshop on 3-D Digital Imaging and Modeling*, Kyoto, Japon, Oct. 2009, pp. 1646–1653, IEEE.

[21] J. Carranza, C. Theobalt, M.A. Magnor, and H.-P. Seidel, "Free-viewpoint Video of Human Actors," in *ACM SIGGRAPH 2003 Papers*, New York, NY, USA, 2003, SIGGRAPH '03, pp. 569–577, ACM.

[22] C. Allene, J.-P. Pons, and R. Keriven, "Seamless image-based texture atlases using multi-band blending," in *ICPR*. 2008, pp. 1–4, IEEE.

[23] V.S. Lempitsky and D.V. Ivanov, "Seamless Mosaicing of Image-Based Texture Maps," in *CVPR*. 2007, IEEE Computer Society.

[24] Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1124–1137, Sept. 2004.

[25] P. Fechteler, A. Hilsmann, and P. Eisert et al, "A Framework for Realistic 3D Tele-Immersion," in *Proceedings of the 6th International Conference on Computer Vision / Computer Graphics Collaboration Techniques and Applications (MIRAGE2013)*, Berlin, Germany, 6th-7th June 2013.