SPA: Sparse Photorealistic Animation Using a Single RGB-D Camera

Kun Li, Jingyu Yang, Leijie Liu, Ronan Boulic, Yu-Kun Lai, Yebin Liu, Yubin Li, and Eray Molla

***Abstract*— Photorealistic animation is a desirable technique for computer games and movie production. We propose a new method to synthesize plausible videos of human actors with new motions using a single cheap RGB-D camera. A small database is captured in a usual office environment, which happens only once for synthesizing different motions. We propose a marker-less performance capture method using sparse deformation to obtain the geometry and pose of the actor for each time instance in the database. Then, we synthesize an animation video of the actor performing the new motion that is defined by the user. An adaptive model-guided texture synthesis method based on weighted low-rank matrix completion is proposed to be less sensitive to noise and outliers, which enables us to easily create photorealistic animation videos with new motions that are different from the motions in the database. Experimental results on the public data set and our captured data set have verified the effectiveness of the proposed method.**

***Index Terms*— Marker-less performance capture, photorealistic animation, RGB-D camera, sparse representation, texture synthesis.**

I. Introduction

PHOTOREALISTIC animation aims to create a plausible photorealistic video of an actor performing a new motion based on a database, which is highly desirable for both computer games and movie production [1]–[3]. On the one hand, the rendered results of fully animated human characters are not realistic and on the other hand, the captured videos are difficult to synthesize new motions. Video texture methods [4], [5] attempt to generate videos with new motions by rearranging the subsequences, but it is difficult to create truly new motions. Recently, some methods [6], [7] set up a multicamera system to help better synthesize videos with new motions. They achieve promising results with the help of multiview information. However, systems of this kind are expensive, difficult to maintain, and need many manual operations. Moreover, they need the actor to clench their hands. In this paper, we try to achieve photorealistic animation using a single cheap RGB-D camera by capturing the database in a usual office environment. Our system is nonintrusive and easy to set up.

The input for photorealistic animation contains several videos with various basic motions, while the output is a plausible video corresponding to a new motion defined by users. Creating a new video with new motions from limited existing motions is essentially an ill-posed problem, because the limited available textures of motions do not contain complete information to reconstruct the textures for the new motion. Moreover, the appearance of actors continually changes with their motions when they perform different motions, such as folds and wrinkles. The key to generate appealing textures is to impose proper priors to make the problem well posed. Recently, sparse representation has shown its great power in the regularization of the ill-posed estimation problem, such as in surface reconstruction [8], [9], 3D shape denoising [10], and depth enhancement [11].

Data-driven photorealistic animation method can effectively use knowledge from the database. By using the sparse priors, we can reduce the dependency on high quality or complete input, which makes it possible to use a single RGB-D camera for photorealistic animation. In our work, we propose a new method to synthesize photorealistic videos of human actors with user-defined motions based on a small database captured by a single RGB-D camera in a usual office environment, allowing a small change of viewpoint. We present a sparse deformation optimization method to make the marker-less performance capture less affected by noise and outliers, which gives a pivotal constraint for video synthesis. Besides, we address the video synthesis problem by adaptive weighted low-rank matrix completion. Our method offers the following advantages.

1) *Cheap Nonintrusive System:* Our method achieves photorealistic animation using only a single cheap RGB-D camera.
2) *Less Requirement:* Our method does not require the actor to wear skin-tight garments, attach markers, or clench the hands.
3) *Accuracy:* By using sparse priors, our method can recover the textures with high accuracy from a very small data set.
4) *Robustness:* By using sparse representation, our method is less sensitive to noise and outliers than previous methods.

Manuscript received August 19, 2015; revised December 25, 2015; accepted April 5, 2016. Date of publication April 20, 2016; date of current version April 3, 2017. This work was supported by the National Natural Science Foundation of China under Grant 61571322, Grant 61302059, Grant 61372084, and Grant 61522111 and by the National Key Foundation for Exploring Scientific Instrument under Grant 2013YQ140517. This paper was recommended by Associate Editor P. Eisert. *(Corresponding author: Jingyu Yang.)*

K. Li and Y. Li are with Tianjin Key Laboratory of Cognitive Computing and Application, School of Computer Science and Technology, Tianjin University, Tianjin 300072, China (e-mail: lik@tju.edu.cn).

J. Yang and L. Liu are with the School of Electronic Information Engineering, Tianjin University, Tianjin 300072, China (e-mail: yjy@tju.edu.cn).

R. Boulic and E. Molla are with the School of Computer and Communications, École Polytechnique Fédérale de Lausanne, Lausanne 1015, Switzerland.

Y.-K. Lai is with the School of Computer Science and Informatics, Cardiff University, Cardiff CF10 3XQ, U.K.

Y. Liu is with the Department of Automation, Tsinghua University, Beijing 10084, China.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2016.2556419

We demonstrate the power and effectiveness of our method on a public data set and our captured data sets. Our method achieves plausible animation videos for all the new motions.

The main contributions of this paper are as follows.

1) *Photorealistic Animation Using a Single Cheap RGB-D Camera in a Usual Environment:* We capture the database using a Kinect v2.0 camera in a usual office environment, and generate a plausible photorealistic animation video using the proposed method.
2) *Marker-Less Performance Capture Method:* To obtain more consistent results with the real motion, we propose a two-step marker-less performance capture method. A multipriority inverse kinematics method and the skinning method are first used to get an initial mesh for each frame, and then a sparse deformation method is adopted to generate more accurate meshes.
3) *Adaptive Sparse Texture Synthesis Method:* To recover the textures from limited data set with different motions, we propose an adaptive weighted low-rank matrix completion method. Through this method, not only is the frame that has high percentages of missing data recovered, but also the noise and outliers in the initially estimated image are reduced.

The remainder of this paper is structured as follows. Section II provides a brief review of related work in the field of animation. A system overview of the proposed method is given in Section III. The technical details of database setup, retrieval, and video synthesis are described in Sections IV–VI, respectively. Validation experiments and results are presented in Section VII, and this paper is concluded in Section VIII.

II. Related Work

This section provides a brief review of related work in the field of animation. Animation approaches are mainly divided into three categories: 1) skeleton-based methods; 2) model-based methods; and 3) image-based methods. The most common skeleton-based method is linear blend skinning (LBS) [12], but this method has bad twisting animation result around joints. To overcome this problem, Wang and Phillips [13] and Merry *et al.* [14] compute different weights from different poses of models. Mohr and Gleicher [15] generate animation by introducing some dummy joints. Kavan *et al.* [16] replace the classic LBS method by a dual quaternions skinning method. However, the results of these methods look unrealistic.

Model-based methods can generate more realistic animations by building a database that has some 3D model samples with high accuracy and strong sense of reality. De Aguiar *et al.* [17] use principal component analysis to reduce the dimensions of human body models and clothing models, and then learn their relationship by linear regression to generate new models with target motions. Wang *et al.* [18] analyze surface vertices for different body parts, and choose different samples to synthesize the 3D model with a new motion. However, these kinds of methods usually demand very high computational complexities, and need to sacrifice some accuracy and authenticity for reasonable computational complexity.

Image-based methods aim at generating realistic textures and have low computational complexity. The video texture method [4] is an early work in this field, which analyzes a video clip to extract its structure and creates a new video by rearranging the frames. As an extension, video sprites [19] are proposed to animate moving objects. It finds good frame arrangements based on repeated partial replacements of the sequence, which allows the user to specify animations using a flexible cost function.

It is challenging to use the above methods to generate new videos for moving humans without any knowledge of 3D shapes and poses. Celly and Zordan [20] preliminarily solve the problem by identifying transition regions with human-specific feature extraction and performing an image-based warping afterward. Flagg *et al.* [5] adopt marker-based motion capture and construct a video-based motion graph to synthesize a new video.

All the above methods generate a new video by rearranging the subsequences with specific motions, and no novel motions can be created. Starck *et al.* [21] use 3D video sequences by combining image-based reconstruction and video-based animation to allow controlled animation of human body from captured multiview video sequences. The blended video sequences are constructed offline and represented as a motion graph for interactive animation. Similarly, Huang *et al.* [22] synthesize novel 3D video sequences by finding the optimal path in the motion graph between user-specified key-frames for control of movement, location, and timing. Casas *et al.* [6] introduce a new representation, 4D video texture, for rendering photorealistic animations from a multiview multimotion database. However, the generated new sequences are restricted to the space of basic motions in the database, and texture synthesis is not considered. Xu *et al.* [7] set up a multicamera system with 12 cameras and achieve realistic video synthesis of a novel target motion using a model-guided image warping method. However, this method relies on an expensive multicamera system, and requires the actor to clench hands. When the query motion is different from the basic database motions and less view information is available, there are obvious blurring artifacts on the recovered textures. In contrast, with sparse representation, our approach only uses a single cheap camera and achieves plausible animation videos without such artifacts.

The Microsoft Kinect camera has been widely used due to its low cost and multisensing [23]–[25]. The version 2.0 sold last year has more accuracy in color, depth, and skeleton tracking. In this paper, we use a single Kinect v2.0 camera to capture a multimotion database. A multipriority inverse kinematics method is used to ensure the topology consistency of Kinect skeletons and a sparse deformation method is proposed to ensure the accuracy of mesh deformation. For synthesizing textures, instead of using existing projective texturing and blending methods [26], [27], which tend to produce texture ghosting, we propose a new adaptive model-guided texture synthesis method based on weighted low-rank matrix completion.

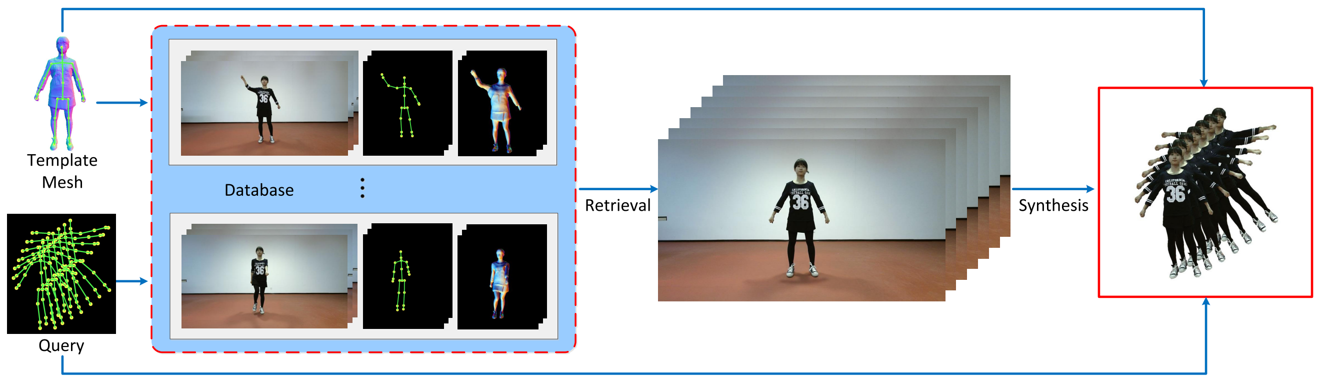


Fig. 1. Framework of the proposed method. The system contains a small database of an actor performing several basic motions: color images are captured by a single Kinect camera and skeletons and dynamic meshes are calculated by a marker-less performance capture method. The user gives a new query skeleton sequence with a new motion. We select the similar images from the database and synthesize a photorealistic video of the actor performing the new motion using an adaptive model-guided sparse texture reconstruction (STR) method.

III. System Overview

Fig. 1 illustrates the workflow of our sparse photorealistic animation approach. The input to our system is a skeleton sequence and a rigged surface mesh of an actor. The output is a synthesized photorealistic animation video with the input motion. To make this possible, we take a data-driven approach. A small database is first set up offline, which contains videos and meshes with skeletons performing various basic motions (Section IV). Then, appropriate images are retrieved from the database according to the input skeleton sequence (Section V). Finally, a photorealistic animation video is synthesized based on adaptive weighted low-rank matrix completion using the retrieved images (Section VI).

Database: We use a Kinect camera to capture RGB-D images of a character performing various basic motions, and obtain a 3D scanned template mesh of the character using the Kinect Fusion technique [28]. Then, we manually embed a skeleton into the template mesh, compute the skinning weight [29], and segment the body of the mesh into 16 parts according to the maximum skinning weight of each vertex. Finally, we compute a 3D skeleton and a 3D surface mesh for each frame of the database. Therefore, the database contains a color image sequence, a skeleton sequence, and a mesh sequence for each motion. The database is only needed to be built once for generating new motions of the same character.

Query: A query sequence is a user-defined skeleton sequence that specifies the actor motion in the target video, which uses the same template mesh model and skeleton as the database. Arbitrary animation tools can be used to define the whole animation sequence. Alternatively, it is also possible to use motion retargeting techniques [30] to apply motion capture data from existing databases to the given skeleton.

Retrieval: A candidate image is retrieved from the database according to the spatiotemporal similarity for each frame of target motion. To achieve high performance retrieval, we consider the similarity of motion and the completeness of sequence in our scheme.

Synthesis: We synthesize the target sequence using the retrieved frames based on adaptive weighted low-rank matrix completion, which can recover the matrix that has high percentages of missing data and can also reduce the noise and outliers in the known elements.

IV. Database Setup

We capture the color images, depth images, and skeletons for a database using a single Kinect v2.0 camera, and then we compute and optimize the skeletons and dynamic meshes using the proposed method.

A. Acquisition

We capture an actor (actress) performing various basic motions using a Kinect v2.0 camera. The basic motions we used are *walking with flexed legs*, *running*, *marching*, *waving*, *stretching*, *front-kicking*, *side-kicking*, *salute*, and *kungfu*. Each depth image is transformed into a depth mesh and segmented by an RANSAC algorithm to remove the floor and a bounding box of the actor to remove the background. Skeleton is identified using a pose recognition method [31] at each time instance.

Besides, we also obtain a 3D scanned mesh without textures of the actor using the Kinect camera by capturing the 3D point clouds in a circle and merging them together using the Kinect Fusion technique [28]. The Kinect Fusion method only works for rigid objects, so the actor being scanned needs to stand still and the Kinect camera is controlled by another person to scan from 360° directions. The underlying skeleton of the mesh is generated by manually marking the joint positions. We calibrate the coordinate system relationship of the static template mesh and the captured depths of Kinect by computing the rigid transformation using four skeleton joints (spine base, spine center, left hip, and right hip) of the skeleton of the template mesh and the skeleton of the first frame of Kinect.

B. Marker-Less Performance Capture

With the 3D template mesh and the captured depth+skeleton sequences, we present a skeleton-based marker-less performance capture method to obtain a new skeleton sequence and a dynamic mesh sequence for each motion. Because the skeletons captured by Kinect do not maintain

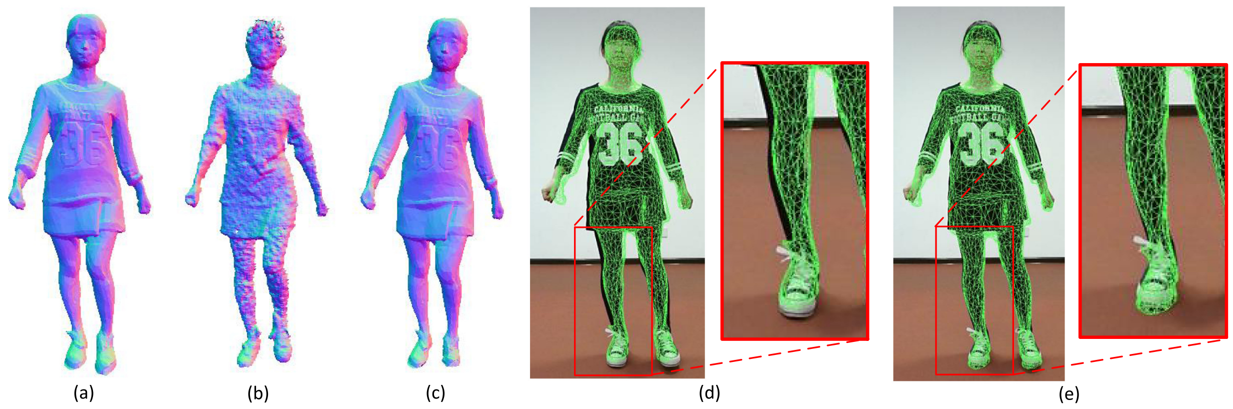


Fig. 2. Example of sparse deformation optimization. (a) Mesh before optimization. (b) Depth mesh. (c) Mesh after optimization. (d) Mesh projection before optimization. (e) Mesh projection after optimization.

topological consistency, we use a multipriority inverse kinematics method [32] to obtain pose parameters and new skeletons with the consistent topology. The skinning weights are automatically calculated for each vertex, which describe the association of the vertex with each bone [29]. With these weights, an LBS method is adopted to deform the template mesh using the calculated pose parameters. Then, we optimize the deformed meshes with depths based on sparse representation to make the deformed mesh consistent with the data captured by Kinect.

Let us define the deformed mesh by the LBS method as $\mathscr{M}_s$ and the vertices of $\mathscr{M}_s$ as $\mathscr{P} \triangleq \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$, where $\mathbf{p}_i \triangleq [x_i, y_i, z_i, 1]^\top$ and N is the number of vertices. Similarly, we denote the vertices of the depth mesh $\mathscr{M}_t$ by $\mathscr{Q} \triangleq \{\mathbf{q}_1, \ldots, \mathbf{q}_M\}$. We compute the visible vertices of the deformed mesh at the viewpoint of the depth camera and find the closest correspondences on the depth mesh. Specifically, we find the closest point for each vertex of the deformed mesh and calculate the distance between this vertex and its closest point. Then, we choose the ones whose distances are smaller than the median as the correspondences. Denote the correspondence point of $\mathbf{p}_i \in \mathscr{P}$ by $\mathbf{q}_{f(i)} \in \mathscr{Q}$, where $f : \{1, \cdots, N\} \mapsto \{1, \ldots, M\}$ represents the index mapping. Given the correspondence mapping f [33], we compute a 3×4 transformation matrix $\mathbf{T}_i$ for each vertex of $\mathscr{M}_s$ by minimizing the following energy function:

$$E = \sum_{\mathbf{p}_i \in \mathscr{P}} \omega_i \left\|\mathbf{T}_i\mathbf{p}_i - \tilde{\mathbf{q}}_{f(i)}\right\|_2^2 + \gamma \sum_{\mathbf{p}_i \in \mathscr{P}} \sum_{\mathbf{p}_j \in \mathscr{N}_i} \pi_{ij} \left\|\mathbf{T}_i - \mathbf{T}_j\right\|_1 \tag{1}$$

where $\tilde{\mathbf{q}}_{f(i)}$ is the Cartesian coordinate of $\mathbf{q}_{f(i)}$, $\mathscr{N}_i$ represents one round neighborhood connecting with edges, and $\|\cdot\|_1$ represents the ℓ_1 norm of the matrix. $\pi_{ij} = \exp(-H^2/\sigma^2)$ allows larger nonrigid transformation around curved places, where H is the mean curvature and σ is a constant. The weight ω_i is set at zero if $\mathbf{p}_i$ does not have a corresponding point in $\mathscr{Q}$, and set at one otherwise. The first term reflects the fidelity of the estimated transformations and ensures the accuracy of the reconstruction, while the second term as regularization ensures the smoothness of the transformations.

We further define a differential matrix $\mathbf{L} \in \{-1, 1\}^{G \times N}$ with G representing the number of edges of $\mathscr{M}_s$. Each row of $\mathbf{L}$ corresponds to an edge of $\mathscr{M}_s$, and each column of $\mathbf{L}$ corresponds to a vertex of $\mathscr{M}_s$. For the rth edge that connects vertex $\mathbf{p}_i$ and vertex $\mathbf{p}_j$, we have $L_{r,i} = 1$ and $L_{r,j} = -1$. Therefore, (1) can be rewritten as

$$\min_{\mathbf{T},\mathbf{X}} \left\|\mathbf{\Omega}\left(\mathbf{PT} - \tilde{\mathbf{Q}}_f\right)\right\|_F^2 + \gamma \left\|\mathbf{\Pi} \circ \mathbf{X}\right\|_1, \quad \text{s.t.} \quad \mathbf{X} = (\mathbf{L} \otimes \mathbf{I}_4)\,\mathbf{T} \tag{2}$$

where

$$\begin{aligned} \mathbf{\Omega} &= \operatorname{diag}\left(\sqrt{\omega_1}, \ldots, \sqrt{\omega_N}\right) \\ \mathbf{P} &= \operatorname{diag}\left(\mathbf{p}_1^\top, \ldots, \mathbf{p}_N^\top\right) \\ \tilde{\mathbf{Q}}_f &= \left[\tilde{\mathbf{q}}_{f(1)} \ldots \tilde{\mathbf{q}}_{f(N)}\right]^\top \end{aligned} \tag{3}$$

$\mathbf{T}$ is a $4N \times 3$ matrix by taking $\mathbf{T}_i$ as its column, $\mathbf{\Pi}$ is a weight matrix with π_{ij} as its elements, "$\circ$" represents element-wise multiplication of two matrices, $\mathbf{I}_4$ is a 4×4 identity matrix, and $\otimes$ denotes the operator of the Kronecker product. We iteratively find the closest correspondences and solve (2) using the alternate direction method (ADM) [34] until convergence. In our experiments, we use 15 outer iterations for sparse deformation and 25 inner iterations for the ADM algorithm. Fig. 2 demonstrates the effectiveness of the proposed optimization method. The mesh before optimization is not very consistent with the real motion, while the mesh after optimization is very consistent with the real motion, which can be seen from the projection results.

V. Retrieval

Similar to [7] but simpler, we use database retrieval to find the best matched candidate frames for video synthesis based on spatiotemporal consistency. In order to compare the query skeletons with the database skeletons, we register them to the same coordinate system by translating the spine-base joints of database skeletons to the same 3D position of the query's joint and rotating the database skeletons to face the same direction. We define the front viewpoint in the database as the reference viewpoint. So given a new query viewpoint, we compute the transformation matrix between the new viewpoint and the reference viewpoint, and transform the query skeleton using this matrix for later retrieval.

Then, we compute an energy for each frame of the data set and choose T (2 in our experiments) frames with lowest energy values as the candidates for video synthesis. Considering the spatiotemporal continuity, the energy function is expressed as

$$\begin{aligned} E(\mathbf{F}) = &\sum_{i\in I(\mathbf{F})} D_s\left(\mathbf{S}_d^i - \mathbf{S}_q^i\right) \\ &+\alpha\left(\sum_{i\in I(\mathbf{F})} D_s\left(\mathbf{S}_d^i - \mathbf{S}_d^{i-1}\right) - \sum_{i\in R} D_s\left(\mathbf{S}_q^i - \mathbf{S}_q^{i-1}\right)\right) \\ &-\beta N(\mathbf{F}) \end{aligned} \tag{4}$$

where $\mathbf{F}$ is the unknown candidate sequence, $N(\mathbf{F})$ is the number of frames in the candidate sequence $\mathbf{F}$, $I(\mathbf{F})$ is the set of frame indices in the original sequence, and R is the number of frames in the query sequence. $\mathbf{S}_d$ is the vector of skeleton joints of a database sequence and $\mathbf{S}_q$ is the vector of skeleton joints of the query sequence. The skeletal distance is defined as

$$D_s(\mathbf{S}_m - \mathbf{S}_n) = \sqrt{\sum_{j=1}^{J} \frac{\left(S_m^j - S_n^j\right)^2}{\sigma_j}} \tag{5}$$

where m and n are d or q, S^j is the 3D position of the jth skeletal joint in the world coordinate system, J is the number of joints, and σ_j is the variance of the position of joint j in the database.

In (4), the first term is a spatial constraint, which ensures the similarity between the skeleton in the candidate sequence and that in the query sequence. The second term is to ensure the similarity of amount of motion between the candidate sequence and the query sequence, and the third term is to make the sequence from the same data set as long as possible. These two terms impose the temporal continuity, avoiding the jitter effect.

VI. Video Synthesis Based on Adaptive Matrix Completion

In this section, we synthesize each target frame based on adaptive weighted low-rank matrix completion. Initial estimations are first obtained by warping the retrieved candidates with the help of 3D models. Then, the frames warped from the candidate that has the lowest energy value are optimized using an adaptive weighted matrix completion method to synthesize a frame of the animation video.

For our problem, no existing image completion methods can be directly applied after initial estimation because the video synthesis in our problem is challenging.

1) A large range of missing data need to be repaired.
2) Many of missing pixels are around the boundary (the junction of the foreground and the background), which is difficult to correctly recover.
3) The texture to be restored may be complex.

Therefore, we propose a video synthesis method based on weighted low-rank matrix completion and elegantly design an adaptive block size scheme to remove the block artifacts around boundary.

A. Initial Estimation

With query skeletons, we can obtain query meshes by deforming the static template mesh using the multipriority inverse kinematics method and DQS skinning method mentioned in Section IV. Based on the assumption that the surface meshes in the database and query are generated by deforming the same static 3D model, i.e., the meshes in the database and query have the same connectivity but different poses, we warp the retrieved database frames (source frames) using the vertex correspondences based on moving least squares [35]. Specifically, we first segment the body of the query mesh into 16 parts according to the maximum skinning weight of each vertex, and hence obtain 16 segments in the target frame via projection. Then, we project the meshes of the database and the mesh of the query onto the source and target frames. Using the vertex correspondences as a guide, moving least squares is adopted to compute the corresponding pixels in the source frames for all the pixels in each body part of the target frame. For the pixels on the boundary of two body parts, we compute a weighted blending of two warping results on the assumption about the attribution of the pixel. In this way, we obtain an image with some missing regions for each retrieved candidate frame.

B. Sparse Texture Reconstruction

The initial estimation generated by Section VI-A may contain missing pixels because not all the needed information is included in the retrieved database frame. In this step, we take the initial estimated image $\mathscr{I}$ from the candidate with the lowest retrieval energy $E(\mathbf{F})$ from (5) as reference and interpolate the missing regions based on the adaptive weighted matrix completion. Fig. 3 shows the procedure of the proposed sparse texture reconstruction (STR) method.

We generate a silhouette image $\mathscr{I}_s$ by projecting the query mesh onto the target frame. Let us denote the known human body region in $\mathscr{I}$ by $\mathscr{F}$, the unknown human body region by $\mathscr{U}$, and the background region by $\mathscr{G}$. For each pixel $\mathbf{x}_i$ at the boundary of the unknown region $\mathscr{U}$, we get a block $\mathscr{B}_i$ with size $m \times m$ centered at $\mathbf{x}_i$ and compute its priority P by

$$P = w_i \frac{\left|\nabla I_i^{\perp} \cdot n_i\right|}{255} \exp\left(-\xi \frac{\sum_{j\in\mathscr{B}_i\cap\mathscr{F}} d_j}{|\mathscr{B}_i \cap \mathscr{F}|}\right). \tag{6}$$

The first term w_i is set at 0 for the pixels around background and 1 otherwise, which is judged by examining the difference in pixel values on $\mathscr{I}_s$ between left and right pixels or top and bottom pixels. This guarantees that the texture recovery always uses the foreground pixels. The second term gives high priority to the pixels on edges, e.g., folds and wrinkles, which guarantees that the recovered texture has rich details. $\nabla I_i^{\perp}$ is a vector orthogonal to the gradient at pixel $\mathbf{x}_i$ but has the same magnitude, and n_i is a unit vector orthogonal to the boundary

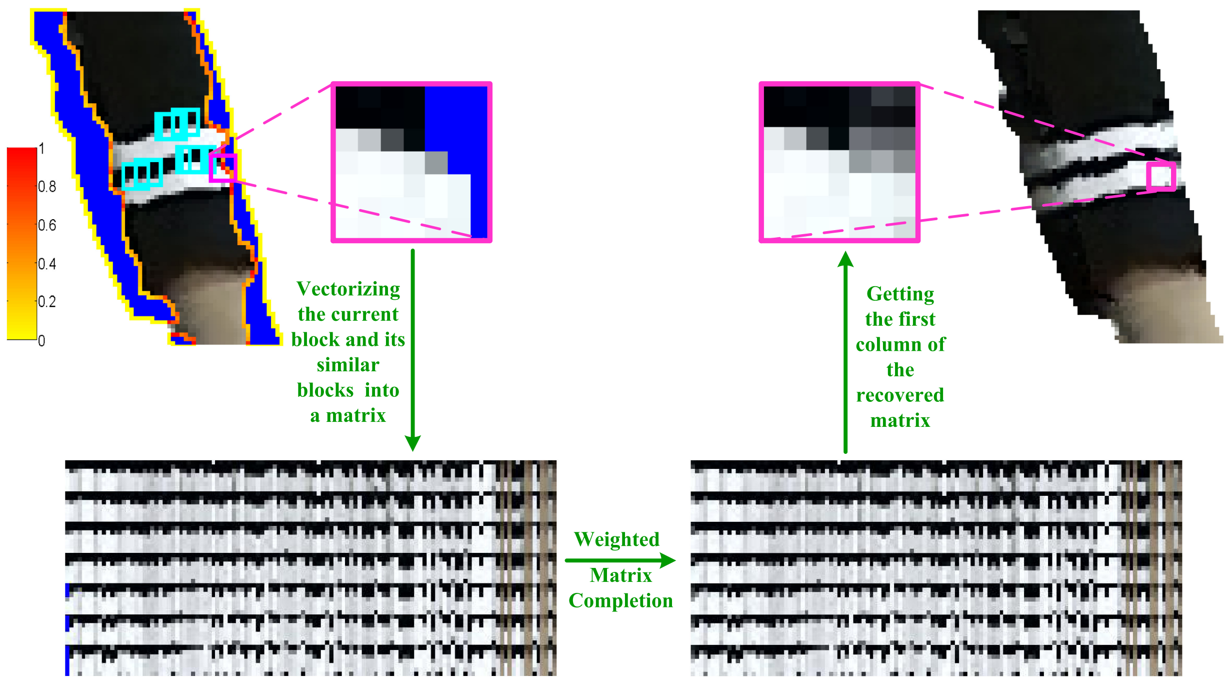

1
0.8
0.6
0.4
0.2
0
Vectorizing the current block and its similar blocks into a matrix
Weighted Matrix Completion
Getting the first column of the recovered matrix

Fig. 3. Illustration of the proposed STR method. The top-left image shows the computed priorities for the pixels at the boundary of the unknown regions. 1 represents the highest priority and 0 is the lowest priority. The block centered at the pixel with the highest priority is illustrated with a pink border and enlarged for closer observation. This block and several searched similar blocks (shown with a cyan border) are arranged into a matrix, and then the matrix is recovered with a weighted low-rank matrix completion method. In this way, we get the reconstructed texture for the current block from the first column of the recovered matrix. The top-right image gives the reconstructed texture for the top-left image.

of the unknown region. The third term considers that the pixels close to the known human body region $\mathscr{F}$ are more reliable than those far away. Instead of searching the nearest pixel in $\mathscr{F}$, we estimate the distance in a greedy manner

$$d_j = \min_{\mathbf{x}_k \in \mathscr{N}_j \cap \mathscr{F}} d_k + ||\mathbf{x}_k - \mathbf{x}_j||_2 \quad (7)$$

where $\mathscr{N}_j$ is the eight-connected neighborhood of $\mathbf{x}_j$. If $\mathbf{x}_j$ lies in the original known region, we set $d_j = 0$. The parameter ξ controls the decay rate of the exponential and is set at 3 in our experiments. Specifically, the parameter is tuned by the bisection method. We found that the optimal parameters for different data sets were quite close, and the performance around the optimal values was stable. Therefore, we use the same parameter setting for all the experiments in our paper.

After determining the priorities for all the pixels at the boundaries, we recover the textures beginning from the pixel with the highest priority. Let us denote $\mathscr{B}_0$ as the block centered at the pixel with the highest priority. We search K similar blocks within a range of $n \times n$ pixels on $\mathscr{I}$ and the same locations on the other candidate images by computing the sum of zero-mean normalized cross-correlation (ZNCC) on the silhouette image $\mathscr{I}_s$ and the RGB channels of $\mathscr{I}$

$$C(\mathscr{B}_0, \mathscr{B}_i) = (ZNCC_r + ZNCC_g + ZNCC_b + ZNCC_s)/4. \quad (8)$$

Suppose the pixel (a, b) in image $\mathscr{I}$ is the center of block $\mathscr{B}_0$ and the pixel $(a+x, b+y)$ is the center of block $\mathscr{B}_i$ with disparity (x, y). The ZNCC between two blocks centered at the corresponding pixels is computed by

$$\frac{\sum_{u,v}\left(R_{u,v} - \overline{R}\right)\left(S_{x+u,y+v} - \overline{S}_{x,y}\right)}{\sqrt{\sum_{u,v}\left(R_{u,v} - \overline{R}\right)^2 \sum_{u,v}\left(S_{x+u,y+v} - \overline{S}_{x,y}\right)^2}} \quad (9)$$

where u and v are pixel indices in the two $(2U+1) \times (2V+1)$ blocks. $R_{u,v}$ and $S_{x+u,y+v}$ represent the value of the pixel $(a+u, b+v)$ in block $\mathscr{B}_0$ and the value of the pixel $(a+x+u, b+y+v)$ in block $\mathscr{B}_i$, respectively. $\overline{R}$ and $\overline{S}_{x,y}$ are the mean values of the $(2U+1) \times (2V+1)$ block. Note that we compute ZNCC using the pixels in the known regions. In our experiments, we set U and V at $(m-1)/2$.

We process the image for each channel of RGB color space separately. Taking one channel for example, we vectorize the current block and all the similar blocks, and stack them into a matrix: $\mathbf{D} := [vec(\mathscr{B}_0), vec(\mathscr{B}_1), \ldots, vec(\mathscr{B}_N)]$. On the one hand, there is a strong correlation between each column of $\mathbf{D}$, and hence, the rank of $\mathbf{D}$ should be very low. On the other hand, the known entries in $\mathbf{D}$ are corrupted by considerable amount of noise and outliers due to nonconsistent texture between different candidate images, self-occlusion, and lighting. Therefore, we use an observation model with missing data and noise: $\mathbf{D} = \mathbf{A} + \mathbf{E}$, where $\mathbf{A}$ is the latent low-rank matrix to be recovered and $\mathbf{E}$ represents the noise and outliers in the known elements that is sparse. So we formulate the problem as

$$\min_{\mathbf{A},\mathbf{E}} \operatorname{rank}(\mathbf{A}) + \lambda||\mathbf{E}||_0 \quad \text{s.t. } P_\Omega(\mathbf{D}) = P_\Omega(\mathbf{A} + \mathbf{E}) \quad (10)$$

where $\|\cdot\|_0$ represents the ℓ_0 norm of the matrix (number of nonzero entries), λ is the weighting factor, $\mathbf{\Omega}$ is the index set of known elements, and P_Ω is a projection operator that projects the matrix onto the domain of $\mathbf{\Omega}$. The optimization problem (10) is extremely difficult (NP-hard in general) to solve. So the rank and the ℓ_0-norm are relaxed into the nuclear-norm (sum of the singular values) and ℓ_1-norm, respectively. There may be some mismatching in the searched similar blocks. Hence, considering different contributions of each similar block, we adopt a weighted matrix completion model

$$\min_{\mathbf{A},\mathbf{E}} ||\mathbf{A}||_* + \lambda||\mathbf{W} \circ \mathbf{E}||_1, \text{ s.t. } P_\Omega(\mathbf{D}) = P_\Omega(\mathbf{A}+\mathbf{E}) \tag{11}$$

where $||\mathbf{A}||_*$ is the nuclear norm of the matrix $\mathbf{A}$, $\circ$ represents element-wise multiplication of two matrices, and $\mathbf{W}$ is a weighting matrix to consider the similarity of searched blocks. Concretely, the weighting matrix is defined as $\mathbf{W} = \mathbf{1} \otimes \mathbf{w}^\top$, where $\mathbf{1} := [1, 1, \ldots, 1]^\top \in \mathbb{R}^{m^2 \times 1}$ is an all-one vector, and $\mathbf{w} := [w_0, w_1, w_2, \ldots, w_K] \in \mathbb{R}^{(K+1)\times 1}$ is a vector containing the similarity of the searched blocks to the current block $\mathscr{B}_0$

$$w_i = C(\mathscr{B}_0, \mathscr{B}_i), \quad i = 0, 1, \ldots, K. \tag{12}$$

In this way, blocks that are more similar to the current block are assigned with larger weights, and hence contribute more to the final restoration results.

We solve the weighted matrix completion problem (11) using the augmented Lagrangian method [36]. The augmented Lagrangian function of minimization (11) is

$$\begin{aligned} L(\mathbf{A}, \mathbf{E}, \mathbf{Y}, \mu) = {} & ||\mathbf{A}||_* + \lambda||\mathbf{W} \circ \mathbf{E}||_1 \\ & + \langle \mathbf{Y}, P_\Omega(\mathbf{D} - \mathbf{A} - \mathbf{E}) \rangle \\ & + \frac{\mu}{2}||P_\Omega(\mathbf{D} - \mathbf{A} - \mathbf{E})||_F^2 \end{aligned} \tag{13}$$

where $\mathbf{Y}$ is the Lagrangian multiplier, $\langle\rangle$ denotes the inner product of two matrices (defined in the same way as the inner product of two vectors), $\|\cdot\|_F$ denotes the matrix Frobenius norm, and $\mu > 0$ is a scalar variable to adjust the consistency of the recovered matrix to the observed values. We minimize the augmented Lagrangian optimization using alternating direction method (ADM) [34]. We take the first column of the recovered matrix $\mathbf{A}$ and reshape it into an $m \times m$ block as the recovered block of $\mathscr{B}_0$.

Adaptive Block Size: The reconstruction algorithm processes the missing pixels progressively from interior pixels toward the boundary. Then, the anchor block will step across the boundary of the human body and contain background pixels, which would affect the search of similar blocks. We adaptively reduce the block size to handle this issue. The size of the anchor block is recursively reduced until it does not contain background pixels. Then, missing pixels of the anchor blocks with reduced size are reconstructed in the same way as normal blocks. If the size of the anchor block is reduced to one, it is simply filled by the average of available pixels in its four-connected neighborhood.

Since we synthesize the textures frame by frame, some jitter artifacts may occur when the recovered textures are temporally inconsistent. This can be easily improved by any temporal blending technique based on optical flow [7], [37].

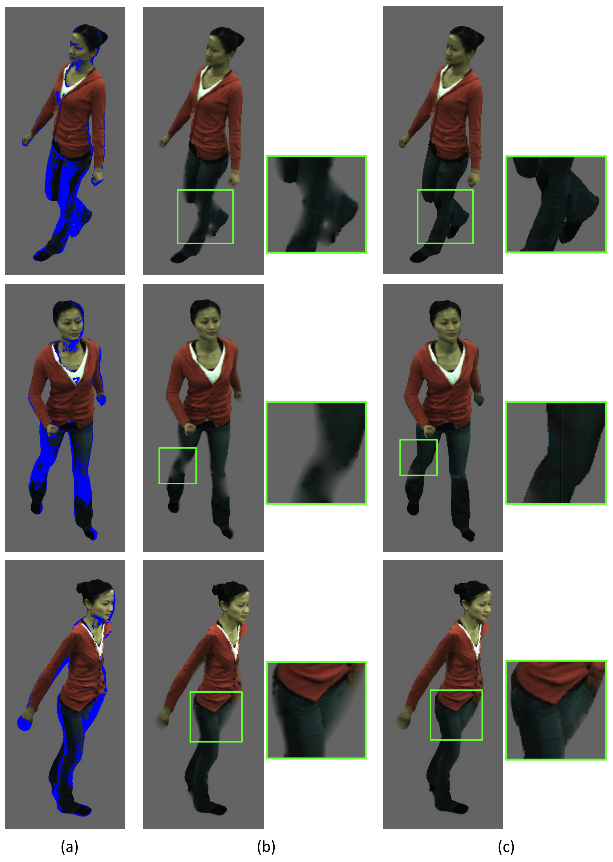


Fig. 4. Reconstructed textures for (a) image with missing data using (b) MVC method and (c) our proposed method. Top to bottom: *walking*, *punching*, and *turning* motions. In order to compare with the MVC method, we use the same background for synthesizing the new video.

VII. Results

In this section, we evaluate the performances of the proposed method on a public data set (Section VII-A) and a real Kinect data set (Section VII-B), and some discussions are given in Section VII-C. We set the parameters as follows: block width $m = 11$, range width $n = 100$, and number of similar blocks $K = 30$.

A. Evaluation on Public Data Set

We generate a data set for evaluation using the multiview data set of the multiview video-based character (MVC) method [7] by keeping only one view for each motion. In order to compare with the method in [7], we use the same retrieval method and the background for synthesizing the new video. Fig. 4 gives the comparison results for *walking*, *punching*, and *turning* motions. The regions highlighted by rectangles are enlarged and shown in the associated images for closer observation. It can be seen that the results of the MVC method have obvious blurring artifacts on the legs, while our method is free of this problem and provides significantly better results than MVC method. This appealing property is attributed to the proposed adaptive weighted sparse reconstruction model (11). The constraint reflects the fidelity of the reconstructed matrix and ensures the accuracy of the reconstruction.

The objective takes the low-rankness and sparsity into account and ensures the smoothness of the reconstruction. By bridging the two terms with a penalization parameter and then minimizing it, not only are the unknown elements recovered, but the noise and outliers in the known elements are also reduced. Adaptive block-size selection ensures that the recovery is implemented on the necessary resolution: large enough to consider the global consistency and small enough to recover the local details.

B. Evaluation on Kinect Data Sets

Our method is further evaluated on the data sets captured with a Kinect RGB-D camera. We capture an actor (actress) with normal clothing performing nine basic motions using a Kinect v2.0 camera at 30 frames/s with a color image resolution of 1920×1080 pixels and a depth image resolution of 512×424 pixels. We calibrate the color camera and the depth camera using OpenCV (Open Source Computer Vision) library. All the data sets will be made publicly available at a project webpage.

1) Sparse Texture Reconstruction: We first assess the performance of the STR component (Section VI-B) that plays an important role in synthesizing photorealistic videos. In Fig. 5, we compare the STR method with three other methods: 1) exemplar-based inpainting method (EBIM) [38]; 2) the MVC method [7]; and 3) the STR method without adaptive block sizes (STRwoABS). In EBIM [38], the missing pixels of the current block are filled by the counterparts of the most similar block. EBIM considers only texture information in determining the order of inpainting, which is not applicable to our case with silhouette constraints. For fair comparison, we use the same priority calculation scheme as the proposed method. As shown in Fig. 5(a), the results generated by the EBIM method present obviously wrong textures around the silhouette of the human body. The reason for the artifacts is that the straightforward copy of similar blocks may introduce some wrong pixels, which are further propagated by the recursive inpainting procedure. The result of the MVC method is blurred. On the contrary, the proposed STR method is able to reconstruct correct textures thanks to the powerful recovery capability of the weighted low-rank matrix completion from incomplete observations. The result in Fig. 5(c) contains severe blocking artifacts around the boundaries of the human body while that in Fig. 5(d) is free of this issue, which demonstrates the effectiveness of the proposed block-size adaptation scheme.

The proposed STR method is able to challenge some extreme cases with large missing regions. For demonstration, we synthesize video frames for a given query *bow-pulling* motion only from a single candidate image (the leftmost image in Fig. 6). The poses of the actress can significantly deviate that in the single available image, e.g., the right arm raises and blends in the bow-pulling motion. In such cases, the initially estimated images would present large missing areas, which are quite challenging to reconstruct as only a small fraction of pixels are available. Fig. 6 shows ten frames of the synthesized video for the *bow-pulling* motion

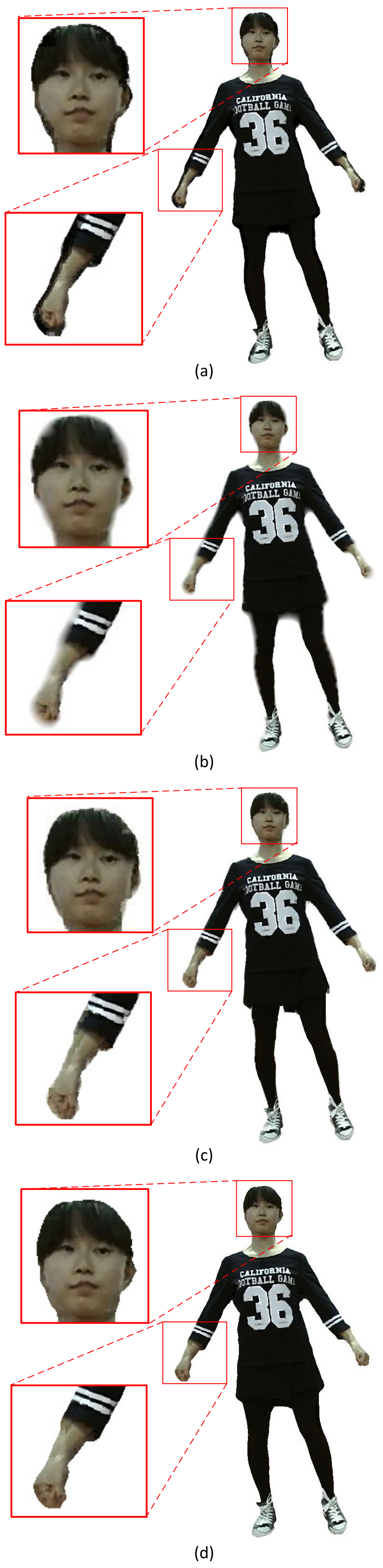


Fig. 5. Comparison of reconstructed textures from an image with missing pixels by (a) EBIM, (b) MVC, (c) STRwoABS, and (d) proposed STR method.

at different poses. The results show that the STR method is able to reconstruct visually appealing results from even one single candidate image.

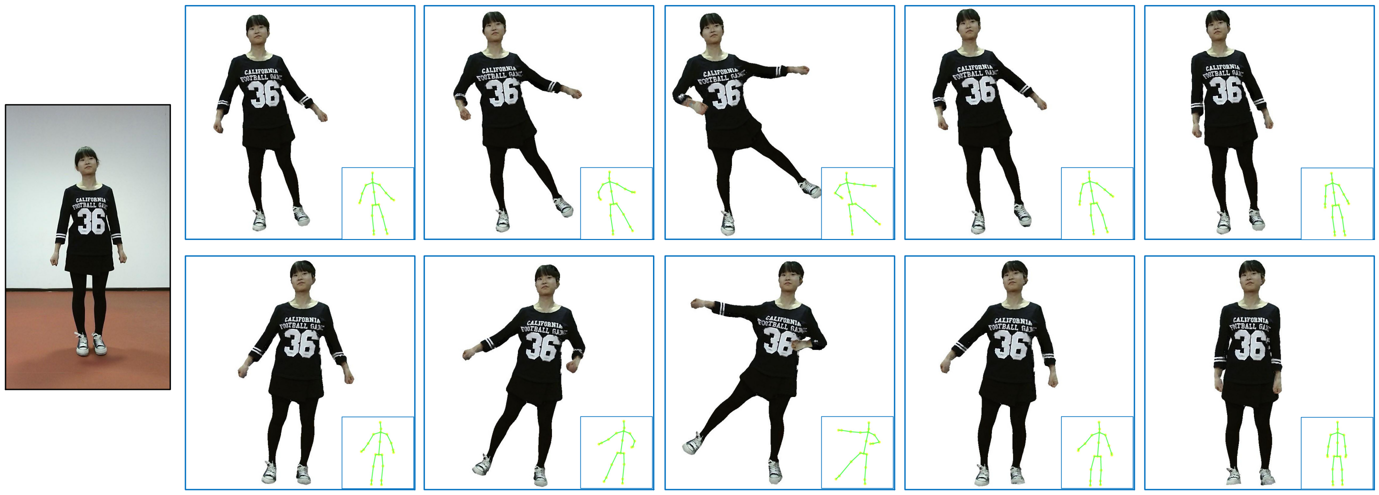


Fig. 6. Ten frames of our created video for *bow-pulling* motion. Only one frame is used as the data set. The inlays show the associated query skeleton used to create the target frame.

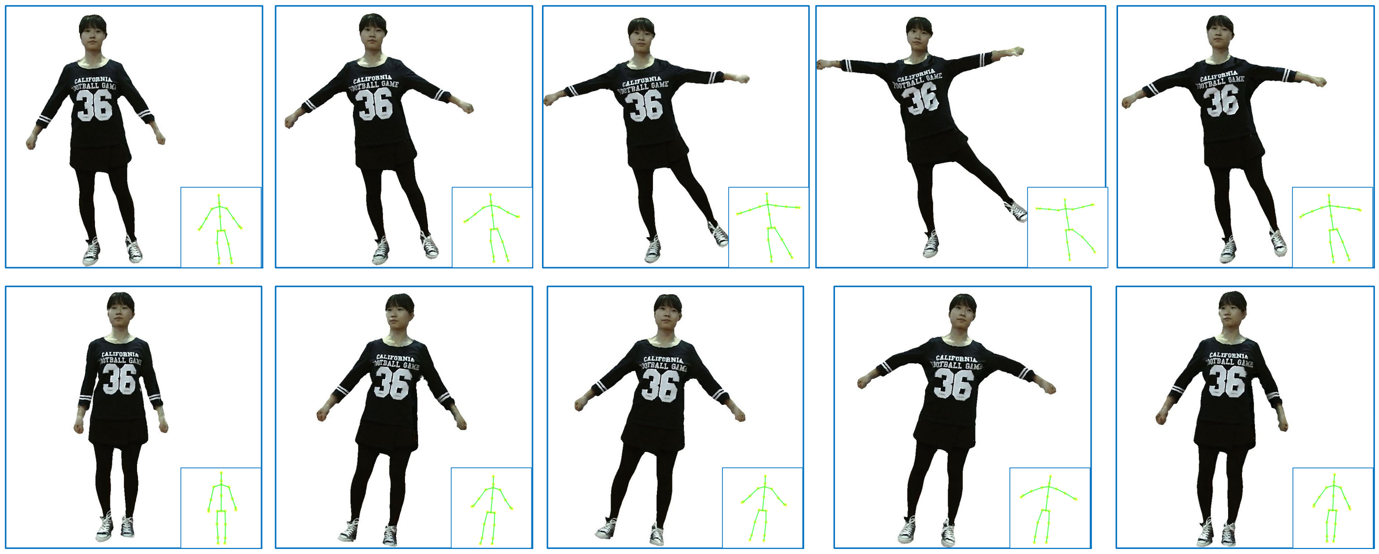


Fig. 7. Ten frames of our created video for *lateral raising* motion. The inlays show the associated query skeleton used to create the target frame.

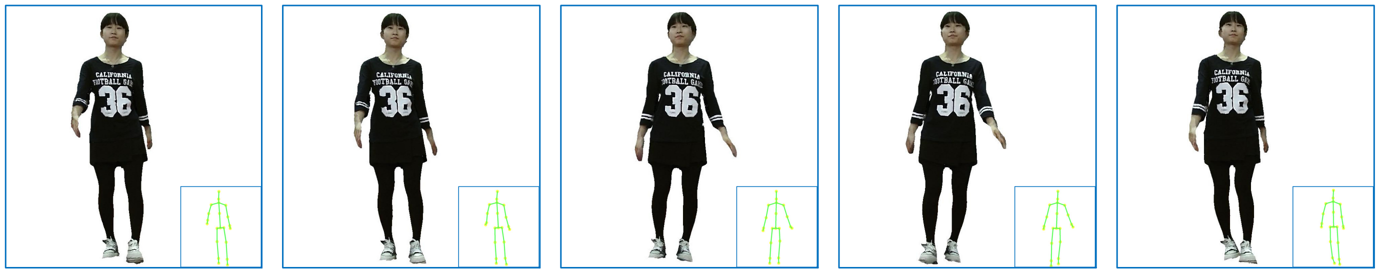


Fig. 8. Five frames of our created video for the actor without clenched fists. The inlays show the associated query skeleton used to create the target frame.

2) Sparse Photorealistic Animation: Fig. 7 shows ten frames of the synthesized video for the query *lateral raising* motion. The results show that the poses of the actress in the synthesized frames are consistent with the poses of the query skeletons. The frames are reliably reconstructed according to the new motion, which is demonstrated by the completeness of the reconstructed textures, e.g., the logo texture on the T-shirt, two white stripes on the sleeves, and the laces on the shoes.

Fig. 8 presents five reconstructed frames for the query *walking* motion. In the candidate videos of the data sets, the actress does not clench her fists, which is different from data

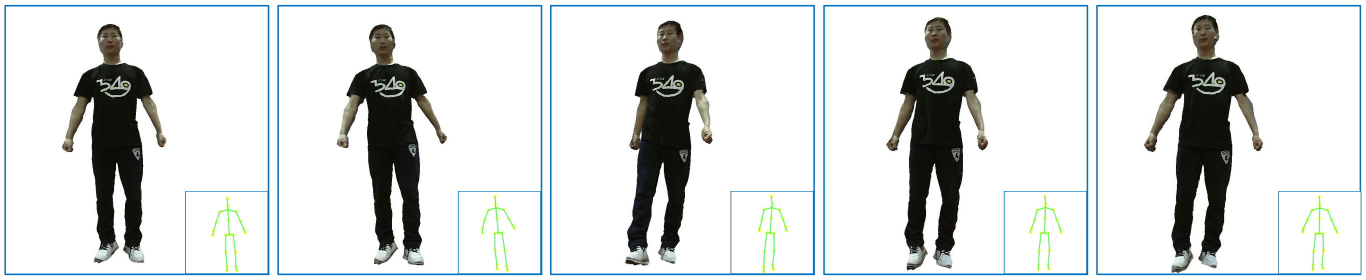


Fig. 9. Five frames of our created video for another actor at a new viewpoint. The inlays show the associated query skeleton used to create the target frame.

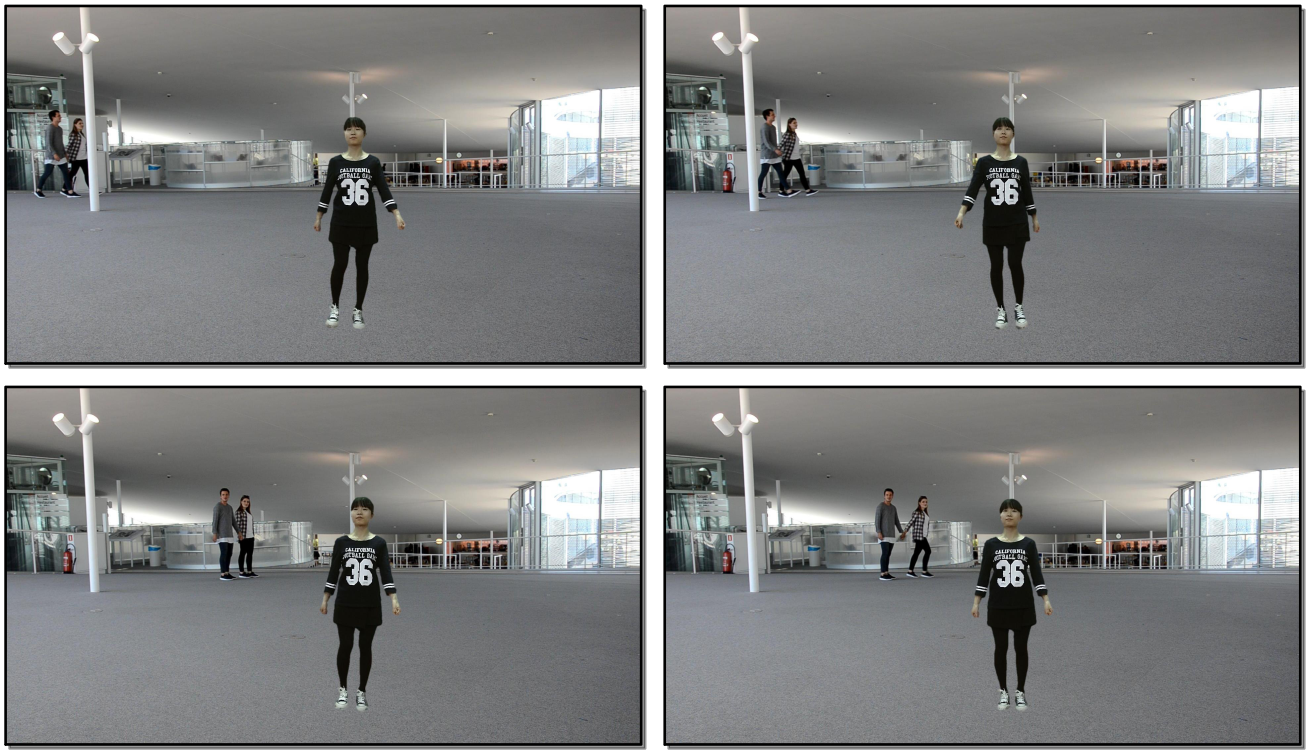


Fig. 10. Animation results of an actor that are created with our method from a small database captured by a single RGB-D camera. The motion is designed by an animator and the background is captured by a commercial camera. Relighting techniques are not used.

sets in most previous photorealistic animation work, and it is difficult to reconstruct the hands. As observed in Fig. 8, our method is able to provide acceptable results although it is difficult to reconstruct the fine details of the hands with flexible motions.

Fig. 9 shows the results of another actor for the *robotic walking* motion at a new viewpoint. The synthesized frames are of the same quality as the results for other data sets, although some small white holes caused by the segmentation error can be seen on the right leg of the actor in the last image. This suggests that our method has stable performance for various actors and query motions.

Fig. 10 shows a composite of our synthesized video into a real scene. Four frames of the created video are given. The motion is designed by an animator and the background is captured by a commercial camera. Relighting techniques are not used. The results demonstrate that it is feasible to insert our video-based animations with new motions into real captured videos.

C. Discussion

Experimental results show that our method achieves plausible animation videos. The visual quality depends on the quality of the database and the similarity between new motion and database motions. Visual artifacts in either segmentation or performance capture will degrade the quality of video synthesis for new motion. Some segmentation errors will bring background color into the foreground region when reconstructing the textures, while errors in performance capture will also affect the accuracy of model-guided image deformation. The reasons for artifacts are as follows.

1) The segmentation of depth is not accurate enough without the use of chroma-key backgrounds. This is the main reason, because the depth boundary directly affects the boundary of the synthesized frame (please see Sections IV-B and VI-A). We use a common segmentation method [39] for Kinect to segment the depth. Even using manual segmentation, it is difficult to ensure the boundary consistency for all the motions

due to the depth accuracy of Kinect, especially for the hair.

2) For database setup, we did not use a special design for uniform lighting. Instead, we capture the data in the normal official environment with several fluorescent lights on the ceiling, which leads to some highlights and inconsistent color on the actor, particularly on the face.

The above analysis can be verified from the experiment using the data set from [7], in which there is no such artifact for our method because the data set in [7] is captured with a chroma-key background and uniform lighting. If a more accurate depth camera is used, the reconstructed mesh and the recovered texture will be better. However, we did not involve post-processing such as temporally blending and did not use an expensive depth camera, because we want to demonstrate the possibility of achieving animation videos in a simple way of using a single cheap RGB-D camera and capturing the database in a usual office environment.

In our experiments, we only use nine data sets and usually only three data sets are used for video synthesis after retrieval. If the new query motion is very different from the database motions, adding more data sets with similar kind of motions might be helpful.

The main limitation of our approach is the computational complexity of the STR method. We run the algorithm on a laptop with an Intel(R) Core(TM) i5-4200M 2.5-GHz CPU and 4.0-GB RAM. The running time is about 10 min/frame, which is not suitable for real-time applications.

Our method does not include color correction and relighting. We try to minimize the color difference of different data sets by using uniform lighting when capturing the data set images. In future work, multisource co-color-correction methods can be used to adjust the colors of different database images to be consistent, and intrinsic image decomposition methods can be adopted together with relighting techniques to increase the realism of the generated video [40].

VIII. Conclusion

In this paper, we present a new photorealistic animation method using a single RGB-D camera. Based on a small database with some basic motions, we create a plausible video of an actor performing a new motion. We obtain more accurate deformed meshes for marker-less performance capture based on sparse representation. We also propose a sparse reconstruction method for textures using adaptive weighted low-rank matrix completion, which is less sensitive to noise and outliers. Our system is cheap and nonintrusive. We demonstrate the effectiveness of the proposed method on a public data set and our captured data sets. The potential of our method is verified on an extreme case in Fig. 6: all frames of the new motion were synthesized from the same frame while the visual quality is still quite good. Our method achieves compelling animations, despite the poor quality of the input database.

Acknowledgment

The authors would like to thank Dr. F. Xu for his helpful discussions and the distribution of data sets.

References

[1] W. Gao, Y. Chen, R. Wang, S. Shan, and D. Jiang, “Learning and synthesizing MPEG-4 compatible 3-D face animation from video sequence,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 11, pp. 1119–1128, Nov. 2003.

[2] Y. Yemez *et al.*, “Scene representation technologies for 3DTV—A survey,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1587–1605, Nov. 2007.

[3] A. Hilsmann, P. Fechteler, and P. Eisert, “Pose space image based rendering,” *Comput. Graph. Forum*, vol. 32, no. 2, pp. 265–274, 2013.

[4] A. Schödl, R. Szeliski, D. H. Salesin, and I. Essa, “Video textures,” in *Proc. 27th Annu. Conf. Comput. Graph. Interact. Techn.*, 2000, pp. 489–498.

[5] M. Flagg *et al.*, “Human video textures,” in *Proc. Symp. Interact. 3D Graph. Games*, 2009, pp. 199–206.

[6] D. Casas, M. Volino, J. Collomosse, and A. Hilton, “4D video textures for interactive character appearance,” *Comput. Graph. Forum*, vol. 33, no. 2, pp. 371–380, 2014.

[7] F. Xu *et al.*, “Video-based characters: Creating new human performances from a multi-view video database,” *ACM Trans. Graph.*, vol. 30, no. 4, p. 32, Jul. 2011.

[8] H. Avron, A. Sharf, C. Greif, and D. Cohen-Or, “ℓ_1-sparse reconstruction of sharp point set surfaces,” *ACM Trans. Graph.*, vol. 29, no. 5, p. 135, Oct. 2010.

[9] R. Preiner, O. Mattausch, M. Arikan, R. Pajarola, and M. Wimmer, “Continuous projection for fast ℓ_1 reconstruction,” *ACM Trans. Graph.*, vol. 33, no. 4, p. 47, Jul. 2014.

[10] R. Wang, Z. Yang, L. Liu, J. Deng, and F. Chen, “Decoupling noise and features via weighted ℓ_1-analysis compressed sensing,” *ACM Trans. Graph.*, vol. 33, no. 2, p. 18, Mar. 2014.

[11] S. Lu, X. Ren, and F. Liu, “Depth enhancement via low-rank matrix completion,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 3390–3397.

[12] J. P. Lewis, M. Cordner, and N. Fong, “Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation,” in *Proc. 27th Annu. Conf. ACM SIGGRAPH*, 2000, pp. 165–172.

[13] X. C. Wang and C. Phillips, “Multi-weight enveloping: Least-squares approximation techniques for skin animation,” in *Proc. ACM SIGGRAPH/Eurograph. Symp. Comput. Animation*, 2002, pp. 129–138.

[14] B. Merry, P. Marais, and J. Gain, “Animation space: A truly linear framework for character animation,” *ACM Trans. Graph.*, vol. 25, no. 4, pp. 1400–1423, 2006.

[15] A. Mohr and M. Gleicher, “Building efficient, accurate character skins from examples,” *ACM Trans. Graph.*, vol. 22, no. 3, pp. 562–568, 2003.

[16] L. Kavan, S. Collins, J. Žára, and C. O’Sullivan, “Geometric skinning with approximate dual quaternion blending,” *ACM Trans. Graph.*, vol. 27, no. 4, 2008, Art. no. 105.

[17] E. de Aguiar, L. Sigal, A. Treuille, and J. K. Hodgins, “Stable spaces for real-time clothing,” *ACM Trans. Graph.*, vol. 29, no. 4, 2010, Art. no. 106.

[18] H. Wang, F. Hecht, R. Ramamoorthi, and J. F. O’Brien, “Example-based wrinkle synthesis for clothing animation,” *ACM Trans. Graph.*, vol. 29, no. 4, 2010, Art. no. 107.

[19] A. Schödl and I. A. Essa, “Controlled animation of video sprites,” in *Proc. ACM SIGGRAPH/Eurograph. Symp. Comput. Animation*, 2002, pp. 121–127.

[20] B. Celly and V. B. Zordan, “Animated people textures,” in *Proc. 17th Int. Conf. Comput. Animation Soc. Agents (CASA)*, 2004, pp. 331–338.

[21] J. Starck, G. Miller, and A. Hilton, “Video-based character animation,” in *Proc. ACM SIGGRAPH/Eurogr. Symp. Comput. Animation*, 2005, pp. 49–58.

[22] P. Huang, A. Hilton, and J. Starck, “Human motion synthesis from 3D video,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1478–1485.

[23] H. Li, E. Vouga, A. Gudym, L. Luo, J. T. Barron, and G. Gusev, “3D self-portraits,” *ACM Trans. Graph.*, vol. 32, no. 6, 2013, Art. no. 187.

[24] X. Wei, P. Zhang, and J. Chai, “Accurate realtime full-body motion capture using a single depth camera,” *ACM Trans. Graph.*, vol. 31, no. 6, 2012, Art. no. 188.

[25] M. Hornácek, C. Rhemann, M. Gelautz, and C. Rother, "Depth super resolution by rigid body self-similarity in 3D," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2013, pp. 1123–1130.

[26] P. E. Debevec, C. J. Taylor, and J. Malik, "Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach," in *Proc. SIGGRAPH*, 1996, pp. 11–20.

[27] J. Carranza, C. Theobalt, M. A. Magnor, and H.-P. Seidel, "Free-viewpoint video of human actors," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 569–577, 2003.

[28] R. A. Newcombe *et al.*, "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. 10th IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*, Oct. 2011, pp. 127–136.

[29] I. Baran and J. Popović, "Automatic rigging and animation of 3D characters," *ACM Trans. Graph.*, vol. 26, no. 3, 2007, Art. no. 72.

[30] M. Gleicher, "Retargetting motion to new characters," in *Proc. SIGGRAPH*, 1998, pp. 33–42.

[31] J. Shotton *et al.*, "Real-time human pose recognition in parts from single depth images," *Commun. ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[32] P. Baerlocher and R. Boulic, "An inverse kinematics architecture enforcing an arbitrary number of strict priority levels," *Vis. Comput.*, vol. 20, no. 6, pp. 402–417, 2004.

[33] J. Yang, K. Li, K. Li, and Y.-K. Lai, "Sparse non-rigid registration of 3D shapes," *Comput. Graph. Forum*, vol. 34, no. 5, pp. 89–99, 2015.

[34] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[35] S. Schaefer, T. McPhail, and J. Warren, "Image deformation using moving least squares," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 533–540, 2006.

[36] Z. Lin, M. Chen, and Y. Ma. (2010). "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices." [Online]. Available: http://arxiv.org/abs/1009.5055

[37] H.-C. Chang, S.-H. Lai, and K.-R. Lu, "A robust and efficient video stabilization algorithm," in *Proc. IEEE Int. Conf. Multimedia Expo*, vol. 1. Jun. 2004, pp. 29–32.

[38] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, Sep. 2004.

[39] K. Guo, F. Xu, Y. Wang, Y. Liu, and Q. Dai, "Robust non-rigid motion tracking and surface reconstruction using L_0 regularization," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 3083–3091.

[40] J. Imber, J.-Y. Guillemaut, and A. Hilton, "Intrinsic textures for relightable free-viewpoint video," in *European Conf. Computer Vision*. Springer, 2014, pp. 392–407.

Kun Li received the B.E. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2006, and the master's and Ph.D. degrees from Tsinghua University, Beijing, in 2011.

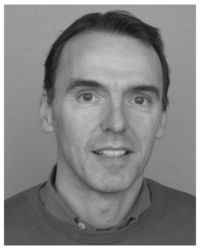
She visited École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, in 2012 and from 2014 to 2015. She is currently an Associate Professor with the School of Computer Science and Technology, Tianjin University, Tianjin, China. Her research interests include dynamic scene 3D reconstruction and image/video processing.

Jingyu Yang received the B.E. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2003 and the Ph.D. (Hons.) degree from Tsinghua University, Beijing, in 2009.

He has been a Faculty Member with Tianjin University, Tianjin, China, since 2009, where he is currently an Associate Professor with the School of Electronic Information Engineering. He was with Microsoft Research Asia (MSRA), Beijing, in 2011, within the MSRA's Young Scholar Supporting Program, and the Signal Processing Laboratory, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, in 2012, and from 2014 to 2015. His research interests include image/video processing, 3D imaging, and computer vision.

Leijie Liu received the B.E. degree from Tianjin University, Tianjin, China, in 2013, where he is currently working toward the M.E. degree with the School of Electronic Information Engineering.

His research interests include image/video processing.



Ronan Boulic (SM'05) received the Ph.D. degree in computer science from University of Rennes, Rennes, France, in 1986 and the Habilitation degree in computer science from University of Grenoble, Grenoble, France, in 1995.

He is currently a Senior Scientist with École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland. He leads the Immersive Interaction Research Group. He has co-authored over 140 refereed publications, including 39 in ISI-indexed journals, and contributed to ten books.

Dr. Boulic is a Senior Member of the Association for Computing Machinery and a Member of Eurographics. He has served on over 50 program committees of key conferences in computer graphics, computer animation, and virtual Reality.

Yu-Kun Lai received the bachelor's and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 2003 and 2008, respectively.

He is a Lecturer of visual computing with the School of Computer Science and Informatics, Cardiff University, Cardiff, U.K. His research interests include computer graphics, geometry processing, image processing, and computer vision.

Dr. Lai is on the editorial board of *The Visual Computer*.

Yebin Liu received the B.E. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2002 and the Ph.D. degree from the Automation Department, Tsinghua University, Beijing, in 2009.

He was a Research Fellow with the Computer Graphics Group, Max Planck Institute for Informatik, Saarbrücken, Germany, in 2010. He is currently an Associate Professor with Tsinghua University. His research interests include computer vision and computer graphics.

Yubin Li received the B.E. degree from Tianjin University, Tianjin, China, in 2015. He is currently working toward the M.Sc. degree in computer science with Memorial University of Newfoundland, St. John's, NL, Canada.

His research interests include image/video processing, video games, and bioinformatic.

Eray Molla received the B.Sc. degree in computer engineering from Middle East Technical University, Ankara, Turkey, and the master degree in computer science, together with a minor degree program in management of technology and entrepreneurship, and the Ph.D. degree from École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, in 2016.

He participated in research activities on augmented reality during his master's studies. Moreover, he had a six-month internship with Logitech, Romanel-sur-Morges, Switzerland, where he was involved in virtual reality and computer vision.