Markerless Shape and Motion Capture from Multi-view Video Sequences

Kun Li, Qionghai Dai, *Senior Member, IEEE,* and Wenli Xu

***Abstract*—We propose a new markerless shape and motion capture approach from multi-view video sequences. The shape recovery method consists of two steps: separating and merging. In the separating step, the depth map represented with a point cloud for each view is generated by solving a proposed variational model, which is regularized by four constraints to ensure the accuracy and completeness of the reconstruction. Then, in the merging step, the point clouds of all the views are merged together and reconstructed into a 3D mesh using a marching cubes method with silhouette constraints. Experiments show that the geometric details are faithfully preserved in each estimated depth map. The 3D meshes reconstructed from the estimated depth maps are watertight and present rich geometric details, even for non-convex objects. Taking the reconstructed 3D mesh as the underlying scene representation, a volumetric deformation method with a new positional-constraint computation scheme is proposed to automatically capture motions of 3D objects, especially human performances. Our method can capture non-rigid motions even for loosely dressed humans without the aid of markers.**

***Index Terms*—Shape recovery, depth map, 3D mesh, deformation, motion capture.**

I. Introduction

CAPTURING shapes and motions of 3D objects (especially human bodies) from multi-view video sequences is a classic and hot issue in computer vision and graphics [1]–[3]. Some methods jointly estimate the shapes and motions of 3D scenes [4], [5]. However, in general, decoupling the two problems are more efficient. This involves two key techniques: shape recovery and motion capture. Shape recovery refers to reconstructing the static model of a 3D object while motion capture refers to tracking the motion of the 3D object with the help of the reconstructed model. These two techniques have great application potentials in virtual reality and future movie industry. Therefore, high quality recovery of dynamic scenes is highly demanded.

Reconstructing a static 3D object by a laser range scanner is a straightforward approach, but is quite time consuming and requires expensive special hardware. Besides, the object has to be kept actionless during the scanning. As an alternative, recovering a 3D shape from a set of photographic images [6]–[8] has attracted great interests from both the academic community and the industrial community. For this category, the input is an array of multi-view images. The main difficulty for this avenue is to simultaneously ensure both the completeness and accuracy of reconstructed models.

Shape recovery methods are mainly designed for the reconstruction of static scenes. For dynamic scenes, one intuitive method is to reconstruct an individual mesh for each time instant. However, the mesh topology and connectivity change over time. Although spherical parameterization and remeshing [9] can transform the unstructured representation to a single consistent mesh structure, the implementation is complicated and not robust. Moreover, high time and storage complexities are still problematic for practical applications. It is much more efficient to modify only the vertex positions of a 3D mesh according to captured motions while preserving the connectivity of the mesh. A variety of motion capture approaches have been proposed in literature, and can be categorized into marker-based ones and markerless ones. Marker-based motion capture systems are widely used in the game/movie design and production for measuring the motions of real performers [10]. Though the marker-based methods are accurate, they often require performers to wear skin-tight garments with reflective markers, which makes it difficult to capture shapes and textures. Moreover, they also demand extra manual adjustments to clean up the recorded data. To overcome the restrictions of marker-based approaches, markerless motion capture methods are brought into the literature. Although markerless methods are more flexible and some have shown amazing results in the error range of marker-based methods [11]–[13], most of them employ a kinematic skeleton (or kinematic chains) to help capture the motions. A kinematic skeleton only allows the tracking of rigid motions, and hence they have to be combined with other scanning technologies to capture the time-varying shapes of the object. Moreover, all these approaches start the tracking process with a laser scanner, and mesh registration technologies are also needed. Therefore, the animated models generated by these methods generally have two clenched fists, which would not change over time. The wrinkles captured by the scanner also remain the same over time, not even consistent with the first time instant, which looks unrealistic.

In this paper, we propose a new markerless shape and motion capture approach, which does not rely on a laser scanner. For the shape recovery, a variational method is proposed to generate a watertight and accurate 3D model, which owns the main advantages of both shape from silhouette and shape

This work was supported by the National Basic Research Project of China (973 Program, Grant 2010CB731800), the Key Program of NSFC (Grant 60932007), the Science Fund for Creative Research Groups (Grant 60721003), the National High Technology Research and Development Program of China (863 Program, Grant 2009AA01Z327), and the Joint Research Fund for Overseas Chinese Scholars of NSFC (Grant 60828002).

K. Li, Q. Dai, and W. Xu are with TNList; Department of Automation, Tsinghua University, Beijing 10084, China. E-mail: (kunli06@mails.tsinghua.edu.cn; qhdai@tsinghua.edu.cn; xuwl@tsinghua.edu.cn).

from stereo methods. Then, taking the reconstructed model as underlying scene representation, a volumetric deformation method is used to achieve a photo-realistic display of the captured performance. More concretely, multi-view videos are first captured using a multi-camera dome system. Second, the object is automatically segmented from the background for all the images and an initial shape of the object is obtained by the EPVH (Exact Polyhedral Visual Hulls) method [14] for the first time instant. Third, for each view, a depth map represented with a point cloud is reconstructed by optimizing the visible points of the initial visual hull with the proposed variational model. Fourth, the point clouds of all the views are merged together to generate a complete 3D mesh using a marching cubes method [15] with silhouette constraints. Finally, an iterative volumetric Laplacian deformation framework with a new positional-constraint computation scheme is used to animate the model according to the motions in multi-view videos. The proposed shape recovery and motion capture method is evaluated on two publicly available datasets and seven datasets captured by a real multi-camera system. Experiments show that accurate and watertight 3D meshes are obtained and realistic time-varying shapes are captured over time. Some multi-view video datasets can be downloaded from the first author's homepage[1]. This shape and motion capture method is automatic, easy to implement, and can handle loosely dressed humans.

The main contributions of this paper are summarized into the following three aspects:

- A complete automatic shape and motion capture system. Unlike most previous methods, our method does not rely on a 3D scanner or optical markers. Instead, we use the high-quality 3D model reconstructed by our proposed shape recovery method as the underlying scene representation and deform it to follow the motions recorded in the multi-view video sequences.
- A variational method towards shape recovery. Bridging with a penalization factor, both the accuracy and completeness of reconstructions are jointly formulated in the proposed variational model. The generated depth maps present rich geometric details without loss of smoothness. As a result, a watertight 3D mesh is obtained by a simple and fast merging method rather than some existing complicated methods.
- A volumetric deformation method. Positional constraint candidates are first calculated by a new method based on sparse representation. The most reasonable ones are chosen with three spatio-temporal selection criteria, and drive an iterative volumetric Laplacian deformation framework to obtain realistic time-varying shapes.

II. Related Work

This section provides a brief review of related work in the fields of shape recovery and motion capture.

[1]http://media.au.tsinghua.edu.cn/likun.jsp

A. Shape Recovery

The most straightforward way for shape recovery is to use a laser range scanner. However, this approach requires expensive devices and time-consuming postprocessing. Besides, the object has to keep still during the scanning, which severely restricts its applicable scenarios. As an alternative, recovering 3D shape from real images has attracted great attentions from both the academic and industrial fields. A family of such techniques is called *shape from X*, where X can be shading, silhouette, and stereo, etc.

In *shape from shading* methods, the shape of a 3D surface is computed from a gradual variation of shading in a single grayscale image [16]. Most recovered shapes are not a complete surface of the 3D object, which is referred to as a partial 2.5D reconstruction. Jin et al. [17] extend the classical shape from shading to the case of multiple views under unknown illumination. The main restriction of these methods is that they assume the Lambertian model of image formation, which is not valid for some practical imaging scenarios.

In *shape from silhouette* methods, the silhouettes of a 3D object is extracted by simple differencing or blue screen segmentation techniques from the captured multi-view images. An approximate 3D model, i.e., visual hull [14], [18], is obtained from the computed silhouettes. The shape from silhouette methods are straightforward to implement, and would be a good choice when crude models are acceptable. Therefore, it has been widely used in many realtime systems for generating and replaying 3D digital videos [19] as well as commercial object modeling packages [20]. However, methods of this category suffer from severe distortion when computing the shape of non-convex objects. The reason is that the concave geometry of the object results in self-occlusion, which cannot be resolved from any viewpoint unless additional information is provided.

In *shape from stereo* methods, correspondences among multi-view images are computed, and Delaunay triangulation is used to reconstruct the objects. The representation of a 3D object can be a set of depth maps [21]–[23], a relief surface [24], [25], or a whole model [6]–[8]. The correspondences between adjacent views can be computed by correlation-based or feature-based methods. The correlation-based algorithms typically produce dense, but a little over-smoothed, measurements of depth. For the feature-based approaches, the obtained 3D points in depth maps are sparse and nonuniform, only around the detected feature points. The shape from stereo methods are simple and efficient. However, the reconstructed surfaces are view-dependent and present artifacts, such as over-smoothness or holes. Moreover, a main limitation is that they can only represent depth maps with a unique disparity for each pixel. More accurate 3D shape can be recovered from images taken under different lighting conditions by photometric stereo [26], [27]. However, as most shape from stereo methods, the photometric stereo methods can recover a shape for only a fixed viewpoint, which produces a partial 2.5D reconstruction.

To get complete 3D models, multi-view stereo algorithms are proposed. The state-of-the-art multi-view stereo methods

employ powerful nonlinear energy minimization techniques, e.g., graph cuts [28], [29], level set evolution [30], and mesh evolution [31]. Most global minimization methods are proposed in a discrete formulation, which introduces some reconstruction artifacts and entails considerable memory. Kolev et al. [32], [33] develop a continuous convex relaxation scheme for global optimization, which is free of discretization artifacts and also reduces the memory requirements. They transform the binary non-convex minimization problems into convex minimization problems and globally solve the problem by means of variational techniques. However, the computational complexity of this method is a little high, taking several hours to reconstruct a 3D model from 16 multi-view images. Moreover, these global methods emphasize the completeness of reconstructed models, and fill the uncertain regions by regularizations. This would lead to over-smoothness in the reconstructed geometries. Besides, these methods often need to handle visibility conditions and silhouette constraints carefully, and their convergence properties in the presence of noise are not well understood. To reconstruct more complex geometries, Goesele et al. [8] propose a two-step algorithm, which is a departure from most modern multi-view stereo methods. In this algorithm, the depth map for each view is calculated by a robust window matching method with a small number of neighboring views, similar to the method proposed in Ref. [34]. Each 3D point must be seen in at least three views to be reconstructed. Then, the resulting depth maps are merged using a volumetric method proposed by Curless and Levoy [35]. The estimated depth maps by this algorithm are more accurate than previous methods. However, since only a simple NCC (Normalized Cross-Correlation)-based photo-consistency measure is used, the individual depth map for each view may contain numerous holes around uncertain areas such as occlusions, highlights, and low-textured regions. As a result, the reconstructed models present some holes of various sizes.

To reconstruct complete 3D models while preserving detailed geometries, this paper proposes a new shape recovery method using a separating and merging strategy, which possesses the main advantages of both shape from silhouette and shape from stereo methods. In the separating step, the depth maps are generated by a proposed variational model initialized with visual hull. Then, the depth maps are merged to obtain a complete 3D model. The proposed shape recovery method shows promising reconstruction quality on the Middlebury's datasets and the datasets captured by a real multi-camera system. By using the proposed variational model, the depth map for each view contains detailed geometries, avoiding the over-smoothness suffered in most modern multi-view stereo methods. The final reconstructed 3D mesh is watertight and has rich geometric details. Moreover, there is no constraint on the number of visible views for each reconstructed 3D point.

B. Motion Capture

While shape recovery methods aim at reconstructing static objects, motion capture methods try to capture motions of dynamic objects. The motion capture methods, especially for human performances, can be classified into two categories: marker-based methods and markerless methods.

Marker-based methods generally need to attach many optical markers on the object to gather motion information. Though the motions estimated by these methods are accurate, they often demand tedious manual adjustments to clean up recorded data. Moreover, they often require skin-tight garments and reflective markers, which makes it difficult to capture general shapes and textures. Park et al. [36] try to extract a model of human skin deformation by using hundreds of optical markers. This method provides realistic animation results, but the manual mark-up and data cleanup are quite time-consuming. Other researchers design patterns that can be printed on the clothing to estimate complicated motions of garments [37], [38]. However, markers, either attached or printed, make it inconvenient to capture the motions of humans wearing casual clothing.

Markerless motion capture approaches are designed to overcome some restrictions of marker-based techniques. Some work focus on the face deformation which can keep mesh consistency [39]–[41]. For body deformation, some methods try to capture some shape details in addition to skeletal joint parameters by adapting the models closer to the observed silhouettes [42], or by using range scan data [43]. However, these algorithms require the performer to be tightly dressed. Rosenhahn et al. [44] aim at capturing the performances of humans in more general clothes by jointly relying on a kinematic body and cloth models. However, they focus on joint parameter estimation under occlusion rather than detailed geometry capturing. Brox et al. [45] combine region fitting and dense optical flow for 3D tracking. But they also do not focus on the accurate geometry recovery of time-varying shapes. Vlasic et al. [46] try to capture the motion of both the skeleton and the shape from synchronized multi-view video sequences. This method is based on an articulated template of the performer, which is obtained with the help of a 3D laser scanner. A main disadvantage of this method is that a considerable amount of manual interaction is required, namely up to every 20th frame, to correct the errors of the skeleton estimation. Gall et al. [13] also try to achieve motion capture by joint skeleton tracking and surface deformation, which is evaluated on the HumanEva benchmark [47]. However, the kinematic skeleton only allows the tracking of rigid motions.

Recently, some new methods for animation design [48]–[50], animation editing [51], and deformation transfer [52] are no longer based on kinematic skeletons and motion parametrizations. Instead, they employ surface models and general shape deformation approaches, which enables capturing of both rigid and non-rigid deformation. Along this avenue, de Aguiar et al. [53] abandon the traditional skeletal shape or motion parametrization, and achieve high quality performance capture by volumetric deformation based on Ref. [54]. This method needs users to specify the key vertices and a multi-view stereo method is used to refine the surface at each time instant. Moreover, as many other markerless methods, they employ a laser range scanner to reconstruct the initial 3D model, and hence some local surface details remain unchanged over time, which looks unrealistic.

In this paper, we apply an iterative volumetric Laplacian deformation framework inspired by Sorkine's work for tri-

angle meshes [50]. In order to implicitly preserve certain shape properties, we use a tetrahedral construction instead of a triangle mesh construction. Positional constraints are first computed by a method based on sparse representation and three spatio-temporal selection criteria. Then, an iterative Laplacian deformation system with rigidity requirement for the local transformations is applied. Because this method does not employ a kinematic skeleton, both rigid and non-rigid deformation can be achieved. Our markerless method requires no manual processing and enables simultaneous capturing of the shape, motion and texture of an arbitrarily dressed human.

III. Overview of the Proposed Method

Fig. 1 illustrates the workflow of our shape and motion capture approach. The input data are multi-view video sequences. To calculate the depth maps, we propose a variational model (Section IV). For each view, a depth map represented with a point cloud is reconstructed through optimizing the variational model. Then, the point clouds of all the views are merged together to generate a complete 3D mesh (Section V). Finally, taking this 3D mesh as the underlying scene representation, we apply an iterative volumetric Laplacian deformation framework with a new positional-constraint computation scheme to extract the 3D motion information from the multi-view video sequences and generate time-varying shapes (Section VI). Experiments show that the automatic shape recovery and motion capture method can capture the shape and motion from multi-view video sequences, without any manual intervention or optical markers (Section VII).

IV. Variational Model for Depth Map Computation

Depth map estimation is a key step to achieve accurate shape recovery from a set of multi-view images. The correspondence estimation in depth map computation is quite similar to the optical flow estimation: Both are to solve the correspondences of objects between adjacent images. Of many optical flow estimation methods, variational methods [55]–[58] are one category of most successful approaches. In Ref. [56], optical flow is estimated by solving a variational model which contains a robust data term and a spatio-temporal TV (Total Variation) regularizer to ensure the accuracy and denseness. The model is linearized in a numerical scheme with two nested fixed point iterations and a multi-resolution strategy is used to avoid falling into local minimum. Experiments show that the novel method gives significantly smaller errors than previous methods, thanks to the design of the variational model.

Inspired by the work in Ref. [56], we propose a variational model for depth map computation in shape recovery. The images are captured by multiple cameras in shape recovery rather than a single camera in the optical flow estimation. Therefore, besides the luminance constancy and gradient constancy constraints, an epipolar geometry constraint is incorporated into the model to ensure accurate estimation, and also to reduce the searching range. In the data term, an occlusion mask is used to handle the occlusion problem. The spatial consistency of multi-view images is regularized by a total variation term.

A. Problem Statement

In the proposed method, the input is a set of N images captured by N cameras, and the goal is to determine a dense depth map for each view with the help of its neighboring view. Then, the N depth maps represented with point clouds are merged and reconstructed into a 3D mesh.

Consider two images of the same object captured by two cameras, c and $c+1$, whose optical centers are $\mathbf{O}_c$ and $\mathbf{O}_{c+1}$, respectively. For each 3D point on the object, there is an epipolar plane passing through the 3D point and the optical centers, $\mathbf{O}_c$ and $\mathbf{O}_{c+1}$. The intersection lines of the epipolar plane and the two image planes are called *epipolar lines*. Each epipolar line is a projection of the ray connecting the 3D point with the optical center of the other image plane onto the current image plane. Hence, each pixel in the image plane of camera c has an associated epipolar line in the image plane of camera $c+1$.

As shown in Fig. 2, $\mathbf{x} := (x, y, c)$ denotes a pixel location (x, y) in image of the reference camera c. Let $I : \Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}$ denote a set of multi-view images, and the luminance at $\mathbf{x}$ is denoted as $I(\mathbf{x})$. On the associated epipolar line of $\mathbf{x}$ in image of camera $c+1$, $\mathbf{x}_b := (x_b, y_b, c)$ is the projection of the optical center $\mathbf{O}_c$ to the image of camera $c+1$, called *epipole*. The correspondence $\mathbf{q}$ of pixel $\mathbf{x}$ in image of $c+1$ is on the associated epipolar line. Therefore, the 3D point position associated with $\mathbf{x}$ can be determined by computing the displacement $\mathbf{d} := (a, b, 1)$ relative to the epipole $\mathbf{x}_b$ along the epipolar line. The task of the proposed variational model is to compute the displacement map for each multi-view image.

Denote $\mathbf{w} := (u, v, 1)$ as the disparity between the pixel position $\mathbf{x} := (x, y, c)$ and the correspondence position $\mathbf{q}$, u and v its *x-component* and *y-component*. The relationship between $\mathbf{w}$ and $\mathbf{d}$ is given by

$$\mathbf{x} + \mathbf{w} = \mathbf{x}_b + \mathbf{d}. \tag{1}$$

Let $\Delta\mathbf{d} := (da, db, 0)$ and $\Delta\mathbf{w} := (du, dv, 0)$ be the variation in $\mathbf{d}$ and $\mathbf{w}$, respectively. As shown in Fig. 2, the variation in $\mathbf{d}$ is equal to that in $\mathbf{w}$, i.e., $\Delta\mathbf{d} = \Delta\mathbf{w}$. Therefore, we have

$$\begin{aligned} da &= du, \\ db &= dv. \end{aligned} \tag{2}$$

In our method, a visual hull model is obtained by EPVH (Exact Polyhedral Visual Hulls) [14] from the multi-view silhouette images. For the view c, the initial depth map represented with a point cloud is generated by estimating the visibility of the visual hull model. Specifically, if a visual ray passing through a image pixel $\mathbf{x}$ intersects the visual hull at least once, the first intersection point will be treated as a visible point. The obtained visible 3D points are projected to the neighboring view $c+1$. For each projected point, the displacement relative to the epipole along the epipolar line is computed as the initial displacement $\mathbf{d}$. Then, the displacement $\mathbf{d}$ is optimized using our variational model proposed in the next section.

B. The Variational Model

Before deriving a variational formulation for our shape recovery task, we first present a description for the four

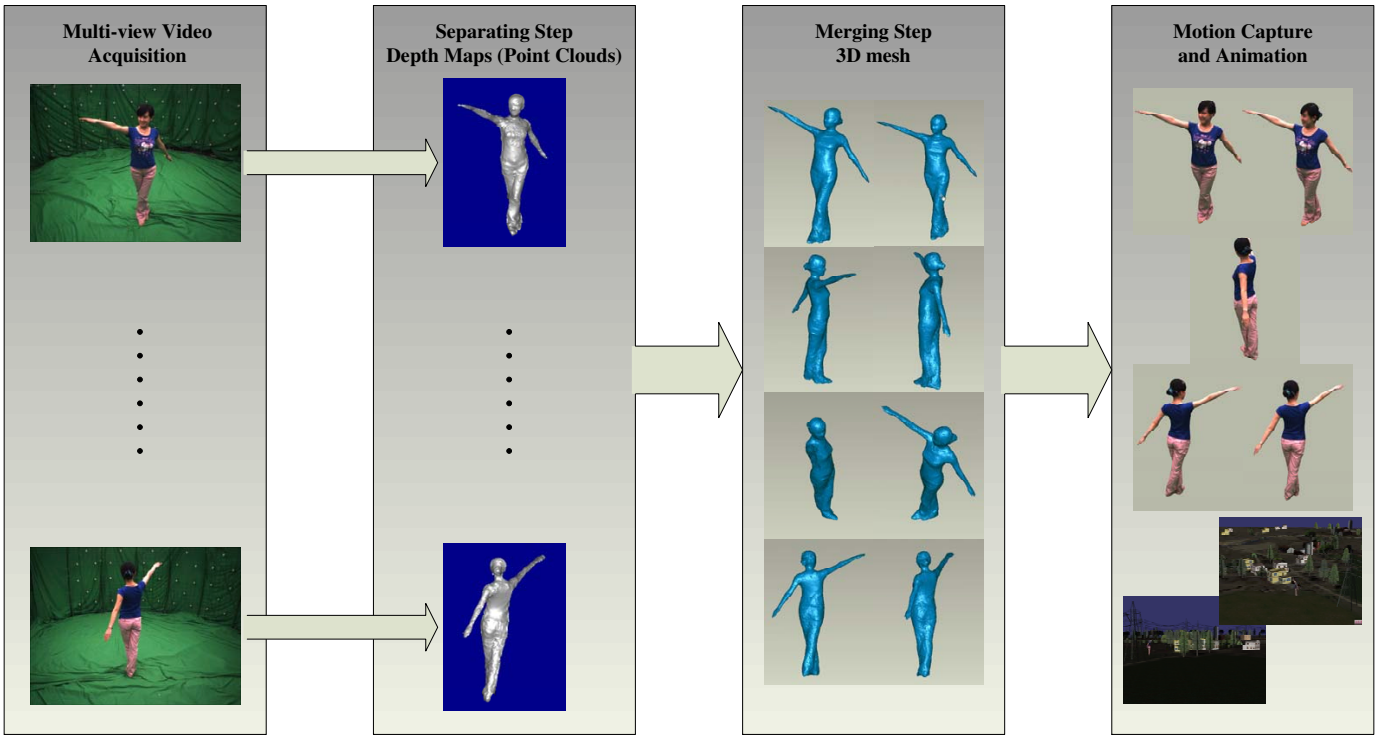


Fig. 1. The workflow of our method.

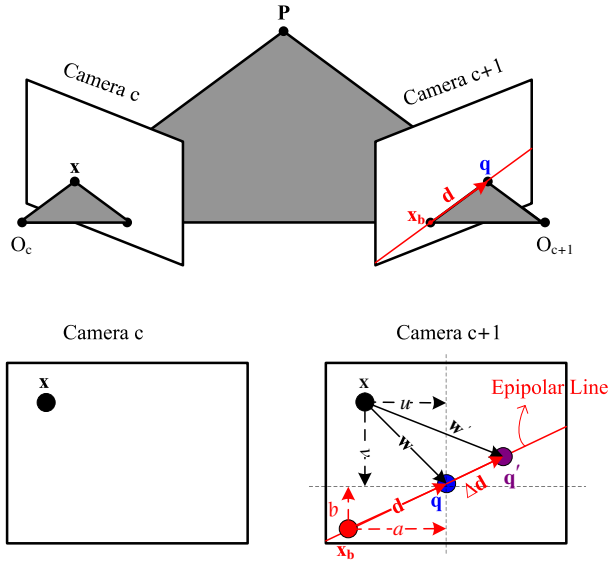


Fig. 2. Illustration of the epipolar geometry.

constraints in our variational model.

1) Epipolar Geometry Constraint: Epipolar geometry, as a specific example of multi-view geometry, is the only available geometric constraint between a pair of stereo images. We estimate the depth map by computing the displacement from the epipole along the epipolar line. This geometric relationship is the basis for the following three constraints.

2) Luminance Constancy Constraint: For a point on the surface of a 3D object, its projections onto image planes (referred to as *correspondences*) approximately have the same luminance values when the lighting is uniform, the reflective property of the object is isotropic for different views, and the color responses of cameras are consistent. Assuming that the luminance values are equivalent between pixel (x, y) in image of camera c and shifted pixel $(x+u, y+v)$ in image of camera $c+1$, this yields the nonlinear luminance constancy constraint:

$$\begin{aligned} I(x, y, c) &= I(x + u, y + v, c + 1) \\ &= I(x_b + a, y_b + b, c + 1). \end{aligned} \tag{3}$$

3) Gradient Constancy Constraint: The luminance constancy assumption can be violated by luminance variation and color inconsistency. Therefore, to allow some small variations in luminance values, we assume that the gradients at the corresponding positions are constant:

$$\nabla I(x, y, c) = \nabla I(x_b + a, y_b + b, c + 1), \tag{4}$$

where $\nabla = (\partial_x, \partial_y)^T$ denotes the spatial gradient.

4) Smoothness Constraint: To handle the outliers in the estimation, it is necessary to impose a smoothness constraint on the depth map. One can use a global smoothness assumption. However, this assumption does not take the geometrical singularities into account. In order to preserve the discontinuities around shape variations (e.g., folds and wrinkles), we generalize the smoothness assumption by demanding a piecewise-smooth displacement field, which is defined as

$$\min \left(|\nabla a|^2 + |\nabla b|^2 \right). \tag{5}$$

Based on the above analysis, we propose an energy function that penalizes deviations from these model assumptions. The

goal of our variational model is to find the parameters a and b by minimizing the energy function:

$$\min_{a,b} \big(E_{data}\,(a,b) + \alpha E_{smooth}\,(a,b)\big), \tag{6}$$

where $E_{data}(a,b)$ is a data term, $E_{smooth}(a,b)$ is a smoothness term, and α is a penalization parameter.

For the data term, $E_{data}(a,b)$ is composed of two terms associated with the luminance and gradient constancy assumptions:

$$\begin{aligned} E_{data}(a,b) = \int_\Omega \beta(\mathbf{x})\big(\,&|I(\mathbf{x}_b+\mathbf{d}) - I(\mathbf{x})|^2 \\ &+ \gamma\,|\nabla I(\mathbf{x}_b+\mathbf{d}) - \nabla I(\mathbf{x})|^2\,\big)d\mathbf{x}, \end{aligned} \tag{7}$$

where γ is a penalization factor and $\beta(\mathbf{x})$ is an occlusion map to exclude occluded pixels in the data term. The occlusion map is set at one for non-occluded pixels and zero otherwise, computed similarly as in Ref. [4]. Specifically, the displacement $\mathbf{d}$ is first warped to the neighboring view via Z buffering, and then is backwarped to the current view. If the distance between the two displacements is smaller than a tolerance (1.5 pixels), the occlusion map is set at one, and zero otherwise. Since the penalizers are quadratic in Eq. (7), the outliers weight too much on the estimation. Therefore, the quadratic penalizer is replaced by a robust function $\Psi(s^2) = \sqrt{s^2+\varepsilon^2}$, where ε is a small constant, which yields a TV regularization. This regularization corresponds to an ℓ_1 norm minimization, but is still differentiable everywhere. Therefore, the data term $E_{data}(a,b)$ becomes

$$\begin{aligned} E_{data}(a,b) = \int_\Omega \beta(\mathbf{x})\Psi\big(&|I(\mathbf{x}_b+\mathbf{d}) - I(\mathbf{x})|^2 \\ &+ \gamma|\nabla I(\mathbf{x}_b+\mathbf{d}) - \nabla I(\mathbf{x})|^2\big)d\mathbf{x}. \end{aligned} \tag{8}$$

Due to the small positive constant ε, $\Psi(s^2)$ is still convex, which offers advantages in the minimization process. In our implementation, ε is set at a fixed value 0.001.

For the smoothness term, $E_{smooth}(a,b)$ describes the model assumption of a piecewise-smooth displacement field:

$$E_{smooth}(a,b) = \int_\Omega \Psi\big(|\nabla a|^2 + |\nabla b|^2\big)d\mathbf{x}. \tag{9}$$

Considering the outliers problem, the same function $\Psi(\cdot)$ is also used as in the data term.

With the derived data term (8) and smoothness term (9), the energy functional to be minimized is expressed as

$$\begin{aligned} E(a,b) &= \int_\Omega F(a,b)d\mathbf{x} \\ &= \int_\Omega \big(\beta(\mathbf{x})\Psi(|I(\mathbf{x}_b+\mathbf{d}) - I(\mathbf{x})|^2 \\ &\quad + \gamma|\nabla I(\mathbf{x}_b+\mathbf{d}) - \nabla I(\mathbf{x})|^2) \\ &\quad + \alpha\Psi(|\nabla a|^2 + |\nabla b|^2)\big)d\mathbf{x}. \end{aligned} \tag{10}$$

On the one hand, the data term reflects the fidelity of the computed depth map, which ensures the accuracy of the reconstruction. On the other hand, the smoothness term takes the smoothness into account, and plays an important role in ensuring the completeness of the reconstruction. By bridging the two terms with a penalization parameter α and minimizing the resulting energy functional, the reconstructed depth map will present rich geometric details without loss of smoothness.

V. Shape Recovery Algorithm

In this section, we first give a numerical solution for the minimization of the energy functional. The energy functional (10) is minimized by two nested fixed point iterations with a multi-resolution strategy, similarly as in Ref. [56]. The full algorithm is summarized in Algorithm 1. Then, with the obtained depth maps of all the views, the whole 3D mesh is generated by a marching cubes method with silhouette constraints.

A. Euler-Lagrange Equations

Since the integrand $F(a,b)$ in the energy functional (10) is differentiable, the minimum of the energy functional should satisfy the Euler-Lagrange equations:

$$\begin{aligned} \frac{\partial F}{\partial a} - \frac{d}{dx}\Big(\frac{\partial F}{\partial a_x}\Big) - \frac{d}{dy}\Big(\frac{\partial F}{\partial a_y}\Big) &= 0, \\ \frac{\partial F}{\partial b} - \frac{d}{dx}\Big(\frac{\partial F}{\partial b_x}\Big) - \frac{d}{dy}\Big(\frac{\partial F}{\partial b_y}\Big) &= 0, \end{aligned} \tag{11}$$

where $a_x = \partial a/\partial x$, $a_y = \partial a/\partial y$, $b_x = \partial b/\partial x$, and $b_y = \partial b/\partial y$. The reflecting boundary conditions are $\partial_n a = 0$ and $\partial_n b = 0$ on $\partial\Omega$.

To solve the equation, Eq. (11) can be expanded and further reformulated as Eq. (12) with the following abbreviations:

$$\begin{aligned} I_x &:= \partial_x I(\mathbf{x}_b+\mathbf{d}) \\ I_y &:= \partial_y I(\mathbf{x}_b+\mathbf{d}) \\ I_z &:= I(\mathbf{x}_b+\mathbf{d}) - I(\mathbf{x}) \\ I_{xx} &:= \partial_{xx} I(\mathbf{x}_b+\mathbf{d}) \\ I_{xy} &:= \partial_{xy} I(\mathbf{x}_b+\mathbf{d}) \\ I_{yy} &:= \partial_{yy} I(\mathbf{x}_b+\mathbf{d}) \\ I_{xz} &:= \partial_x I(\mathbf{x}_b+\mathbf{d}) - \partial_x I(\mathbf{x}) \\ I_{yz} &:= \partial_y I(\mathbf{x}_b+\mathbf{d}) - \partial_y I(\mathbf{x}). \end{aligned} \tag{13}$$

B. Numerical Scheme

When the displacements are larger than one pixel, a minimization algorithm could easily be trapped into a local minimum. However, with continuation methods (or graduated non-convexity methods) [59], one can obtain considerably more satisfactory results than a conventional optimization strategy. The key idea is to transform the complicated original functional to a simplified smoother functional, where a unique minimum exists. Using the heuristic that the minimum of the simplified functional is a good initialization for solving a refined version, it becomes possible to approach the global minimum of the original functional step by step. Therefore, in our method, an incremental multi-resolution strategy [60] is

$$\begin{aligned}\beta(\mathbf{x})\Psi'\Big(I_z^2+\gamma(I_{xz}^2+I_{yz}^2)\Big)\Big(I_xI_z+\gamma(I_{xx}I_{xz}+I_{xy}I_{yz})\Big)-\alpha\mathrm{div}\left(\Psi'(|\nabla a|^2+|\nabla b|^2)\nabla a\right)=0\\ \beta(\mathbf{x})\Psi'\Big(I_z^2+\gamma(I_{xz}^2+I_{yz}^2)\Big)\Big(I_yI_z+\gamma(I_{yy}I_{yz}+I_{xy}I_{xz})\Big)-\alpha\mathrm{div}\left(\Psi'(|\nabla a|^2+|\nabla b|^2)\nabla b\right)=0\end{aligned} \tag{12}$$

adopted, which can ensure that the algorithm converges to a global minimum. Specifically, by iteratively performing low-pass filtering and downsampling, a series of smoothed versions of the original multi-view images are obtained for multi-resolution optimization, which forms a pyramidal structure. A 2D Gaussian filter is employed in the low-pass filtering. For downsampling, a relatively small downsampling factor ($10/9$ in our implementation) is used since continuation methods achieve better performance when the transition from the coarse level to the fine level is smooth enough.

Besides, the energy functional in Eq. (12) is highly nonlinear, and cannot be solved with common numerical methods. There are two kinds of nonlinearities in Eq. (12): One is due to the non-quadratic penalizer $\Psi(s^2)=\sqrt{s^2+\varepsilon^2}$; the other is introduced by the non-linearized data constraints. To remove the nonlinearities, an algorithm with two nested fixed point iterations (similar to Ref. [56]) is used. This algorithm consequently postpones all linearisations to the numerical scheme, which improves its convergence to the global minimum.

Since the problem is non-linear and non-convex, careful initialization is important in order to avoid local minima. We choose to start our algorithm with non-zero values using the visible points of visual hull from the current view. To compute the visible points for each view, we simply calculate the pseudo intersection points of the reprojected rays passing through the image pixels.

Let $\mathbf{d}^k:=(a^k,b^k,1)$ be the solution at the kth scale. The initial displacement map at the finest scale is downsampled to the coarsest scale as the initialization $\mathbf{d}^0$. Let I_*^k be the abbreviations defined in formula (13), and replace $\mathbf{d}$ with $\mathbf{d}^k$ to indicate the scale index[2]. Then, according to Eq. (12), $\mathbf{d}^{k+1}=(a^{k+1},b^{k+1},1)$ is the solution of Eq. (14). When a fixed point in $\mathbf{d}^k$ is reached, we go to the next finer scale, and use this solution as the initialization for the fixed point iteration on that scale. To remove the nonlinearity in I_*^{k+1}, the first order Taylor expansions are used:

$$\begin{aligned}I_z^{k+1}&\approx I_z^k+I_x^kda^k+I_y^kdb^k,\\ I_{xz}^{k+1}&\approx I_{xz}^k+I_{xx}^kda^k+I_{xy}^kdb^k,\\ I_{yz}^{k+1}&\approx I_{yz}^k+I_{xy}^kda^k+I_{yy}^kdb^k,\end{aligned} \tag{15}$$

where $a^{k+1}=a^k+da^k$ and $b^{k+1}=b^k+db^k$, i.e., the unknown parameters a^{k+1} and b^{k+1} are split into two parts: the solution of the previous iteration step, a^k and b^k, and the unknown increments, da^k and db^k. With the following two abbreviations:

$$\begin{aligned}(\Psi')_D^k&:=\Psi'\begin{pmatrix}(I_z^k+I_x^kda^k+I_y^kdb^k)^2\\ +\gamma(I_{xz}^k+I_{xx}^kda^k+I_{xy}^kdb^k)^2\\ +\gamma(I_{yz}^k+I_{xy}^kda^k+I_{yy}^kdb^k)^2\end{pmatrix},\\ (\Psi')_S^k&:=\Psi'\left(|\nabla(a^k+da^k)|^2+|\nabla(b^k+db^k)|^2\right),\end{aligned} \tag{16}$$

[2]The symbol "$*$" denotes the subscripts of I in formula (13).

Eq. (14) can be rewritten as Eq. (17).

The only remaining nonlinearity in Eq. (17) is due to $\Psi'(\cdot)$, and can be removed by an inner fixed point iteration loop. In this inner iteration loop, da and db are both initialized with zeros. Therefore, Eq. (17) is cast into a linear system Eq. (18), where l denotes the inner fixed point iteration index, and $(\Psi')_D^{k,l}$ and $(\Psi')_S^{k,l}$ are calculated as formula (16) using $da^{k,l}$ and $db^{k,l}$. This linear system can be solved by SOR (Successive Over Relaxation) iterations after discretization. Let m denote the iteration index for the SOR iterations, and then the iteration scheme for each pixel i is presented in Eq. (19), where $(\Psi'_S)_{i\sim j}^{k,l}$ denotes the discrete diffusivity between pixel i and pixel j, $\mathcal{N}^-(i)$ the index set of the causal neighbors of pixel i, and $\mathcal{N}^+(i)$ the index set of the non-causal neighbors. The scheme converges when the relaxation parameter ω belongs to $(0,2)$, and we set ω at 1.6 in experiments.

C. Full Reconstruction Algorithm

Our reconstruction algorithm consists of two steps: the separating step and the merging step. The algorithm for the separating step, including the initialization, is detailed in Algorithm 1. The merging step does not require coordinate transformation since the point clouds for all the views are already in the unified world coordinate system. Therefore, they are first put together and the prominent outliers in the whole point cloud are removed by using the visual hull as a constraint. Then, a traditional space partitioning method is used to reconstruct the mesh. Specifically, the overall point cloud is first converted to an octree representation, and then the mesh reconstruction reduces to an iso-surface contouring problem using either marching cubes [15] or dual contouring [61]. The main trend of merging methods is to use complicated techniques to achieve good reconstructions; instead, we choose to pursue the simplicity and rapidity of the method since a simple merging method can also produce high quality 3D reconstructions thanks to the accurate depth map estimation in the separating step. Therefore, the marching cubes method is used in our merging step. The obtained mesh is guaranteed to be valid and watertight. For further denoising, simplication or remeshing is optional and can be employed to improve the quality of the obtained meshes.

Note that our shape recovery method can also be considered as a shape optimization method. The initial input model of our method can be the reconstructed 3D mesh by other algorithms, and the output 3D mesh is a refined version of the input model.

VI. Motion Capture Algorithm

Taking the 3D mesh generated by the above shape recovery method as the underlying scene representation, we can capture the motions of the 3D object, especially human performances.

$$\begin{aligned}\beta(\mathbf{x})\Psi'\left((I_z^{k+1})^2+\gamma((I_{xz}^{k+1})^2+(I_{yz}^{k+1})^2)\right)\left(I_x^k I_z^{k+1}+\gamma(I_{xx}^k I_{xz}^{k+1}+I_{xy}^k I_{yz}^{k+1})\right)\\ -\alpha\text{div}\left(\Psi'(|\nabla a^{k+1}|^2+|\nabla b^{k+1}|^2)\nabla a^{k+1}\right)=0\\ \beta(\mathbf{x})\Psi'\left((I_z^{k+1})^2+\gamma((I_{xz}^{k+1})^2+(I_{yz}^{k+1})^2)\right)\left(I_y^k I_z^{k+1}+\gamma(I_{yy}^k I_{yz}^{k+1}+I_{xy}^k I_{xz}^{k+1})\right)\\ -\alpha\text{div}\left(\Psi'(|\nabla a^{k+1}|^2+|\nabla b^{k+1}|^2)\nabla b^{k+1}\right)=0\end{aligned} \tag{14}$$

$$\begin{aligned}\beta(\mathbf{x})(\Psi')_D^k\begin{pmatrix} I_x^k(I_z^k+I_x^k da^k+I_y^k db^k)\\ +\gamma I_{xx}^k(I_{xz}^k+I_{xx}^k da^k+I_{xy}^k db^k)\\ +\gamma I_{xy}^k(I_{yz}^k+I_{xy}^k da^k+I_{yy}^k db^k)\end{pmatrix}-\alpha\text{div}\left((\Psi')_S^k\nabla(a^k+da^k)\right)=0\\ \beta(\mathbf{x})(\Psi')_D^k\begin{pmatrix} I_y^k(I_z^k+I_x^k da^k+I_y^k db^k)\\ +\gamma I_{xy}^k(I_{xz}^k+I_{xx}^k da^k+I_{xy}^k db^k)\\ +\gamma I_{yy}^k(I_{yz}^k+I_{xy}^k da^k+I_{yy}^k db^k)\end{pmatrix}-\alpha\text{div}\left((\Psi')_S^k\nabla(b^k+db^k)\right)=0\end{aligned} \tag{17}$$

$$\begin{aligned}\beta(\mathbf{x})(\Psi')_D^{k,l}\begin{pmatrix} I_x^k(I_z^k+I_x^k da^{k,l+1}+I_y^k db^{k,l+1})\\ +\gamma(I_{xx}^k(I_{xz}^k+I_{xx}^k da^{k,l+1}+I_{xy}^k db^{k,l+1})\\ +I_{xy}^k(I_{yz}^k+I_{xy}^k da^{k,l+1}+I_{yy}^k db^{k,l+1}))\end{pmatrix}-\alpha\text{div}\left((\Psi')_S^{k,l}\nabla(a^k+da^{k,l+1})\right)=0\\ \beta(\mathbf{x})(\Psi')_D^{k,l}\begin{pmatrix} I_y^k(I_z^k+I_x^k da^{k,l+1}+I_y^k db^{k,l+1})\\ +\gamma(I_{xy}^k(I_{xz}^k+I_{xx}^k da^{k,l+1}+I_{xy}^k db^{k,l+1})\\ +I_{yy}^k(I_{yz}^k+I_{xy}^k da^{k,l+1}+I_{yy}^k db^{k,l+1}))\end{pmatrix}-\alpha\text{div}\left((\Psi')_S^{k,l}\nabla(b^k+db^{k,l+1})\right)=0\end{aligned} \tag{18}$$

Algorithm 1 Full algorithm for depth map computation

Require: $M \in \mathbb{N}, M \geq 1$, $maxOutIter$, $maxInIter \in \mathbb{N}$.
For each image pair of camera c and camera $c+1$, compute the epipolar lines in image of camera $c+1$ for each pixel in image of camera c. Initialize $\mathbf{d} = (a, b)$ with the visible points of visual hull.

for $scale = M$ to 1 **do**
  for ($k = 0$ to $maxOutIter$) or (no improvement) **do**
    Compute the occlusion map $\beta(\mathbf{x})$.
    $da \leftarrow 0, db \leftarrow 0$.
    for ($l = 0$ to $maxInIter$) or (no improvement) **do**
      Compute $(\Psi')_D$ and $(\Psi')_S$ using da, db.
      Obtain (da, db) via SOR iterations (Eq. (19)).
    end for
    $a \leftarrow a + da$, $b \leftarrow b + db$.
  end for
  Update $\mathbf{d}$ for the next scale using bilinear interpolation.
end for
Output the point cloud according to the final $\mathbf{d}$.

In order to implicitly preserve certain shape properties, we first create a tetrahedral version of the surface mesh by a finite element method and a finite volume method [62], [63]. Then, an iterative as-rigid-as-possible deformation scheme is applied to extract 3D motion information from the multi-view videos. Specifically, we first calculate a set of positional constraint candidates with a new method based on sparse representation, and then select the most reasonable ones with three spatio-temporal selection criteria. An iterative Laplacian deformation framework with rigidity requirement for the local transformations is driven by these positional constraints. By applying this algorithm to all subsequent time instants, the mesh can be animated over the whole multi-view video sequences.

A. Laplacian Deformation Framework

Sorkine et al. [64] review the definition of differential coordinates (δ-coordinates) and the associated mesh Laplacian operator for triangle meshes. In our motion capture method, tetrahedral meshes are used since the volumetric deformation framework can better prevent unintuitive shape transformations, such as local self-intersections of opposing surfaces. Furthermore, it enables distance preservation not only on the surface of an object, but also throughout its interior, which makes the deformation resistant to changes in volume and cross-sectional areas. In this section, we describe the relevant definitions and the framework of Laplacian mesh deformation for tetrahedral meshes.

1) Differential Coordinates: Let $\mathscr{M} := (V, E, T)$ denote a given tetrahedral mesh with n vertices, V, E, and T the set of its vertices, edges, and tetrahedra, respectively. Each vertex $\mathbf{v}_i := (x_i, y_i, z_i)$ in V is under the absolute Cartesian coordinate. The differential coordinates (δ-coordinates) of $\mathbf{v}_i$ are defined as the difference between the absolute coordinates of $\mathbf{v}_i$ and the mass center of its immediate neighbors on the mesh:

$$\boldsymbol{\delta}_i = (\delta_i^{(x)}, \delta_i^{(y)}, \delta_i^{(z)}) = \mathbf{v}_i - \frac{1}{d_i}\sum_{j\in\mathcal{N}(i)}\mathbf{v}_j, \tag{20}$$

where the set $\mathcal{N}(i) = \{j|(\mathbf{v}_i, \mathbf{v}_j) \in E\}$ contains the indices of immediate neighbors of $\mathbf{v}_i$ and d_i is the number of elements in $\mathcal{N}(i)$.

$$
\begin{aligned}
da_i^{k,l,m+1} =& (1-\omega) da_i^{k,l,m} \\
&+\omega \frac{\sum\limits_{j\in\mathcal{N}^-(i)} (\Psi'_S)_{i\sim j}^{k,l}(a_j^k + da_j^{k,l,m+1}) + \sum\limits_{j\in\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l}(a_j^k + da_j^{k,l,m}) - \sum\limits_{j\in\mathcal{N}^-(i)\cup\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l} a_i^k}{\sum\limits_{j\in\mathcal{N}^-(i)\cup\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l} + \frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left(\left((I_x)_i^k\right)^2 + \gamma\left((I_{xy})_i^k\right)^2 + \gamma\left((I_{xx})_i^k\right)^2\right)} \\
&-\omega \frac{\frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left((I_x)_i^k\left((I_y)_i^k db_i^{k,l,m} + (I_z)_i^k\right)\right)}{\sum\limits_{j\in\mathcal{N}^-(i)\cup\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l} + \frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left(\left((I_x)_i^k\right)^2 + \gamma\left((I_{xy})_i^k\right)^2 + \gamma\left((I_{xx})_i^k\right)^2\right)} \\
&-\omega \frac{\frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left(\gamma\left((I_{xx})_i^k\left((I_{xy})_i^k db_i^{k,l,m} + (I_{xz})_i^k\right) + (I_{xy})_i^k\left((I_{yy})_i^k db_i^{k,l,m} + (I_{yz})_i^k\right)\right)\right)}{\sum\limits_{j\in\mathcal{N}^-(i)\cup\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l} + \frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left(\left((I_x)_i^k\right)^2 + \gamma\left((I_{xy})_i^k\right)^2 + \gamma\left((I_{xx})_i^k\right)^2\right)} \\
db_i^{k,l,m+1} =& (1-\omega) db_i^{k,l,m} \\
&+\omega \frac{\sum\limits_{j\in\mathcal{N}^-(i)} (\Psi'_S)_{i\sim j}^{k,l}(b_j^k + db_j^{k,l,m+1}) + \sum\limits_{j\in\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l}(b_j^k + db_j^{k,l,m}) - \sum\limits_{j\in\mathcal{N}^-(i)\cup\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l} b_i^k}{\sum\limits_{j\in\mathcal{N}^-(i)\cup\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l} + \frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left(\left((I_y)_i^k\right)^2 + \gamma\left((I_{xy})_i^k\right)^2 + \gamma\left((I_{yy})_i^k\right)^2\right)} \\
&-\omega \frac{\frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left((I_y)_i^k\left((I_x)_i^k da_i^{k,l,m+1} + (I_z)_i^k\right)\right)}{\sum\limits_{j\in\mathcal{N}^-(i)\cup\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l} + \frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left(\left((I_y)_i^k\right)^2 + \gamma\left((I_{xy})_i^k\right)^2 + \gamma\left((I_{yy})_i^k\right)^2\right)} \\
&-\omega \frac{\frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left(\gamma\left((I_{xy})_i^k\left((I_{xx})_i^k da_i^{k,l,m+1} + (I_{xz})_i^k\right) + (I_{yy})_i^k\left((I_{xy})_i^k da_i^{k,l,m+1} + (I_{yz})_i^k\right)\right)\right)}{\sum\limits_{j\in\mathcal{N}^-(i)\cup\mathcal{N}^+(i)} (\Psi'_S)_{i\sim j}^{k,l} + \frac{\beta(\mathbf{x}_i)(\Psi'_D)_i^{k,l}}{\alpha}\left(\left((I_y)_i^k\right)^2 + \gamma\left((I_{xy})_i^k\right)^2 + \gamma\left((I_{yy})_i^k\right)^2\right)}
\end{aligned}
\tag{19}
$$

Let $\mathbf{D}$ be a diagonal matrix with $\mathbf{D}_{ii} = d_i$ and $\mathbf{W}$ the adjacency (connectivity) matrix of $\mathscr{M}$ with

$$
\mathbf{W}_{ij} = \begin{cases} 1 & (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases} \tag{21}
$$

The absolute Cartesian coordinates can be transformed into the differential coordinates with an operator $\mathbf{L}$ defined as

$$
\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}. \tag{22}
$$

That is, $\mathbf{L}\mathbf{x} = \boldsymbol{\delta}^{(x)}$, $\mathbf{L}\mathbf{y} = \boldsymbol{\delta}^{(y)}$, and $\mathbf{L}\mathbf{z} = \boldsymbol{\delta}^{(z)}$, where $\mathbf{x}$ is an n-vector containing the x absolute coordinates of all the vertices, and $\mathbf{y}$ and $\mathbf{z}$ are defined in the same manner. The matrix $\mathbf{L}$ is called the *topological Laplacian* of the mesh.

2) Laplacian Mesh Deformation: The appealing properties of Laplacian operators have been explored in various ways. Sorkine et al. [65] propose a geometry compression algorithm, which benefits from the strong quantization in the differential coordinates. Alexa [66] shows the effectiveness of Laplacian differential coordinates for mesh deformation. He suggests differential coordinates as a local mesh description, which is a more suitable constraint for a global deformation of the mesh.

For mesh deformation, the differential coordinates of the vertices of $\mathscr{M}$ are assumed to be invariable. Hence, the problem to be solved is how to recover the Cartesian coordinates of $\mathscr{M}$'s vertices, given the differential coordinates. We cannot uniquely restore the Cartesian coordinates from the differential coordinates, because the matrix $\mathbf{L}$ is singular and, therefore, the expression $\mathbf{x} = \mathbf{L}^{-1}\boldsymbol{\delta}^{(x)}$ is undefined (taking the *x-component* for example). The sum of each row of $\mathbf{L}$ is zero, which implies that $\mathbf{L}$ has a non-trivial eigenvector $(1, 1, \ldots, 1)^T$ associated with the zero eigenvalue. Define the Laplacian operator with the weights ω_{ij} that sum up to 1 and translate the mesh by a vector $\mathbf{t}$ to obtain the new vertices $\{\mathbf{v}'_i\}_{1\le i\le n}$, then we have

$$
\begin{aligned}
L(\mathbf{v}'_i) &= \sum_{j\in\mathcal{N}(i)} \omega_{ij}\left(\mathbf{v}'_i - \mathbf{v}'_j\right) \\
&= \sum_{j\in\mathcal{N}(i)} \omega_{ij}\left((\mathbf{v}_i + \mathbf{t}) - (\mathbf{v}_j + \mathbf{t})\right) \\
&= \sum_{j\in\mathcal{N}(i)} \omega_{ij}\left(\mathbf{v}_i - \mathbf{v}_j\right) \\
&= L(\mathbf{v}_i).
\end{aligned}
\tag{23}
$$

Hence, the differential coordinates are translation invariant, which also demonstrates the singularity of the Laplacian matrix $\mathbf{L}$. Eq. (23) follows that

$$
rank(\mathbf{L}) = n - r, \tag{24}
$$

where r is the number of connected components of $\mathscr{M}$ since each connected component has one translational degree of freedom.

We need to solve a full-rank linear system to uniquely restore the Cartesian coordinates. In general, $\mathscr{M}$ is connected, so we can specify the Cartesian coordinates of one vertex

to make the matrix invertible. Specifically, substituting the coordinates of vertex $\mathbf{v}_i$ is equivalent to dropping the ith row and column from $\mathbf{L}$. However, we usually place more than one constraint on spatial positions of $\mathscr{M}$'s vertices and treat the positional constraints in the least-squares sense. Denote $C := \{1, 2, \ldots, m\}$ as the set of indices of the vertices whose spatial positions are known, and we have

$$\mathbf{v}'_j = \mathbf{c}_j, \ j \in C. \tag{25}$$

Then, the Laplacian deformation is formulated as the following linear system:

$$\left(\frac{\mathbf{L}}{\gamma \mathbf{I}_{m\times m} | \ 0}\right)\mathbf{x} = \begin{pmatrix} \boldsymbol{\delta}^{(x)} \\ \gamma \mathbf{c}_{m\times 1} \end{pmatrix}, \tag{26}$$

where the weight $\gamma > 0$ is used to adjust the importance of the positional constraints. Denoting the system matrix in Eq. (26) by $\widetilde{\mathbf{L}}$, we have $\widetilde{\mathbf{L}}\mathbf{x} = \widetilde{\boldsymbol{\delta}}^{(x)}$. The same goes for the other two dimensions ($\mathbf{y}$ and $\mathbf{z}$). The over-determined linear system can be solved by direct methods for moderate-size meshes, or by iterative methods for large-size meshes.

B. Positional Constraints Calculation

The estimation of positional constraints is a key step in Laplacian deformation. The quality of estimated positional constraints has a great impact on the accuracy of the motion capture. In our method, the positional constraint candidates are first estimated by a method based on sparse representation. Then, to improve the accuracy, the positional constraints are selected from the candidates by three spatio-temporal criteria.

- *Candidate Computation Based on Sparse Representation*

The motion of 3D scene is called *scene flow* and can be described by a three-dimensional velocity field. Several methods have been presented for stereo data with energy minimization frameworks to provide dense scene flow [4], [67], [68]. The normal stereo epipolar geometry is assumed, i.e., the disparities are only along the horizontal direction in the left and right images. A few dense scene flow methods have been proposed in multi-camera set-ups [5], [69]. However, the scene flows estimated by these methods present some noise and outliers, which would affect the performance of motion capture. With the rapid development of sparse representation, a remarkable new field, matrix completion [70]–[72], has emerged very recently. This field addresses a board range of significantly practical problems, i.e., the recovery of a data matrix from a nearly minimal set of perhaps corrupted entries. Positional constraint candidate estimation based on scene flow with noise can also be formulated into such a data recovery problem. We propose a new candidate computation method based on matrix completion, which uses the prior of low-rank property of the multi-view scene flow matrix to achieve reliable estimation. The method consists of the following main steps:

(1) Calculate optical flow fields $\{\overrightarrow{o}^c\}_{0\le c\le N-1}$ for all the views. For each view c, a 3D scene flow field $\overrightarrow{f}_c$ is computed by solving a linear system for each surface vertex $\mathbf{v}_i$ which is visible from view c and its neighboring view $c+1$ [69]. For the invisible surface vertices, their 3D scene flow values are set at a large value, e.g., 10000.

(2) Arrange the calculated 3D scene flow field from view c as the cth column of a matrix $\mathbf{M} = [\overrightarrow{f}_0, \ldots, \overrightarrow{f}_{N-1}] \in \mathbb{R}^{N_v\times N}$, where N_v is the number of surface vertices. Note that each column of $\mathbf{M}$ corresponds to the same 3D scene flow field, and ideally should have the same value, which means that the rank of the matrix should be very low and theoretically one. However, there are lots of unknown entries in $\mathbf{M}$, corresponding to the invisible vertices. Therefore, the scene flow estimation can be cast into a matrix completion problem: recovery the low-rank matrix given a fraction of known entries. To recover the complete matrix, the only information available about $\mathbf{M}$ is a sampled set of entries M_{ij}, $(i,j) \in \Omega$, where Ω is a subset of the complete set of entries $[N_v] \times [N]$ ($[n]$ denotes the list $\{1, \ldots, n\}$.). Hence, the problem can be formulated as

$$\begin{aligned} &\text{minimize} && \text{rank}(\mathbf{X}) \\ &\text{subject to} && \mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{M}), \end{aligned} \tag{27}$$

where $\mathbf{X} \in \mathbb{R}^{N_v\times N}$ is the decision variable and $\mathcal{P}_\Omega(\mathbf{X}) : \mathbb{R}^{N_v\times N} \to \mathbb{R}^{N_v\times N}$ is a sampling operator defined by

$$[\mathcal{P}_\Omega(\mathbf{X})]_{ij} = \begin{cases} X_{ij} & (i,j) \in \Omega, \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

However, this rank-minimization problem is NP-hard. It is analogous to the intractability of l_0-minimization in sparse signal recovery. Fortunately, Candès and Recht [70] prove that most low-rank matrices can be perfectly recovered by solving the optimization problem

$$\begin{aligned} &\text{minimize} && \|\mathbf{X}\|_* \\ &\text{subject to} && \mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{M}), \end{aligned} \tag{29}$$

where the functional $\|\mathbf{X}\|_*$ is the nuclear norm of the matrix $\mathbf{X}$, i.e., the sum of singular values. This optimization problem is convex and can be recast as a semidefinite program [73]. Eq. (29) is the tightest convex relaxation of Eq. (27), since the nuclear ball $\{\mathbf{X} \in \mathbb{R}^{N_v\times N} : \|\mathbf{X}\|_* \le 1\}$ is the convex hull of the set of rank-one matrices with their spectral norms bounded by one. We use the singular value thresholding algorithm [74] to solve the nuclear norm minimization problem (29), and obtain the recovered complete matrix $\mathbf{X}$. Then, the average of each row is regarded as the final 3D motion of its corresponding vertex.

(3) The generated 3D flow field $\overrightarrow{f}(\mathbf{v}_i) = (u, v, w)$ represents the displacement by which $\mathbf{v}_i$ at time instant t should move from its current position, and $\mathbf{v}'_i = \mathbf{v}_i + \overrightarrow{f}(\mathbf{v}_i)$ is regarded as a positional constraint candidate at time instant $t+1$.

Our method for candidate computation is robust to noise, which benefits from the matrix completion method derived from sparse representation. In our problem, noise exists in some known entries of the matrix $\mathbf{M}$ due to the errors of the

initial scene flows. Candès et al. [71] show that matrix completion with noise is provably accurate. By matrix completion, not only the unknown entries of the matrix are recovered, but also the noise and outliers existed in the known entries are reduced. As a result, our method for candidate computation is robust to noise, and outperforms the method by using conventional LS (least square)-based scene flow methods, which will be demonstrated by experimental results in Section VII-C.

- *Spatio-Temporal Selection:*

To improve the accuracy of positional constraints, three spatio-temporal criteria are used to select the accurate candidates as final positional constraints. A candidate $\mathbf{v}_i' = \mathbf{v}_i + \overrightarrow{f}(\mathbf{v}_i)$ is considered as a final positional constraint if it has low errors under three spatio-temporal selection criteria. The criteria are defined as follows:

$$C_{sp} = \frac{1}{N}\sum_{c=0}^{N-1}\left(1 - P_{sil}^{c}(\mathbf{v}_i')\right), \tag{30a}$$

$$C_{tmp} = \frac{1}{N_v}\sum_{c\in\mathcal{V}(i)}\left(1 - P_{z}^{c}(p(\mathbf{v}_i), p(\mathbf{v}_i'))\right), \tag{30b}$$

$$C_{smth} = \|\overrightarrow{f}(\mathbf{v}_i) - \frac{1}{N_s}\sum_{j\in\mathcal{N}(i)}\overrightarrow{f}(\mathbf{v}_j)\|. \tag{30c}$$

In formula (30a), $P_{sil}^{c}(\mathbf{v}_i')$ is a function that evaluates to 1 if $\mathbf{v}_i'$ projects inside the silhouette image of camera c at time instant $t+1$, and to 0 otherwise. Hence, the spatial criteria C_{sp} penalizes the candidates that do not project into the silhouettes at all camera views. In formula (30b), $\mathcal{V}(i)$ is the set of visible camera indices for the vertex $\mathbf{v}_i$ and N_v is the number of visible cameras. $P_z^c(p(\mathbf{v}_i), p(\mathbf{v}_i'))$ is a function that calculates the ZNCC (Zero-mean Normalized Cross-Correlation) score between the projection $p(\mathbf{v}_i)$ of the patch centered at $\mathbf{v}_i$ and the projection $p(\mathbf{v}_i')$ of the patch centered at $\mathbf{v}_i'$. The projections of the patches centered at $\mathbf{v}_i$ and $\mathbf{v}_i'$ are on the images of camera c at time instants t and $t+1$, respectively. This temporal criteria C_{tmp} penalizes the candidates whose new positions have low color consistency with their original position in all camera views. In formula (30c), N_s is the number of immediate neighbors of $\mathbf{v}_i$. Considering the existence of outliers, the smooth criteria C_{smth} penalizes the candidates whose motions are not consistent with the average motion of their immediate neighbors.

These three spatio-temporal criteria verify the accuracy of selection from the perspective of spatial and temporal domain and suppress the influence of outliers. A candidate $\mathbf{v}_i'$ will be accepted as a positional constraint if $C_{sp} < TH_{sp}$, $C_{tmp} < TH_{tmp}$ and $C_{smth} < TH_{smth}$, where TH_{sp}, TH_{tmp}, and TH_{smth} are thresholds for the three spatio-temporal selection criteria. These thresholds are set at 1, 0.4, and 0.1, respectively, in experiments. The procedure for spatial-temporal selection is summarized as follows.

(1) Project all the vertices $\mathbf{v}'$ in the candidate set onto all the views and compute the C_{sp} criteria by formula (30a). Remove the candidate if its C_{sp} value is not smaller than TH_{sp}.
(2) For each candidate $\mathbf{v}'$, compute the patch p centered at $\mathbf{v}$ and the patch p' centered at $\mathbf{v}'$. A patch is essentially a local tangent plane approximation of a surface [7]. Its geometry is fully determined by its center, unit normal vector of the vertex and the frontal visible view image $I(p)$. More concretely, a patch is a rectangle in 3D space, one of whose edges is parallel to the x-axis of the current camera. Overlay a $(2\mu+1)\times(2\mu+1)$ grid on the patches p and p'. The size of the rectangle is chosen so that the smallest axis-aligned square in $I(p)$ that contains its image projection is of size $(2\mu+1)\times(2\mu+1)$ pixels.
(3) For each visible view c of one candidate vertex $\mathbf{v}'$, sample the pixel colors $r(p, I_t^c)$ and $r(p', I_{t+1}^c)$ through bilinear interpolation at image projections of all the grid points in images I_t^c and I_{t+1}^c. Compute the C_{tmp} criteria by formula (30b). If this criteria is not smaller than TH_{tmp}, remove the candidate $\mathbf{v}'$.
(4) Compute the smooth criteria C_{smth} by formula (30c) for each retained candidate $\mathbf{v}' = \mathbf{v} + \overrightarrow{f}(\mathbf{v})$ and accept the candidate if its smooth criteria C_{smth} is smaller than TH_{smth}.

C. Iterative Laplacian Deformation

Sorkine et al. [50] propose a surface deformation method based on a simple modeling operation that asks for rigidity of the local transformations. The minimization procedure is guaranteed to not increase energy in each step and the whole algorithm is effective and notably easy to implement. Inspired by this surface-based deformation method, we present an iterative volumetric Laplacian deformation framework. This framework is driven by the above positional constraints with rigidity requirement for the local transformations. Assume that a tetrahedral mesh $\mathscr{M}$ with geometric embedding $\mathbf{v}_i$ is deformed into $\mathscr{M}'$ that has the same connectivity and different geometric embedding $\mathbf{v}_i'$. Initialize rotation matrix $\mathbf{R}_i$ with an identical matrix. The following iterations are performed.

(1) Solve the tetrahedral Laplacian system $\widetilde{\mathbf{L}}\mathbf{v} = \widetilde{\boldsymbol{\delta}}$ with the calculated positional constraints. The ith row of $\widetilde{\boldsymbol{\delta}}$ is

$$\sum_{j\in\mathcal{N}(i)}\frac{\omega_{ij}}{2}(\mathbf{R}_i + \mathbf{R}_j)(\mathbf{v}_i - \mathbf{v}_j),$$

where $i = \{0, 1, \ldots, |V|-1\}$ and $|V|$ is the number of elements in V.
(2) Denote the covariance matrix by

$$\mathbf{C}_i = \sum_{j\in\mathcal{N}(i)}\omega_{ij}(\mathbf{v}_i - \mathbf{v}_j)(\mathbf{v}_i' - \mathbf{v}_j')^T = \mathbf{V}_i\mathbf{D}_i\mathbf{V}_i'^T.$$

We can derive $\mathbf{R}_i$ from the singular value decomposition of $\mathbf{C}_i = \mathbf{U}_i\boldsymbol{\Sigma}_i\mathbf{V}_i^T$:

$$\mathbf{R}_i = \mathbf{V}_i\mathbf{U}_i^T.$$

If $\det(\mathbf{R}_i) \leq 0$, change the sign of the column of $\mathbf{U}_i$ that corresponds to the smallest singular value.
(3) Update $\widetilde{\boldsymbol{\delta}}$ with the new $\mathbf{R}_i$ and return to (1) until the silhouette error for all the views is below a threshold S_{th} or the number of iterations is larger than a given maximum.
(4) Update the mesh $\mathscr{M}$.

Note that there would exist tracking errors to a degree due to the nonideal imaging of multi-view video sequences, such as noise and/or color inconsistency. Since the model at a time instant is predicted from the one at the last time instant, the errors would drift over time. To avoid error propagation, the model is refreshed by the shape recovery method if the tracking error is above a certain level. Since the ground-truth motions are not available for the markerless multi-view sequences, the tracking error is estimated by the silhouette error, i.e., the average of the XORs between the projection of the model and the multi-view silhouette images. Specifically, a 3D model is first obtained by the shape recovery method and then tracked by the iterative volumetric Laplacian deformation. If the silhouette error of the tracked model is larger than a given threshold, the 3D model is refreshed by reconstructing a new one with the shape recovery method. Experimentally, the refresh frequency is dozens of frames for small and moderate-size meshes (tens of thousands of vertices), and about fifteen frames for large-size meshes or high-quality tracking.

VII. Experimental Results

In this section, we evaluate the performances of the proposed shape recovery method (in Section VII-B) and the motion capture method (in Section VII-C) with publicly available datasets and datasets captured by our real multi-camera dome system (introduced in Section VII-A). Running times of our method are reported in Section VII-D.

A. Acquisition System

In addition to the publicly available datasets, we evaluate our method with multi-view videos captured by a multi-camera dome system. As shown in Fig. 3, the multi-camera dome system is equipped with controllable lighting and has a chroma-key background. The diameter of the dome is six meters, which provides enough space for moderate human performances. For the sake of easy segmentation, the steel tubes of the dome are painted green, and the whole dome is covered with a green blanket. For environment illumination, ambient lighting instead of directional spot lighting is used since it provides less self-shadowing. The ceiling is installed with 100 fluorescent light tubes, which are carefully grouped so that the illumination can be flexibly controlled.

Twenty cameras are placed in an approximately circular arrangement on the dome, about 70 cm off the ground. The employed Point Grey $Flea2$ cameras are capable of capturing up to 30 frames per second at 1024×768 resolution. The focal length of the camera lens is 4 mm, which is enough to film a whole body with moderate motions. All the cameras are geometrically and radiometrically calibrated by our automatic calibration methods for large camera arrays [75], [76].

B. Shape Recovery

The proposed shape recovery method consists of the separating and merging steps. In the separating step, the depth maps are generated by applying the proposed variational method to each pair of adjacent views (summarized in Algorithm 1). Then, in the merging step, the generated depth maps are merged into a 3D mesh. The depth maps and the reconstructed complete 3D meshes are presented as follows.

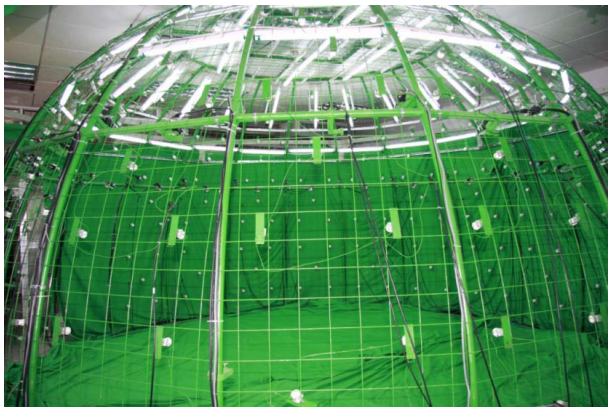


Fig. 3. The multi-camera dome system.

1) Depth Map: The parameters for Algorithm 1 are set as follows: $M = 5, \alpha = 60, \gamma = 7.5, maxOutIter = 20, maxInIter = 30$, and the number of iterations in SOR is 100. We evaluate the performance of the separating step with six datasets captured by our real multi-camera dome system. The datasets are named by their poses: *dancer1*, *dancer2*, *vollyball player*, *walker*, *fighter*, and *prayer*. As an example, the first frames of *dancer1* (20 views) are shown in Fig. 4.

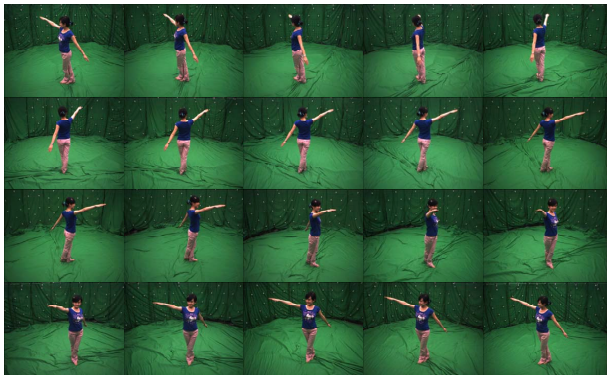


Fig. 4. First frames of *dancer1* dataset captured by our system.

Since the multi-view capturing system has a chroma-key background, silhouettes of the 3D object can be extracted from the multi-view images simply by thresholding. Then, the visual hull obtained from these silhouettes are optimized with the proposed variational model, generating continuous depth maps represented with point clouds. Reconstructed depth maps for *dancer1* and *dancer2* at four viewpoints are shown in Fig. 5 and Fig. 6, respectively. For comparison, the captured real images and the reconstructed results of visual hull at the corresponding viewpoints are also presented. As suggested by the captured images, the depth map reconstructed by our method are much more accurate than the visual hull, such as the regions emphasized by rectangles in the figures. For example, in Fig. 5, the non-convex shapes formed by the interweaving of two legs are more faithfully reconstructed. The

geometries at small scales, such as the shape of the ear, are better recovered. Moreover, our method has stronger capability in reconstructing the non-connected geometries of the objects, e.g., the gap between two lower legs. Similar phenomena can also be observed in Fig. 6. More results for other datasets, i.e., *volleyball player*, *walker*, *fighter*, and *prayer*, are presented in Fig. 7. All these results suggest that our method produces more accurate depth maps than visual hull. This demonstrates the effectiveness of the proposed variational model.

It is worth noting that such fine granularity of reconstructed depth maps are obtained without loss of smoothness. This appealing property attributes to the elegant design of the variational model, which consists of the data and smoothness terms. The data term is derived from the luminance and gradient constancy constraints. Therefore, the data term reflects the fidelity of the reconstructed depth map, and plays an important role in preserving the local geometries. Simultaneously, the smoothness term addresses the smoothness and the completeness of the reconstructed results. By bridging the data and smoothness terms with a penalization parameter and then minimizing it, rich geometric details can be preserved in the reconstructed depth maps while ensuring the smoothness. This is the basis of high quality 3D mesh reconstruction as shown by the following results.

2) 3D Mesh: With the obtained depth maps of all the views, a 3D mesh can be reconstructed in the merging step. Reconstructed 3D meshes are shown in Fig. 8 for *dancer1* dateset. Two other shape recovery methods are compared: the graph cuts method [28] and Furukawa's PMVS (Patch-based Multi-View Stereo) algorithm [7]. The results of PMVS are generated with a software package released by the authors [77]. In our experiments, the parameters for PMVS are set as follows.

- One feature is detected in each 16×16 pixel cell
- One patch is reconstructed in each 2×2 pixel cell
- The threshold on the photometric consistency score: 0.5
- The image window size: 7×7
- The minimum number of visible views for a patch: 3

In Fig. 8, 3D meshes reconstructed by the three methods are shown at ten free viewpoints, including two extreme perspectives given in the first column. The view-independent rendering results at two free viewpoints are also presented. It can be observed that our method provides significantly better results than the other two methods. The 3D meshes reconstructed by the graph cuts method are visually coarse and a little over-smoothed, losing local geometries and presenting unnatural transitions around sharp curvature variations. The 3D meshes recovered by the PMVS method can preserve details to some extent, but cannot ensure completeness. As a result, the whole model is severely distorted around small scale geometries, e.g., the head and arms. On the contrary, our proposed method is free of this problem. The 3D meshes reconstructed by our method are watertight with good visual quality at each viewpoint. The proposed method shows promising performance even for the occlusion regions. The main reason for this is that our method does not impose any constraint on the number of visible views for each surface point, and the occlusion map is also incorporated in the proposed variational model. The view-independent rendering results of the reconstructed meshes are almost visually indistinguishable from the real images, which suggests that the reconstructed 3D meshes are quite accurate.

We also test our method on two of the Middlebury's datasets [78], *dinoSparseRing* (16 views) and *templeSparseRing* (15 views), compared with the PMVS method and the captured images. The results are presented in Fig. 9. It can be observed that the PMVS method can accurately recover local structures, e.g., the steps of the temple, the claws and ridges of the dinosaur. However, the PMVS method does not perform very well in reconstructing smooth regions, e.g., the abdomen of the dinosaur. For these two datasets, our method is comparable to the PMVS method. Further, our method is quantitatively evaluated by the Middlebury's on-line evaluation system on the two datasets. Table I compares quantitative results for nine methods of three categories: 1) methods using the separating and merging strategy (ours and Ref. [8], [79]); 2) global optimization methods in Ref. [28], [29], [32], [80], [81]; 3) the feature-based PMVS method [7]. Three metrics are measured: accuracy, completeness, and normalized time [78]. Compared with the methods that also use the separating and merging strategy, our method significantly outperforms Goesele's method [8] and gives better results than Bradley's method [79] in terms of completeness for the *dinoSparseRing* dataset. As to the global optimization methods, our method shows better performances than the methods in Ref. [28], [29], [32], higher accuracy than Pons' method [80], and less runtime than Zaharescu's method [81]. For the two datasets, the feature-based PMVS method provides the best results in terms of accuracy and completeness, but requires much more ($15\times$) computation than our method. Overall, our method achieves better comprehensive performance in terms of accuracy, completeness and runtime. Please see Ref. [78] for more details about the quantitative evaluation method and the accompanying website for comparison with other methods.

To evaluate the effectiveness as a shape optimization method, we refine a mesh generated by a graph cuts method [29] into a more accurate 3D model. As shown in Fig. 10, the initial input mesh is over-smoothed while more details are generated after optimization by our method, e.g., the ears and the small scale wrinkles on the T-shirt. This demonstrates that our shape recovery method can improve the detailed presentation of input meshes, especially for over-smoothed regions.

3) Discussions: The datasets used in this experiment include four settings in terms of camera numbers: 20 cameras for our captured datasets, 16 cameras for Middlebury's *dinoSparseRing* dataset, 15 cameras for Middlebury's *templeSparseRing* dataset, and 8 cameras for Starck's datasets. Experimental results show that good reconstruction qualities are obtained for these datasets. We test the effect of the camera numbers on the reconstruction qualities with our datasets by evenly removing some cameras. It is found that the reconstruction qualities are still acceptable (comparable to visual hull) when we keep five cameras, which is suggested as the minimum number of cameras for acceptable reconstruction.

It is worth noting that shape recovery methods are developed with their own considerations, and thus have different recon-

TABLE I
QUANTITATIVE EVALUATION OF SHAPE RECOVERY METHODS ON TWO MIDDLEBURY'S DATASETS.

Method	Dataset	Accuracy	Completeness	Normalized Time
Our method	*dinoSR*	0.47mm	97.4%	00:15:15
	templeSR	0.81mm	92.1%	00:11:19
Goesele [8]	*dinoSR*	0.56mm	26%	14:03:12
	templeSR	0.87mm	56.6%	11:26:48
Bradley [79]	*dinoSR*	0.38mm	94.7%	00:07:06
	templeSR	0.48mm	93.7%	00:03:33
Vogiatzis [28]	*dinoSR*	1.18mm	90.8%	00:40:23
	templeSR	2.77mm	79.4%	00:59:01
Starck [29]	*dinoSR*	1.01mm	90.7%	2:57:00
	templeSR	1.27mm	87.7%	2:17:00
Kolev2 [32]	*dinoSR*	0.53mm	98.3%	00:48:46
	templeSR	1.04mm	91.8%	1:00:17
Pons [80]	*dinoSR*	0.71mm	97.7%	00:3:00
	templeSR	0.9mm	95.4%	00:10:00
Zaharescu [81]	*dinoSR*	0.45mm	99.2%	00:20:23
	templeSR	0.78mm	95.8%	00:25:42
Furukawa2 [7]	*dinoSR*	0.42mm	99.2%	3:44:00
	templeSR	0.62mm	99.2%	3:33:20

struction qualities on different datasets. The graph cuts methods mainly focus on the completeness of the reconstructions, and hence generate a little over-smoothed results. Overall, the graph cuts methods are robust to various datasets, but the accuracy is lower than our method. Furukawa's PMVS method is based on features and more emphasizes the details of reconstructions without any smoothing operation. This method recovers more details for Middlebury's datasets. However, for our datasets, the 3D models reconstructed by the PMVS method are severely distorted for low-texture regions, e.g., the head and arms of the human body. Therefore, the PMVS method is sensitive to noises, the richness of textures, and the resolution of the object in multi-view images. By using the separating and merging strategy, our proposed method can simultaneously ensure the accuracy and completeness of reconstructed 3D models, which has been demonstrated by the above experimental results. For the Middlebury's datasets, the nonuniform illumination leads to highlights and shadows in the captured multi-view images, and color inconsistency among different views. The color inconsistency does not conform to the luminance constancy assumption, and the highlights and shadows violate the gradient constancy assumption. Moreover, for the surface with many wrinkles or creases, especially when the captured images are clean and have a high resolution of the object, feature-based methods will work better. Therefore, our method does not show equally high-quality reconstruction, compared with the PMVS method. For our datasets, the ambient lighting in our dome system is uniform, and hence the captured datasets have better color consistency and do not present highlights and shadows. Hence, the proposed method performs quite well on our captured datasets. Overall, our method is versatile for various datasets by simultaneously ensuring the accuracy and completeness of reconstructions.

C. Motion Capture

The 3D model reconstructed by our shape recovery method is represented by a triangle mesh. In order to preserve certain shape properties during deformation, a tetrahedral version of the surface mesh is generated by using the *TetGen* package [82]. With the tetrahedral mesh as the underlying scene representation, positional constraints are first computed by a method based on sparse representation and further selected by three spatio-temporal selection criteria (presented in Section VI-B). Then, an iterative volumetric Laplacian deformation framework (presented in Section VI-C) is driven by these positional constraints to generate the same 3D motions as those recorded in multi-view videos. Four datasets are used to evaluate the proposed motion capture method: *dancer1* (1024×768, 61 frames), *dancer2* (1024×768, 133 frames), *skirt* (1024×768, 133 frames) and Starck's dataset (1920×1080, 500 frames). Qualitative and quantitative evaluations for our motion capture method are presented as follows.

1) Qualitative Evaluation: The motion capture results for *dancer1* dataset are shown in Fig. 11. In the top row, the obtained optical flow results for four viewpoints are printed by arrows on their corresponding images. The associated images at the next time instant are overlayed on the results at 50% opacity to compare the estimated optical flows with the real motion. For better visualization, only the optical flows with their magnitudes larger than one are presented. The regions highlighted by rectangles are enlarged and shown in the associated images for closer observation. It can be seen that most estimated optical flow results are consistent with the real motion. The rendered models with scene flows generated by the method in Ref. [69] and our method are shown in the second row. For better visualization, their magnitudes are magnified twice and only 10% of scene flow (uniformly sampled) are presented. As shown in the figure, the scene flow generated by our method is less noisy, more realistic, and is distributed more uniformly. Lacking the ground truth of 3D models for our dataset, we evaluate the tracking results by two methods. One is to illustrate the overlap error by overlaying the reprojected model (rendered in white) on the real images at 50% opacity. As shown in the third row, the reprojected images of the tracked 3D model at the next time instant almost coincide with the real images (suggested by the small overlap errors). The other is to qualitatively show the positions of 7 selected features tracked across the sequence. In the fourth row, the first two images present the reconstructed meshes at two consecutive time instants; the third image shows the 2D projections of 7 vertex features onto the view image 18 at the first time instant; the fourth image gives the tracking positions of these features at the next time instant. Suggested by the positions of the tracked features, the tracking errors are almost imperceptible by visual inspection. The small overlap errors and the tracked features indicate reliable frame-to-frame tracking performance of the proposed method. This experiment also demonstrates the effectiveness of each component of the proposed motion capture method.

To evaluate motion capture performance on the meshes generated by other methods, we also test our motion capture method on Starck's dataset [29] by using his reconstructed mesh as the underlying scene representation. As shown in Fig. 12, suggested by the small overlap errors and the tracked features, our method successfully tracks the motion of the object for three time instants, especially the variations in hand shape, even when the object wears wide, loose, and low-texture clothes. Although only three frames are tracked, the amount of the motion in this dataset is about four times more intensive than our datasets, which suggests that our method can track objects with high motions.

To evaluate the tracking quality across multiple frames, motion capture results for two longer sequences, *dancer2* and *skirt*, are presented in Fig. 13. The reconstructed mesh for each dataset has more than one hundred thousand vertices. A visual representation of the whole motions over time is shown in the first image for each sequence by putting the meshes at several consecutive time instants together (7 frames for *dancer2* and 20 frames for *skirt*). It can be observed that, via the proposed shape and motion capture method, the 3D model at each time instant is faithfully reconstructed and the motion details are reproduced without any human intervention. The *skirt* sequence is quite challenging since the loose long skirt of the dancing girl is difficult for both shape recovery and motion capture. The promising results on this dataset demonstrate that our method can achieve good shape and motion capture performance even for human body in loose clothes.

2) Quantitative Evaluation: In above experiments, the proposed motion capture method is mainly qualitatively evaluated by visual inspection on the tracking results. To further evaluate our method in a quantitative fashion, we need measure the tracking errors of selected features. For each dataset, seven vertices are selected as features shown in Fig. 11, 12 and 13. The most straightforward way is to calculate the position errors of the selected features between the tracked mesh and the true mesh at the target time instant. However, the ground truth of dynamic meshes are not available. Instead of calculating the 3D position errors of the selected features, we evaluate the tracking errors by measuring the position errors of their 2D projections. Specifically, the features on the mesh are first projected onto a view image at the initial time instant. The correspondences of the projected features at the target time instant are marked by manually adjustment to ensure a high accuracy. Then, the features on the tracked mesh are projected onto the same view image at the target time instant as 2D tracked features. Finally, the position errors between the 2D tracked features and the marked correspondences at the target time instant are taken as the tracking errors.

Table II presents the tracking errors for four datasets, i.e., *dancer1*, *starck*, *dancer2*, and *skirt*. The intervals between the target time instant and initial time instant for these datasets are 1, 2, 6, and 19, respectively. As shown in Table II, exact tracking is achieved for most features of *dancer1* and *dancer 2* that contain relatively low motions. For *starck* with more intensive motions, the average tracking error is within one pixel per feature, which suggests a promising tracking accuracy. Although *skirt* is more challenging due to the loose clothes and larger tracking interval, the tracking errors are within four pixels, which is acceptable for many applications. This also substantiates the discussion in Section VI-C that the mesh should be refreshed periodically to avoid the propagation of tracking errors.

TABLE II
QUANTITATIVE EVALUATION OF THE MOTION CAPTURE METHOD. (TRACKING ERRORS (e_x, e_y) OF SEVEN FEATURES FOR FOUR DATASETS. e_x AND e_y DENOTE THE HORIZONTAL AND VERTICAL TRACKING ERRORS IN PIXELS, RESPECTIVELY.)

Dataset	f0	f1	f2	f3	f4	f5	f6
dancer1	(0, 0)	(0, 0)	(0, 0)	(1, 2)	(0, 0)	(0, 0)	(0, 0)
starck	(0, 0)	(3, 0)	(0, 0)	(0, 0)	(0, 0)	(1, 1)	(0, 3)
dancer2	(4, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)
skirt	(2, 0)	(2, 1)	(4, 1)	(3, 4)	(0, 0)	(2, 0)	(2, 2)

D. Running Times

All the experiments are run on a desktop with Intel Pentium IV 2.8-GHz CPU and 1.0-GB RAM. The proposed shape recovery method requires much less computation than most state-of-the-art methods. The running time is 15min57sec for *dinoSparseRing* dataset (16 views, 640×480), 10min38sec for *templeSparseRing* dataset (15 views, 640×480), and 20min on average for our six datasets (20 views, 1024×768). The separating step (depth maps computation) consumes about 80% of the overall shape recovery computation. As to the motion capture method, the matrix completion for positional-constraint computation is efficiently tackled by the singular value thresholding algorithm (about 5sec for the completion of a 100000×20 rank-one matrix); the required computation of the volumetric deformation is comparable to the positional-constraint computation. The running time of the motion capture method is about 10min for moderate-size meshes (about one hundred thousand vertices for the test datasets).

VIII. CONCLUSION

This paper proposes a new markerless shape and motion capture approach from multi-view video sequences. Both shape recovery and motion capture methods have shown promising results on various datasets.

For shape recovery, the method consists of two steps: the separating step and the merging step. In the separating step, continuous depth maps represented with point clouds are generated by solving a variational model, which is regularized by four constraints to ensure the accuracy and completeness of the reconstruction. In the merging step, the point clouds of all the views are merged together and reconstructed to a 3D mesh using the marching cubes method with silhouette constraints. Experiments on both publicly available and our captured datasets demonstrate that the generated depth maps contain small scale geometric structures without loss of smoothness. As a result, the recovered 3D meshes are watertight with rich surface details. This also demonstrates the effectiveness of the "separating + merging" framework in the shape recovery field.

For motion capture, we propose a volumetric deformation method. First, positional constraints are calculated by a method based on sparse representation, and further selected by three spatio-temporal selection criteria. Then, an iterative volumetric Laplacian deformation framework is driven by these positional constraints to generate the same motions as those recorded in the multi-view videos. The whole process is totally in a automatical mode: identify and track the 3D trajectories of features on a moving object without the need of any priori information, manual intervention, or optical markers. Moreover, our method does not employ a kinematic skeleton and hence can track non-rigid motions. Experimental results demonstrate that this automatic tracking method is accurate, connectivity-preserving, and can track human bodies in loose clothes. These appealing properties attribute to the careful selection of positional constraints and the volumetric deformation framework. With the rapid development of spare representation theory, its application in various fields is arguably of paramount importance these days [83]–[85]. Our success of bringing sparse representation theory into motion capture verifies again the powerful of sparse representation. For the deformation, although keeping mesh connectivity facilitates some applications, e.g., relighting, it is difficult to handle situations in which the mesh topology really changes (e.g., from genus-N to genus-zero) even if the silhouette error checker helps somewhat. How to automatically handle the deformation with varying mesh topology remains a challenging issue, and is to be explored in our future work.

ACKNOWLEDGMENT

The authors are grateful to Dr. B. Wilburn, Dr. S. Lin and Dr. X. Tong for many helpful discussions during K. Li's internship at Microsoft Research Asia, and to Dr. Y. Lai and Dr. J. Yang for their useful suggestions and assistances. The authors thank S. M. Seitz for his help to evaluate the proposed method, and thank the associate editor and anonymous reviewers for their comments, which significantly helped to improve this paper.

REFERENCES

[1] J. Starck, A. Maki, S. Nobuhara, A. Hilton, and T. Matsuyama, "The multiple-camera 3-D production studio," *IEEE Trans. CSVT*, vol. 19, no. 6, pp. 856–869, 2009.

[2] M. Krinidis, N. Nikolaidis, and I. Pitas, "3-D head pose estimation in monocular video sequences using deformable surfaces and radial basis functions," *IEEE Trans. CSVT*, vol. 19, no. 2, pp. 261 – 272, 2009.

[3] T. Matsuyama, X. Wu, T. Takai, and T. Wada, "Real-time dynamic 3-D object shape reconstruction and high-fidelity texture mapping for 3-D video," *IEEE Trans. CSVT*, vol. 14, no. 3, pp. 357– 369, 2004.

[4] F. Huguet and F. Devernay, "A variational method for scene flow estimation from stereo sequences," in *Proc. ICCV*, 2007.

[5] J.-P. Pons, R. Keriven, and O. Faugeras, "Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score," *International Journal of Computer Vision*, vol. 72, no. 2, pp. 179–193, 2007.

[6] G. Vogiatzis, C. Hernández, P. H. S. Torr, and R. Cipolla., "Multi-view stereo via volumetric graph-cuts and occlusion robust photo-consistency," *IEEE Trans. PAMI*, vol. 29, no. 12, pp. 2241–2246, 2007.

[7] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," in *Proc. CVPR*, 2007, pp. 1–8.

[8] M. Goesele, B. Curless, and S. M. Seitz, "Multi-view stereo revisited," in *Proc. CVPR*, vol. 2, 2006, pp. 2402–2409.

[9] J. Starck and A. Hilton, "Surface capture for performance-based animation," *IEEE Computer Graphics and Applications*, vol. 27, no. 3, pp. 21–31, 2007.

[10] A. Menache, *Understanding motion capture for computer animation and video games*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999.

[11] B. Rosenhahn, T. Brox, and H.-P. Seidel, "Scaled motion dynamics for markerless motion capture," in *Proc. CVPR*, 2007.

[12] J. Gall, B. Rosenhahn, and H.-P. Seidel, "Drift-free tracking of rigid and articulated objects," in *Proc. CVPR*, 2008.

[13] J. Gall, C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel, "Motion capture using joint skeleton tracking and surface estimation," in *Proc. CVPR*, 2009.

[14] J. S. Franco and E. Boyer, "Exact polyhedral visual hulls," in *Proc. BMVC*, 1994.

[15] W. E. Lorensen, "Marching cubes: A high resolution 3D surface construction algorithm," in *Proc. ACM SIGGRAPH*, 1987, pp. 163–169.

[16] R. Zhang, P. Tsai, J. E. Cryer, and M. Shah, "Shape from shading: A survey," *IEEE Trans. PAMI*, vol. 21, no. 8, pp. 690–706, 1999.

[17] H. Jin, D. Cremers, D. Wang, A. Yezzi, E. Prados, and S. Soatto, "3-D reconstruction of shaded objects from multiple images under unknown illumination," *International Journal of Computer Vision*, vol. 76, no. 3, pp. 245–256, March 2008.

[18] A. Laurentini, "The visual hull concept for silhouettes-based image understanding," *IEEE Trans. PAMI*, vol. 16, no. 2, pp. 150–162, 1994.

[19] M. Gross, S. Würmlin, M. Naef, E. Lamboray, C. Spagno, A. Kunz, E.Koller-meier, T. Svoboba, L. Goll, S. Lang, K. Strehlke, A. Moere, and O. Staadt, "Blue-c: A spatially immersive display and 3D video portal for telepresence," in *Proc. ACM SIGGRAPH*, 2003.

[20] W. Niem and R. Buschmann, "Automatic modelling of 3D natural objects from multiple images," in *Proc. European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, 1994.

[21] S. B. Kang, R. Szeliski, and J. Chai, "Handling occlusions in dense multi-view stereo," in *Proc. CVPR*, vol. 1, 2001, pp. 103–110.

[22] P. Gargallo and P. Sturm., "Bayesian 3D modeling from images using multiple depth maps," in *Proc. CVPR*, vol. 2, 2005, pp. 885–891.

[23] M. A. Drouin, M. Trudeau, and S. Roy, "Geo-consistency for wide multi-camera stereo," in *Proc. CVPR*, vol. 1, 2005, pp. 351–358.

[24] G. Vogiatzis, P. Torr, S. M. Seitz, and R. Cipolla, "Reconstructing relief surfaces," *Image Vision Comput.*, vol. 26, no. 3, pp. 397–404, 2008.

[25] G. Zeng, S. Paris, L. Quan, and F. Sillion, "Progressive surface reconstruction from images using a local prior," in *Proc. ICCV*, 2005, pp. 1230–1237.

[26] A. Yuille and D. Snow, "Shape and albedo from multiple images using integrability," in *Proc. CVPR*, 1997, pp. 158–164.

[27] N. Joshi and D. Kriegman, "Shape from varying illumination and viewpoint," in *Proc. ICCV*, vol. 2, 2007, pp. 1–7.

[28] G. Vogiatzis, P. Torr, and R. Cipolla, "Multi-view stereo via volumetric graph-cuts," in *Proc. CVPR*, 2005, pp. 391–398.

[29] J. Starck, A. Hilton, and G. Miller, "Volumetric stereo with silhouette and feature constraints," in *Proc. BMVC*, vol. 3, September 2006, p. 1189C1198.

[30] O. Faugeras and R. Keriven, "Variational principles, surface evolution, PDE's, level set methods and the stereo problem," *IEEE Trans. Image Processing*, vol. 7, no. 3, pp. 336–344, 1998.

[31] C. Hernández and F. Schmitt, "Silhouette and stereo fusion for 3D object modeling," *Computer Vision and Image Understanding*, vol. 96, no. 3, pp. 367–392, 2004.

[32] K. Kolev, M. Klodt, T. Brox, and D. Cremers, "Continuous global optimization in multiview 3D reconstruction," *International Journal of Computer Vision*, vol. 84, no. 1, pp. 80–96, August 2009.

[33] K. Kolev and D. Cremers, "Integration of multiview stereo and silhouettes via convex functionals on convex domains," in *Proc. ECCV*, 2008.

[34] C. Hern, E. Esteban, and F. Schmitt, "Silhouette and stereo fusion for 3D object modeling," *Computer Vision and Image Understanding*, vol. 96, pp. 367–392, 2004.

[35] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *ACM SIGGRAPH*, 1996, pp. 303–312.

[36] S. I. Park and J. K. Hodgins, "Capturing and animating skin deformation in human motion," *ACM Trans. Graphics*, vol. 25, no. 3, pp. 881–889, 2006.

[37] V. Scholz, T. Stich, M. Magnor, M. Keckeisen, and M. Wacker, "Garment motion capture using color-coded patterns," in *Proc. Eurographics*, vol. 24, no. 3, 2005, pp. 439–448.

[38] R. White, K. Crane, and D. A. Forsyth, "Capturing and animating occluded cloth," in *Proc. ACM SIGGRAPH*, 2007, pp. 34–41.

[39] A. B. Ashraf, S. Lucey, J. F. Cohn, T. Chen, K. M. Prkachin, and P. E. Solomon, "The painful face - pain expression recognition using active

appearance models," in *Proc. ACM Int. Conf. Multimodal Interfaces*, 2007.

[40] A. B. Ashraf, S. Lucey, J. F. Cohn, T. Chen, Z. Ambadar, K. M. Prkachin, and P. E. Solomon, "The painful face II - pain expression recognition using active appearance models," *International Journal of Image and Vision Computing*, vol. 27, no. 12, pp. 1788–1796, 2009.

[41] H.-C. Lee and G.-T. Hur, "3D face deformation using control points and vector muscles," *International Journal of Computer Science and Network Security*, vol. 7, no. 4, pp. 149–155, 2007.

[42] P. Sand, L. McMillan, and J. Popović, "Continuous capture of skin deformation," in *Proc. ACM SIGGRAPH*, 2003, pp. 578–586.

[43] B. Allen, B. Curless, and Z. Popović, "Articulated body deformation from range scan data," in *Proc. ACM SIGGRAPH*, 2002, pp. 612–619.

[44] B. Rosenhahn, U. Kersting, K. Powell, and H.-P. Seidel, "Cloth X-ray: Mocap of people wearing textiles," in *Proc. Pattern Recognition : 28th DAGM Symposium*, vol. 4174, 2006, pp. 495–504.

[45] T. Brox, B. Rosenhahn, J. Gall, and D. Cremers, "Combined region and motion-based 3D tracking of rigid and articulated objects," *IEEE Trans. PAMI*, vol. 32, no. 3, pp. 402–415, 2010.

[46] D. Vlasic, I. Baran, W. Matusik, and J. Popović, "Articulated mesh animation from multi-view silhouettes," *ACM Trans. Graphics*, vol. 27, no. 3, pp. 1–9, 2008.

[47] L. Sigal and M. J. Black, "Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion," CS-06-08, Brown University, Tech. Rep., 2006.

[48] M. Botsch and O. Sorkine, "On linear variational surface deformation methods," *IEEE Trans. VCG*, vol. 14, no. 1, pp. 213–230, 2008.

[49] E. de Aguiar, C. Theobalt, C. Stoll, and H. P. Seidel, "Marker-less deformable mesh tracking for human shape and motion capture," in *Proc. CVPR*, 2007, pp. 1–8.

[50] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in *Proc. Eurographics Symposium on Geometry Processing*, 2007.

[51] W. Xu, K. Zhou, Y. Yu, Q. Tan, Q. Peng, and B. Guo, "Gradient domain editing of deforming mesh sequences," in *Proc. ACM SIGGRAPH*, 2007, pp. 84–93.

[52] R. W. Sumner and J. Popović, "Deformation transfer for triangle meshes," in *Proc. ACM SIGGRAPH*, 2004, pp. 399–405.

[53] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H. P. Seidel, and S. Thrun, "Performance capture from sparse multi-view video," in *ACM SIGGRAPH*, 2008, pp. 1–10.

[54] C. Stoll, E. de Aguiar, C. Theobalt, and H.-P. Seidel, "A volumetric approach to interactive shape editing," MPI-I-2007-4-004, MPI Institute, Tech. Rep., 2007.

[55] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artifical Intelligence*, vol. 17, pp. 185–203, 1981.

[56] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proc. ECCV*, vol. 3024, 2004, pp. 25–36.

[57] Z. Christopher, P. Thomas, and B. Horst, "A duality based approach for realtime TV-L1 optical flow," in *Proc. DAGM Symposium on Pattern Recognition*, 2007.

[58] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr, "Variational optical flow computation in real-time," *IEEE Trans. Image Processing*, vol. 14, no. 5, pp. 608–615, 2005.

[59] A. Blake and A. Zisserman, *Visual reconstruction*. MIT Press, Cambridge, MA, 1987.

[60] M. Lefébure and L. D. Cohen, "Image registration, optical flow and local rigidity," *Journal of Mathematical Imaging and Vision*, vol. 14, no. 2, pp. 131–147, 2001.

[61] T. Ju, F. Lossaso, S. Schaefer, and J. Warren, "Dual contouring of hermite data," in *ACM SIGGRAPH*, 2002, pp. 339–346.

[62] H. Si and K. Gaertner, "Meshing piecewise linear complexes by constrained delaunay tetrahedralizations," in *Proc. International Meshing Roundtable*, 2005, pp. 147–163.

[63] H. Si, "On refinement of constrained delaunay tetrahedralizations," in *Proc. International Meshing Roundtable*, 2006, pp. 509–528.

[64] O. Sorkine, "Laplacian mesh processing," in *Proc. Eurographics State-of-the-Art Report*, 2005.

[65] O. Sorkine, D. Cohen-Or, and S. Toledo, "High-pass quantization for mesh encoding," in *Proc. Eurographics Symposium on Geometry Processing*, 2003, pp. 42–51.

[66] M. Alexa, "Differential coordinates for local mesh morphing and deformation," *The Visual Computer*, vol. 19, no. 2, pp. 105–114, May 2003.

[67] R. Li and S. Sclaroff, "Multi-scale 3D scene flow from binocular stereo sequences," *Computer Vision and Image Understanding*, vol. 110, no. 1, pp. 75–90, 2008.

[68] A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. Franke, and D. Cremers, "Efficient dense scene flow from sparse or dense stereo data," in *Proc. ECCV*, 2008.

[69] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," *IEEE Trans. PAMI*, vol. 27, no. 3, pp. 475–480, 2005.

[70] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[71] E. J. Candès and Y. Plan, "Matrix completion with noise," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 925–936, 2010.

[72] A. Eriksson and A. van den Hengel, "Efficient computation of robust low-rank matrix approximations in the presence of missing data using the L_1 norm," in *Proc. CVPR*, 2010.

[73] M. Fazel, H. Hindi, and S. Boyd, "Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices," in *Proc. American Control Conference*, 2003.

[74] E. J. Candès and Y. Plan, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[75] K. Li, Q. Dai, and W. Xu, "A system and a geometric calibration method for circular camera arrays," *China Patent*, no. 200710121782.4, September 2007.

[76] ——, "High quality color calibration for multi-camera systems with an omnidirectional color checker," in *Proc. ICASSP*, 2010.

[77] Y. Furukawa, "PMVS: Patch-based multi-view stereo software," [Online]. Available: http://www.cs.washington.edu/homes/furukawa/research/pmvs/index.html.

[78] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *Proc. CVPR*, 2006, pp. 519–528.

[79] D. Bradley, T. Boubekeur, and W. Heidrich, "Accurate multi-view reconstruction using robust binocular stereo and surface meshing," in *Proc. CVPR*, 2008.

[80] J. P. Pons, R. Keriven, and O. Faugeras, "Modelling dynamic scenes by registering multi-view image sequences," in *Proc. CVPR*, 2005, pp. 822–827.

[81] A. Zaharescu, E. Boyer, and R. Horaud1, "Transformesh : A topology-adaptive mesh-based approach to surface evolution," in *Proc. ACCV*, 2007.

[82] H. Si, "A quality tetrahedral mesh generator and three-dimensional delaunay triangulator," Weierstrauss Institute for Applied Analysis and Stochastics, Tech. Rep. 9, 2004.

[83] Y. Deng, Y. Liu, Q. Dai, and Z. Zhang, "Noisy depth maps fusion for multi-view stereo via matrix completion," *submitted to IEEE Trans. PAMI*, 2010.

[84] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. PAMI*, vol. 31, no. 2, pp. 210–227, 2009.

[85] P. Sen and S. Darabi, "Compressive dual photography," *Computer Graphics Forum*, vol. 28, no. 2, pp. 609–618, 2009.

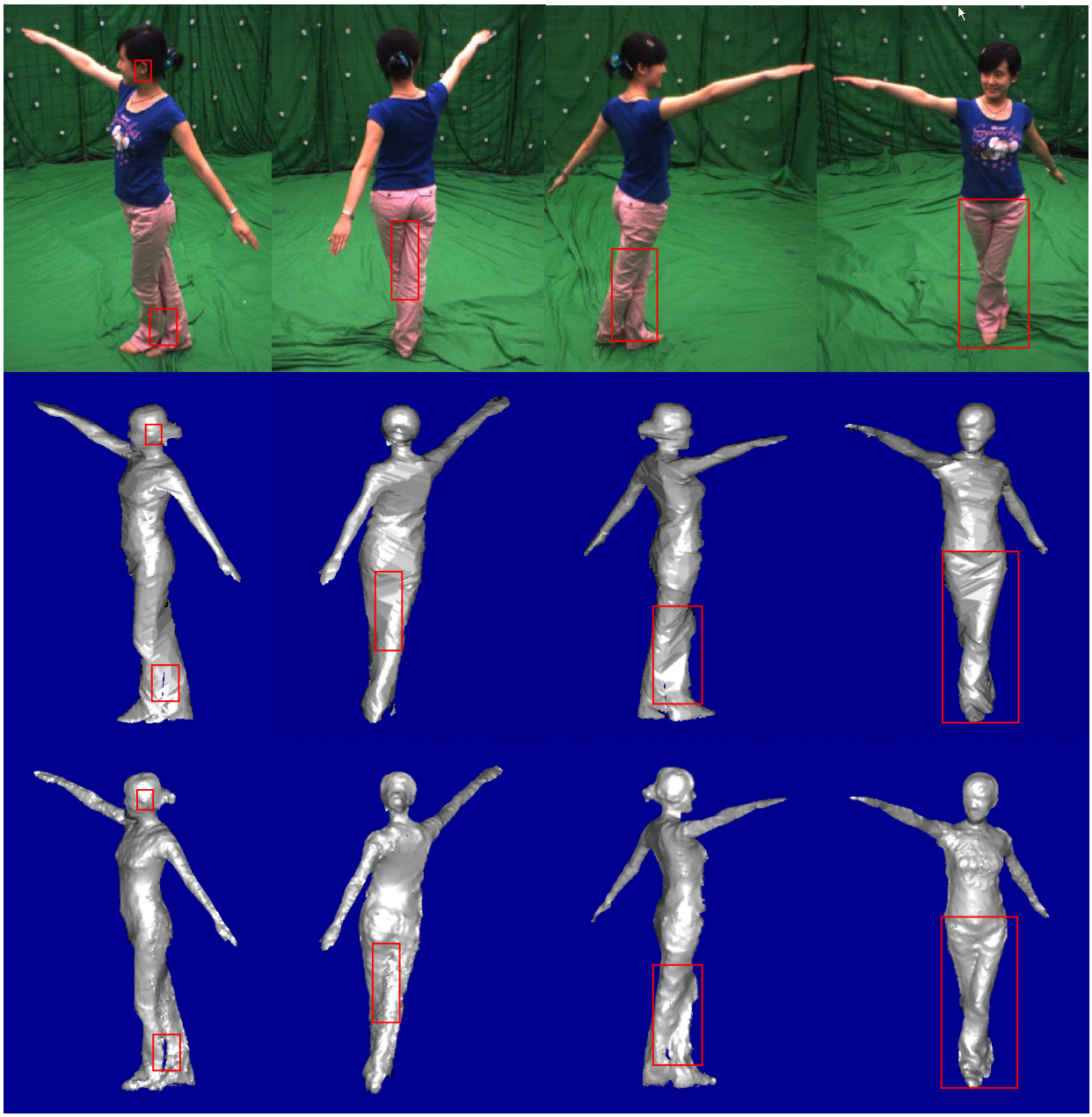


Fig. 5. Reconstructed depth maps of the *dancer1* dataset. Top row: four sample images of the dataset. Middle row: visual hull obtained from silhouettes. Bottom row: our reconstructed results in the separating step.

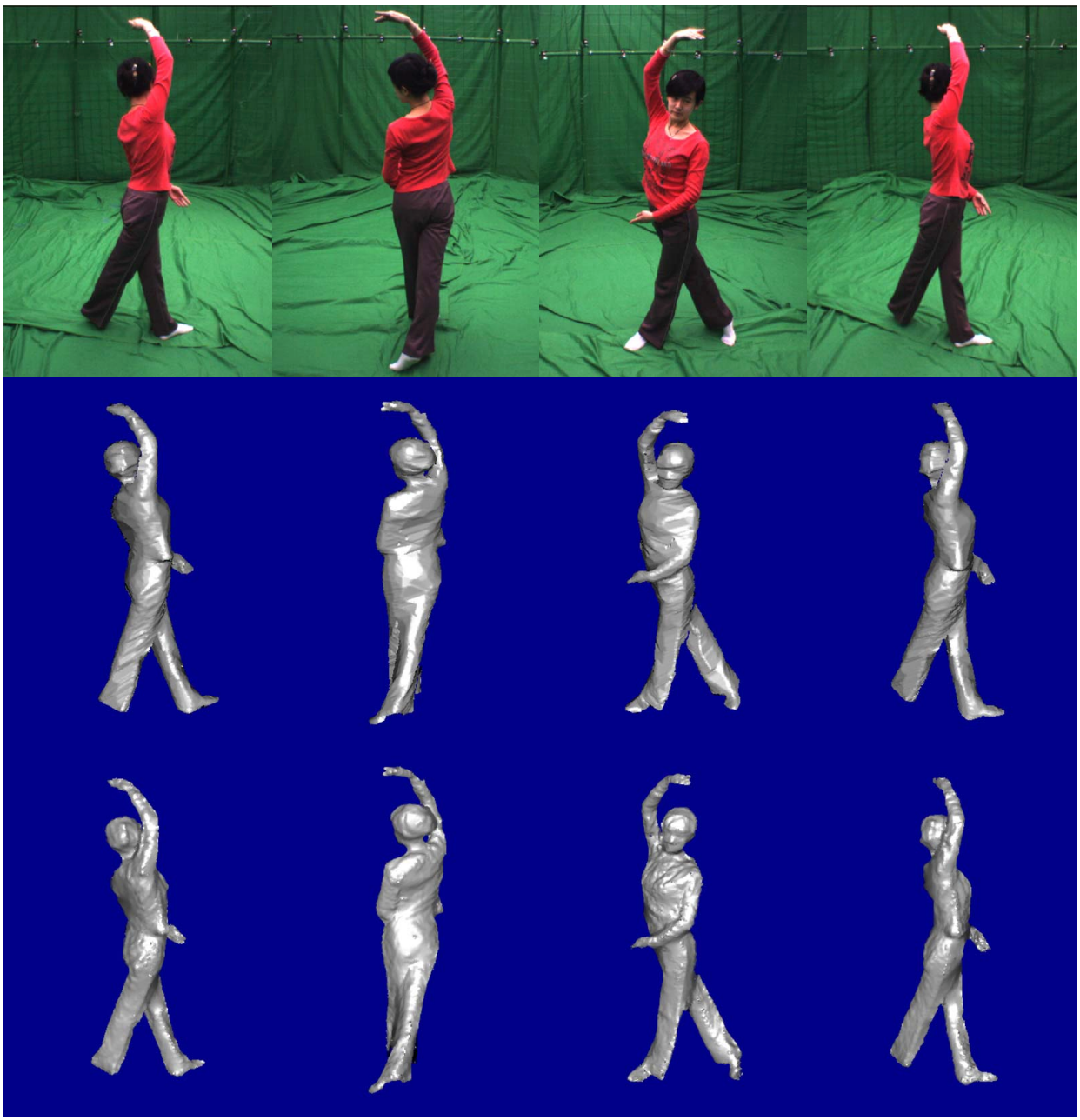
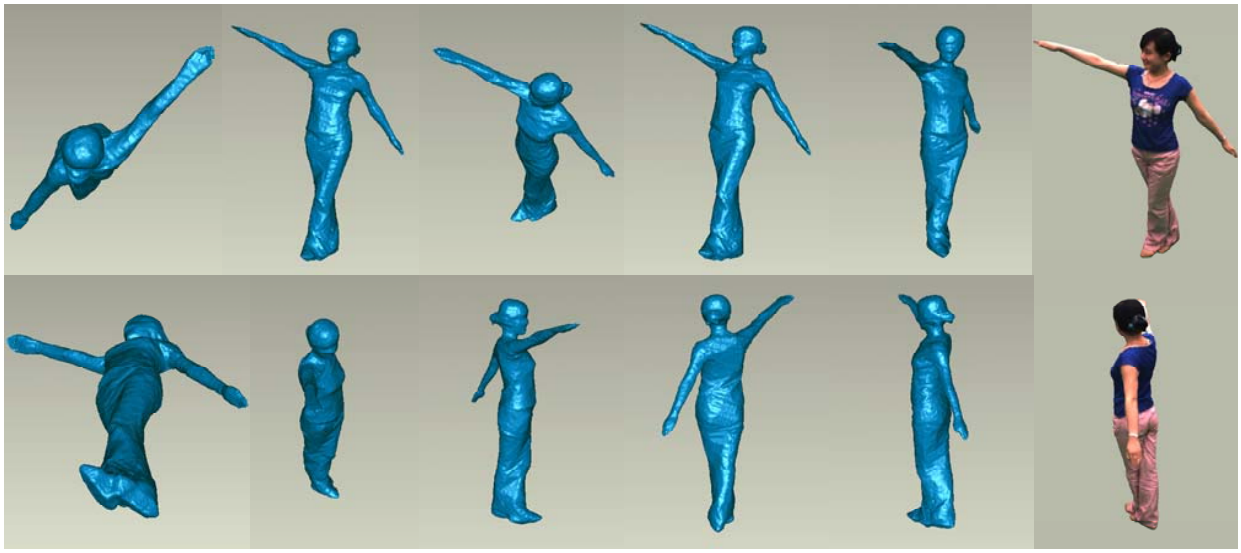
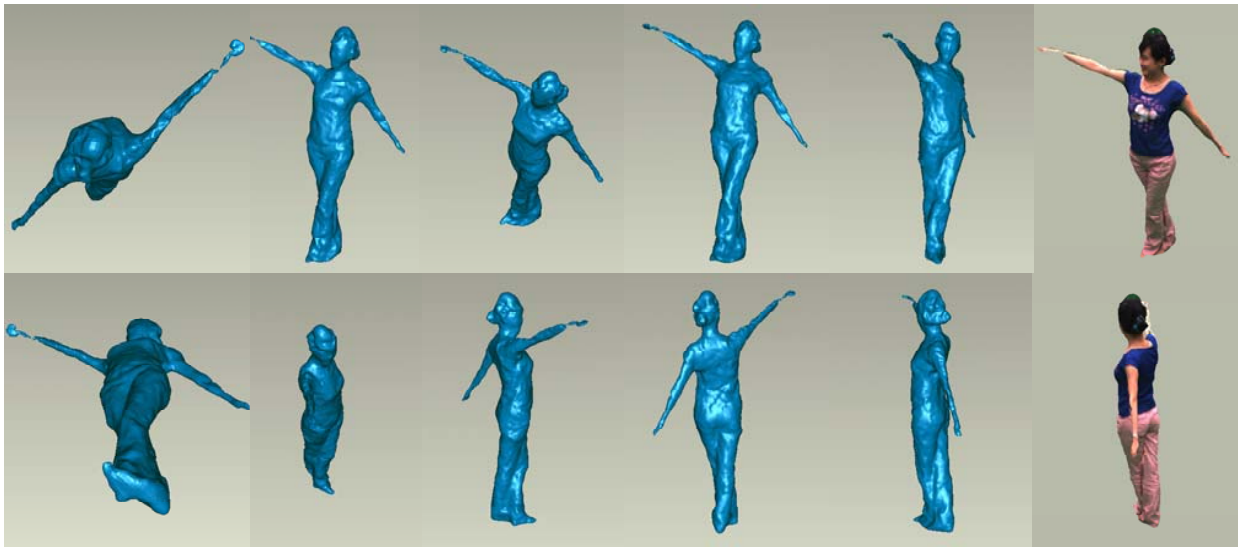


Fig. 6. Reconstructed depth maps of the *dancer2* dataset. Top row: four sample images of the dataset. Middle row: visual hull obtained from silhouettes. Bottom row: our reconstructed results in the separating step.

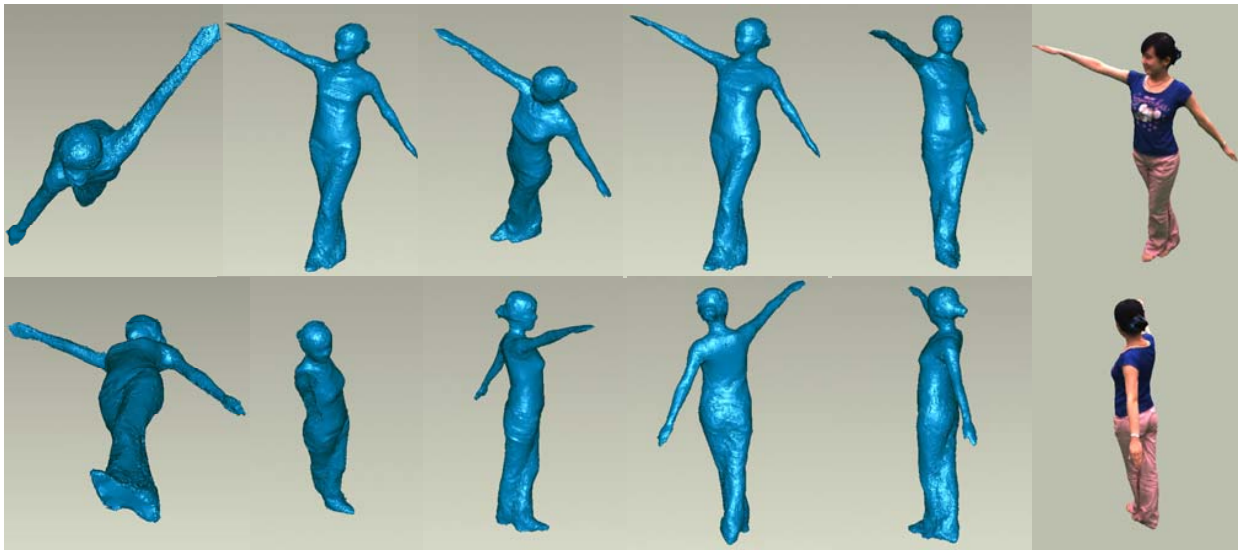
Fig. 7. Reconstructed depth maps of four datasets. Top row: four sample images of these datasets. Middle row: visual hull obtained from silhouettes. Bottom row: our reconstructed results in the separating step.



(a)



(b)



(c)

Fig. 8. 3D mesh for *dancer1* reconstructed by (a) graph cuts, (b) PMVS, and (c) our method. Within each subfigure, the 3D mesh are presented at ten free viewpoints, and view-independent rendering results at two free viewpoints are also presented at the most right column.

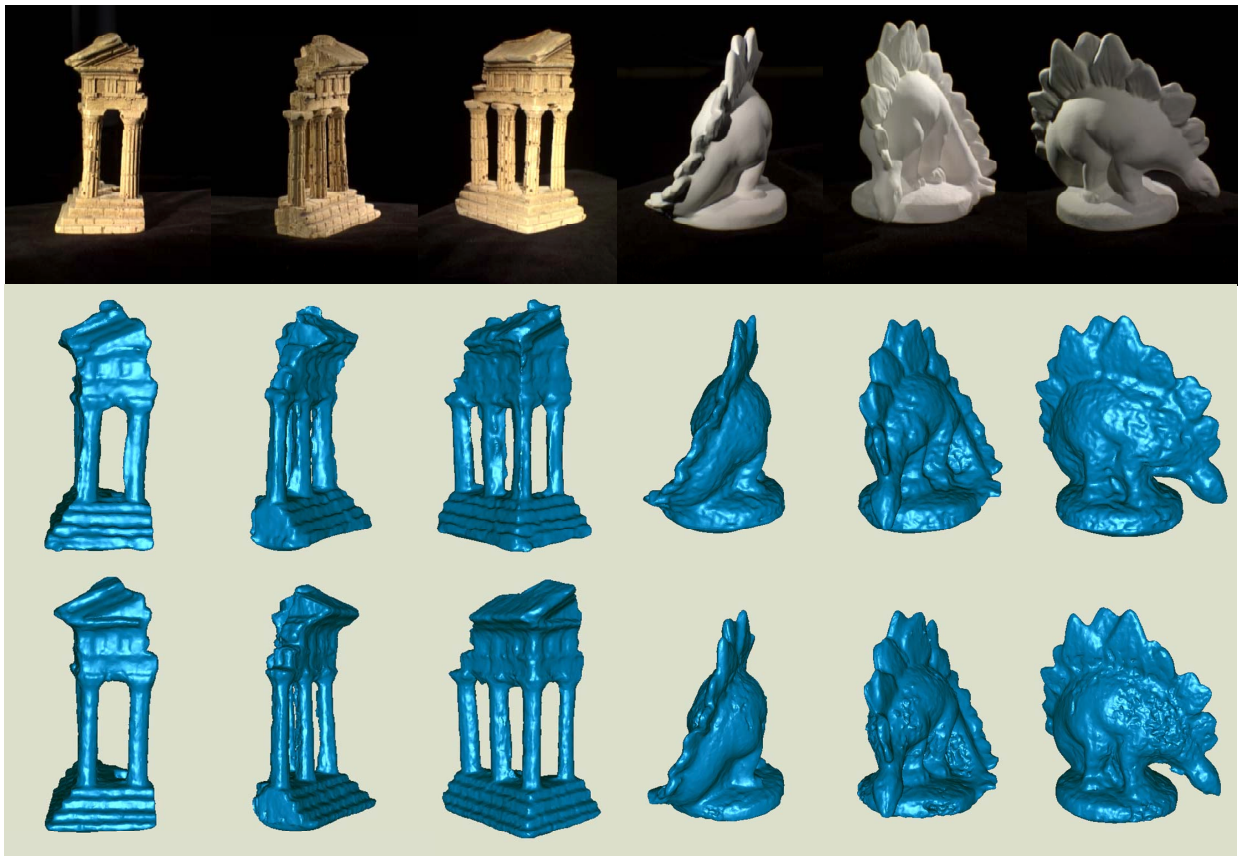


Fig. 9. Reconstructed 3D mesh of Middlebury's datasets. Top row: The real captured images (ground truth); Middle row: 3D meshes reconstructed by PMVS; Bottom row: 3D meshes reconstructed by our proposed method.

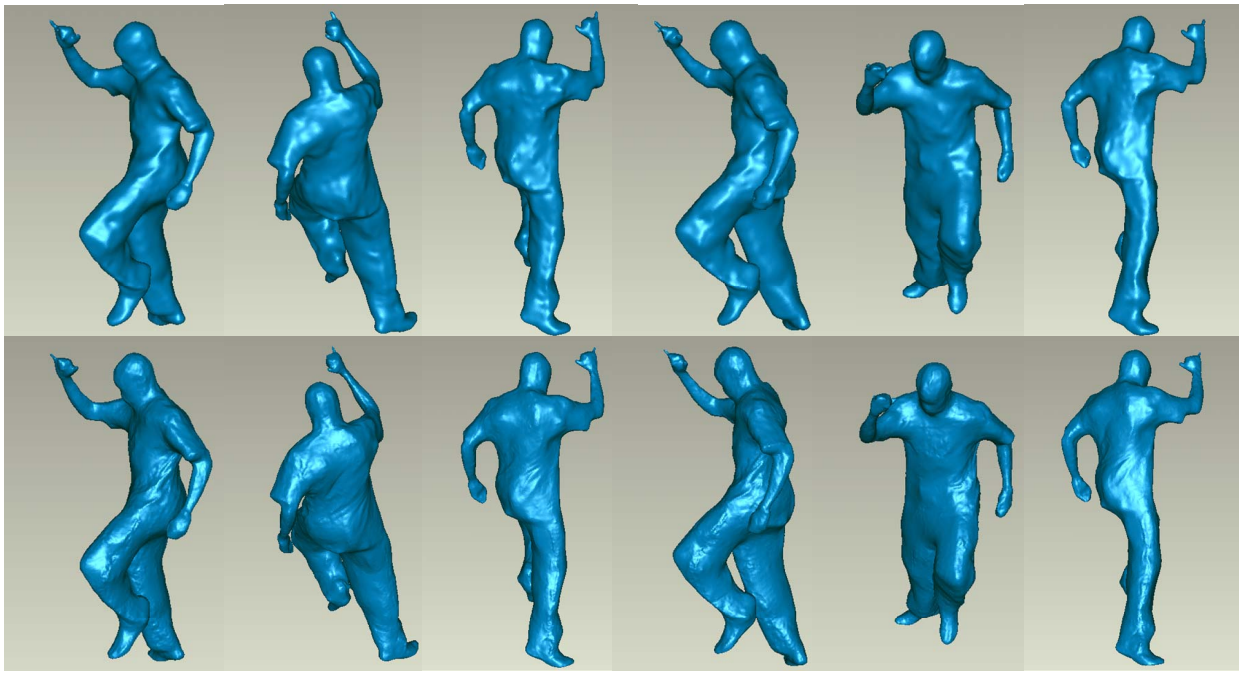


Fig. 10. Shape optimization results on Starck's datasets. Top row: The initial 3D meshes reconstructed by a graph cuts method [29]; Bottom row: 3D meshes optimized by our proposed method.

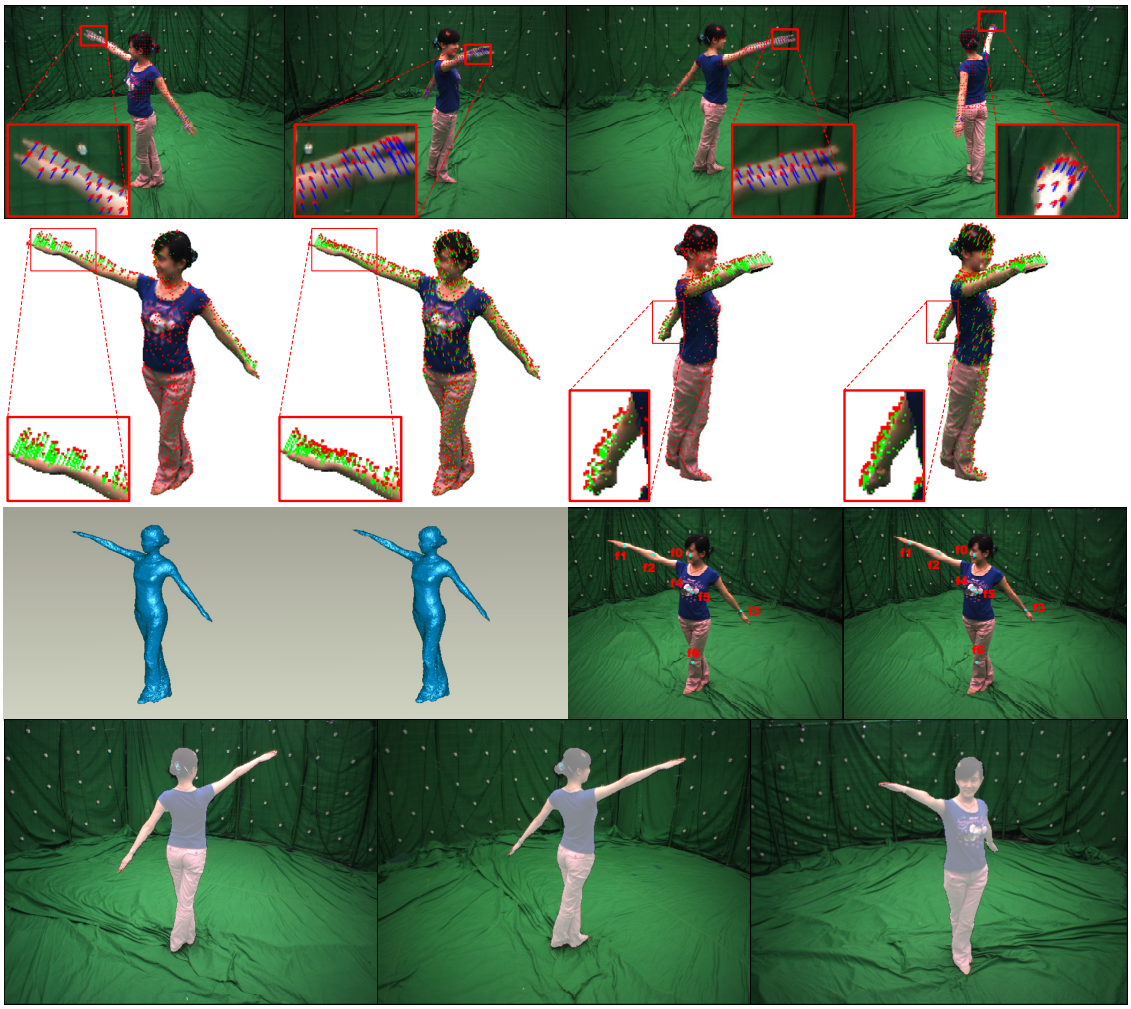


Fig. 11. Tracking results for *dancer1* dataset. Top row: Optical flow results for four viewpoints with enlarged version of regions highlighted by rectangles; Second row: Rendered models with scene flows (magnified twice, 10% uniformly sampled) generated by the method in Ref. [69] (the first and third images) and our proposed method (the second and fourth images); Third row: The mesh before deformation (the first image) and after deformation (the second image), and the tracked features at two frames (the third and fourth images); Bottom row: the overlaps between the reprojected model and three view images.

Fig. 12. Tracking results for Starck's dataset. First column: Rendered models with scene flows (magnified twice, 10% uniformly sampled) generated by the method in Ref. [69] (top) and our method (bottom); Second column: Tracked features at frame 24 and frame 26; The remainder: Top: Optical flow results for two views at frame 25 and frame 26; Bottom: Overlay of the reprojected model (rendered by white) on the real images at 50% opacity.

Fig. 13. Tracking results for *dancer2* (first two rows) and *skirt* (last three rows) datasets. Top row: Meshes for 7 frames of *dancer2* are put together in the first image and are shown separately in the following images; Second row: Tracked features across 7 frames for *dancer2*; The third and fourth rows: Meshes for 20 frames of *skirt* are put together in the first image, and the first 15 meshes are shown separately in the following images; Bottom row: Tracked features across 20 frames for *skirt* (shown at every five time instants for space reason).