Frontiers of Information Technology & Electronic Engineering
www.zju.edu.cn/jzus; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

# A multiscale-contour-based interpolation framework for generating a time-varying quasi-dense point cloud sequence[*]

Chu-hua HUANG[1,2], Dong-ming LU[1], Chang-yu DIAO[†‡3]
([1]*College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*)
([2]*College of Computer Science and Technology, Guizhou University, Guiyang 550025, China*)
([3]*Academy of Cultural Heritage, Zhejiang University, Hangzhou 310058, China*)
[†]E-mail: diaochangyu@gmail.com

Received Sept. 28, 2015; Revision accepted Feb. 21, 2016; Crosschecked Apr. 18, 2016

**Abstract:** To speed up the reconstruction of 3D dynamic scenes in an ordinary hardware platform, we propose an efficient framework to reconstruct 3D dynamic objects using a multiscale-contour-based interpolation from multi-view videos. Our framework takes full advantage of spatio-temporal-contour consistency. It exploits the property to interpolate single contours, two neighboring contours which belong to the same model, and two contours which belong to the same view at different times, corresponding to point-, contour-, and model-level interpolations, respectively. The framework formulates the interpolation of two models as point cloud transport rather than non-rigid surface deformation. Our framework speeds up the reconstruction of a dynamic scene while improving the accuracy of point-pairing which is used to perform the interpolation. We obtain a higher frame rate, spatio-temporal-coherence, and a quasi-dense point cloud sequence with color information. Experiments with real data were conducted to test the efficiency of the framework.

**Key words:** Multi-view video, Free-viewpoint video, Point-pair, Multiscale-contour-based interpolation, Spatio-temporal-contour, Consistency, Time-varying point cloud sequence
http://dx.doi.org/10.1631/FITEE.1500316 **CLC number:** TP391.4

## 1 Introduction

In the past decade, computer graphics and computer vision in many areas have been combined to produce a large number of useful technologies. Free-viewpoint video (FVV) is one such technology. With FVV, a viewer can observe a dynamic scene from any angle. It is a real media, recording dynamic visual events in the real world. In other words, it can record an object with full 3D shape, motion, and surface properties (i.e., color and texture) (Matsuyama *et al.*, 2004).

The reconstruction of 3D dynamic scenes is a key technology to obtain FVV. Reconstructing dynamic scenes which contain moving or deformable objects is essential in various applications including mechanical analysis, virtual reality, computer vision, and computer graphics. In the past, researchers in this field have focused mainly on static and rigid object motions (Matusik *et al.*, 2000; Franco and Boyer, 2009; Furukawa and Ponce, 2009; 2010; Raeesi N. and Wu, 2010; Xia *et al.*, 2011). Recently, with the development of acquisition techniques, 3D reconstruction of real dynamic scenes has been attracting more research effort (Matsuyama *et al.*, 2004; Hasler *et al.*, 2009; Vlasic *et al.*, 2009; Liu *et al.*, 2010; Li *et al.*, 2011; Taneja *et al.*, 2011; Zhang *et al.*, 2011; Bilir

[‡] Corresponding author

[*] Project supported by the National Basic Research Program (973) of China (No. 2012CB725305), the Natural Science Foundation of Guizhou Province, China (No. 20132094), the National Social Science Fund, China (No. 12&ZD32), and the Introducing Talents Foundation of Guizhou University, China (No. 2012028)
ORCID: Chu-hua HUANG, http://orcid.org/0000-0003-3507-4256

and Yemez, 2012; Nakajima *et al.*, 2012; Nakazawa *et al.*, 2012). In particular, tracking for human shape and motion capture has become an important research area (Hofmann and Davrila, 2009; Kanaujia *et al.*, 2011; Huang *et al.*, 2014b; Allain *et al.*, 2015).

In addition, FVV with a high frame rate is preferable in various domains such as entertainment (e.g., 3D games and 3DTV), education (e.g., 3D animal picture books), sports (e.g., sports performance analysis), medicine (e.g., 3D surgery monitoring), and cultural heritage (e.g., 3D archiving traditional dance) (Matsuyama *et al*., 2004; Wang and Yu, 2012; Ahmed and Junejo, 2013).

Speed is more important than accuracy during the reconstruction of a dynamic scene. Generally, reconstruction is not highly dependent on the accuracy of the resulting model sequence. Considering factors such as reconstruction speed, reconstruction accuracy, and the advantage of modeling from multi-view video (MVV), the shape-from-silhouette (SFS) approach is a good choice in cases where the scene includes a moving object or few arranged cameras (Cheung *et al.*, 2003; Matsuyama *et al.*, 2004; Díaz-Más *et al.*, 2010; Haro, 2012; Perez *et al.*, 2012).

In this paper, we propose an efficient framework to reconstruct 3D dynamic objects using a multiscale-contour-based interpolation from MVVs. The framework focuses on obtaining a higher frame rate point cloud sequence to generate FVV. Major contributions of this paper are twofold:

1. Improving efficiency by using a multiscale-contour-based framework: We have integrated multiscale spatio-temporal-contour consistency into a framework in three stages: interpolating single contours, interpolating two spatial contours which belong to the same model, and interpolating two spatial contours which belong to different neighboring models.

2. Improving the accuracy of point-pairs: Here, the term ‘point-pair’ refers to two corresponding points. If the exactness of a point-pair between two 3D points that we have interpolated has low accuracy, it will lead to more outliers and increase the reconstruction time. We search for the corresponding point for each 3D point, and then improve the exactness of the point-pair to obtain a point cloud with higher accuracy.

## 2 Related work

The construction of dynamic scenes usually involves producing model sequences to represent the motion, shape, and appearance of a dynamic object over time. One of the key problems in constructing dynamic scenes is the real-time reconstruction of dynamic objects. In comparison with existing modeling approaches, such as structure from motion, shape from shading, space carving, shape from defocus, shadow carving, multi-view stereo, and structured light, SFS has been considered a most effective approach for real-time and dynamic model sequences (Franco and Boyer, 2009).

SFS is an approach for obtaining a visual hull, which is the upper bound to the actual volume of an object (Baumgart, 1974; Laurentini, 1994). Since reconstruction only from silhouettes cannot recover a concave surface, a visual hull is not the real shape but only the maximal approximation of the object contour. However, SFS can rapidly acquire a complete description of objects and provide a good initial estimate for various complex surface reconstruction algorithms (Xia *et al.*, 2011). So, SFS is very popular in computer vision for its simplicity and high computational efficiency. SFS approaches can be separated into two categories (Franco and Boyer, 2009): volume-based and polyhedral-based. Volume-based approaches generate the final reconstruction result, which is a volumetric representation; polyhedral-based approaches generate the final object’s visual hull, which is computed as the intersection of silhouette cones. Volume-based approaches are less exact and focus on the volume of the visual hull, while polyhedral-based approaches are less numerically stable and time-consuming, and aim to estimate a surface representation of the visual hull (Franco and Boyer, 2009; Zhang *et al.*, 2011; Kim and Dahyot, 2012).

There has been a great amount of literature on SFS approaches. To speed up the approaches, researchers have proposed many methods, which roughly fall into two categories: (1) improving existing methods and (2) implementing existing methods in parallel.

1. Improving existing methods

Cheung *et al.* (2000) proposed a camera system for robust 3D voxel reconstruction using five cameras.

The volume is divided into 64×64×64 voxels. Without displaying the reconstructed 3D voxels, the system obtains a frame rate of about 16 frames/s. Wu *et al*. (2006) proposed an approach using a plane intersection test for the reconstruction of a moving object, obtaining a frame rate of 12 frames/s with nine cameras. Arita and Taniguchi (2001) proposed a system with a resolution of 100×100×100 voxels and a frame rate of 14 frames/s. Borovikov *et al*. (2003) proposed a voxel-based reconstruction method that obtains a frame rate of about 10 frames/s and a volume resolution of 64×64×64 by using 14 cameras.

Franco and Boyer (2009) proposed a robust approach to reconstructing a visual hull. The method is a type of polyhedral-based approach. They obtained real sequences acquired on the GrImage platform. In their experiment, all 800 models generated in a 27 s sequence, in which the resolution of the images was 2000×1500, were verified to be manifold watertight polyhedral surfaces. The average computation time was 0.7 s per sequence time step, as processed by a 3 GHz CPU with 3 GB RAM. However, the running time includes the reconstruction of the shape but not texture mapping. To obtain a good visual appearance, the running time must increase.

None of these methods, however, can meet the requirements of real-time reconstruction. The frame rate does not reach the required 30 frames/s and the model quality is not satisfactory, especially in the case of multi-camera videos or higher resolution videos.

2. Implementing existing methods in parallel

Over the last 10 years, researchers have also focused on accelerating 3D reconstruction using cluster systems or GPU (graphics processing unit). Matsuyama *et al*. (2004) introduced a plane-based volume intersection algorithm to realize the real-time reconstruction of a dynamic scene. They parallelized the algorithm using the cluster system. Duckworth and Roberts (2011) adopted OpenCL to implement the algorithm proposed by Franco and Boyer (2009). They proposed an approach which reconstructs the visual hull from multiple video streams in real time. The reconstruction speed is faster in low resolution cases. Results from their experiment indicated that for low resolution and low camera counts, the CPU algorithm can be implemented more quickly. Perez *et al*. (2012) presented several algorithmic improvements for visual hull reconstruction using a voxel-based approach that reduces resource consumption. They adopted FPGA and GPU for 3D reconstruction. Their approach allows a 256×256×128 reconstruction volume to be obtained in only 33.55 ms in an FPGA. Their experiment showed that this approach can reconstruct a dynamic scene. Hauswiesner *et al*. (2012) rendered an image-based visual hull using multi-GPU. Most approaches using GPU or cluster systems are difficult to implement.

In short, the above-mentioned methods suffer from the problem that the model sequence has a lower frame rate when it is required to meet certain requirements, especially in an ordinary hardware platform. Although many researchers have focused on the problem, it is still difficult to resolve, especially in the case of multi-camera video or high resolution video (such as 5616×3744).

Based on the strengths and drawbacks of the existing SFS methods, we propose a multiscale-contour-based interpolation framework to generate a time-varying quasi-dense point cloud sequence.

## 3 Multiscale-contour-based interpolation framework

### 3.1 Overview

Here, we denote the term 'high confidence points' as points that constitute the sparse point cloud and 'expansion points' as points that are obtained by interpolating high confidence points. In our framework (Fig. 1), we formulate the interpolation as point cloud transport rather than non-rigid surface deformation. The point cloud in between the discrete frames is obtained by interpolating the neighboring models.

First, we interpolate a single contour, based on point level; second, we expand a sparse point cloud using spatio-contour consistency and then obtain a quasi-dense point cloud sequence with color information; third, we interpolate the quasi-dense point cloud sequence using temporal-contour consistency (Fig. 1). Details of each of these steps are given in the following subsections.

### 3.2 Point-level interpolation

In this subsection, we interpolate single contours by point-level interpolation. Consider a set of cameras

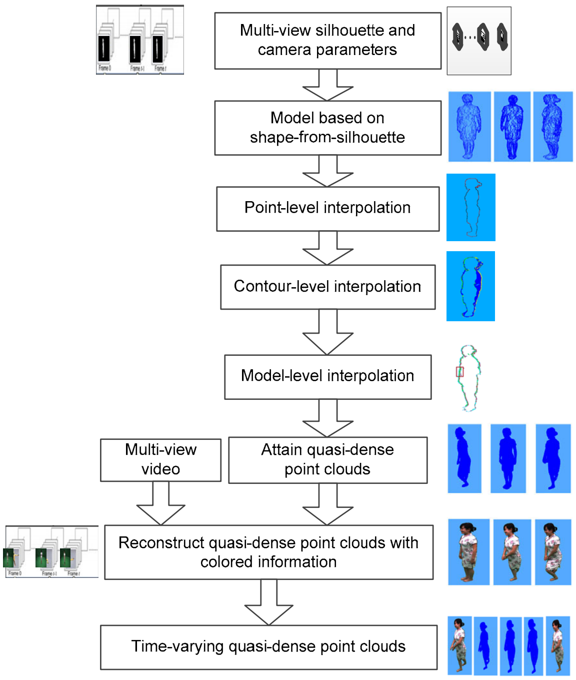
Multi-view silhouette and camera parameters
Model based on shape-from-silhouette
Point-level interpolation
Contour-level interpolation
Model-level interpolation
Multi-view video
Attain quasi-dense point clouds
Reconstruct quasi-dense point clouds with colored information
Time-varying quasi-dense point clouds

**Fig. 1 Block diagram of the multiscale-contour-based reconstruction framework**

$K_i$ ($i$=0, 1, …, $M$−1) and a set of silhouette images $I_i$ ($i$=0, 1, …, $M$−1), where $M$ is the number of cameras. $S_i^j$ stands for the silhouette contour ($j$=0, 1, …, $N$−1), where $N$ is the number of silhouette contours in the $i$th silhouette image. $p_{i,j}^k$ ($k$=0, 1, …, $m$−1) stands for the set of pixels in the $j$th silhouette contour, where $m$ is the number of pixels in the contour. $C_i^j$ stands for the spatial contour corresponding to $S_i^j$. $P_{i,j}^k$ ($k$=0, 1, …, $n$−1) stands for the set of high confidence points, where $n$ is the number of high confidence points.

1. Search for the closest point: Given a set of contour points $C_{\text{Source}} \subseteq C_i^j$ and a set of contour points $C_{\text{Target}} \subseteq C_{\text{Source}}$, we search for the closest point for each point $P_{i,j}^k \in C_i^j$. The rule is the closest Euclidean distance in the spatial domain. Let $d$ stand for the Euclidean distance between $P_{i,j}^{k_1}$ and $P_{i,j}^{k_2}$, $\gamma_{\max}$ the maximum Euclidean distance between neighboring points, and $\lambda_{\max}$ the maximum number of interpolations. The algorithm is shown in Algorithm 1. The algorithm starts with a starting point and continues to its neighboring point recursively, until the algorithm reaches the end point. Finally, a set of point-pairs is obtained.

**Algorithm 1** Searching for the closest point

```
Input: $P_{i,j}^{k_1} \in C_{\text{Source}}$, $P_{i,j}^{k_2} \in C_{\text{Target}}$, $d_{\min}$=10 000, $h_{\text{temp}}$=0
Output: A set of corresponding point-pairs
  for $k_1$=0 to $m$−1
      for $k_2$=0 to $m$−1
          $d = D(P_{i,j}^{k_1}, P_{i,j}^{k_2})$;
          if $d \le d_{\min}$ and $k_2 \ge h_{\text{temp}}$ then
              $h_{\text{temp}} = k_2$;
          end if
      end for
      point-pair $(P_{i,j}^{k_1}, P_{i,j}^{k_{\text{temp}}})$;
  end for
```

2. Expand between point-pairs: To express the idea mathematically, assume $P_{\text{Source}}^i \in C_{\text{Source}}$ ($i$=0, 1, …, $n_1$−1), $P_{\text{Target}}^j \in C_{\text{Target}}$ ($j$=0, 1, …, $n_2$−1), where $n_1$ and $n_2$ are the numbers of points in $C_{\text{Source}}$ and $C_{\text{Target}}$, respectively, $P_{\text{Source}}^i$ and $P_{\text{Target}}^j$ stand for the high confidence points. Let $P_\eta$ stand for the interpolation point, $S$ a set of expansion points, and $S_{\text{sub}}$ a subset of expansion points. We obtain $S$ by the following steps:

First, assuming $P_{\text{Source}}$ is the source point and $P_{\text{Target}}$ the target point, we interpolate the two high confidence points using

$$C_{\text{Middle}} = f(P_{\text{Source}}, P_{\text{Target}}, \eta), \tag{1}$$

where $\eta = l/(\lambda_{\max}-1)$ ($l$=0, 1, …, $\lambda_{\max}$−1) and $\eta \in [0, 1]$. The parameter $\eta$ indicates the progress of transport and the transition rate. Our approach obtains the interpolation point linearly between the two points using

$$f = (1-\eta)P_{\text{Source}} + \eta P_{\text{Target}}. \tag{2}$$

We choose the straight motion path. Therefore, the wrap function $f(\cdot)$ will be the simple linear interpolation. Note that the interpolation is quite simple and straightforward, so our framework can speed up the reconstruction. The method will be used in the following subsections.

Second, we obtain the interpolation point $P_\eta$ and then insert $P_\eta$ into $S_{\text{sub}}$.

Third, we let $S$=$S \cup S_{\text{sub}}$.

The process of interpolation between two points will stop when the termination criterion is satisfied. The criterion is: $l$ exceeds the maximum interpolation number $\lambda_{\max}$ or $d$ is less than the threshold $\gamma_{\max}$.

### 3.3 Contour-level interpolation

There is spatio-contour consistency between two neighboring camera views at the same time $t$ (Fig. 2). Fig. 2a is the source silhouette image, Fig. 2b is the interpolation silhouette image between Figs. 2a and 2c, and Fig. 2c is the target silhouette image. In the area framed by the rectangle, it can be seen that the shape of hand is similar. The consistency between neighboring views will rise as the number of views increases.

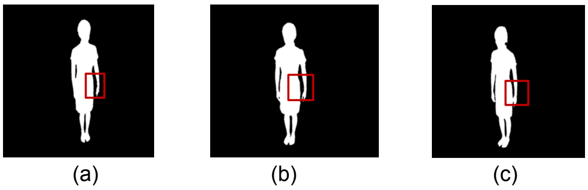

**Fig. 2 The silhouette image for neighboring views**
(a) Source silhouette image; (b) Interpolation silhouette image; (c) Target silhouette image

Franco and Boyer (2009) meshed the point cloud using this consistency to improve the reconstruction quality. We use the consistency to expand the sparse point cloud.

Following the spatio-contour consistency theory, we expand the sparse point cloud at the contour level. The easiest way to produce blends of two contour curves is to interpolate the coordinates of vertices. So, we obtain linearly the interpolation curves between two neighboring curves.

To explain the motivation here, consider a sparse point cloud $\Omega_s$ which consists of $M$ contour curves corresponding to $M$ camera views (for details about $\Omega_s$ refer to Huang *et al.* (2013)). The expansion procedure is implemented in sequential order: search for the corresponding point-pairs, interpolate point- pairs between two contour curves, obtain a quasi- dense point cloud with color information, and remove outliers.

1. Search for the corresponding point-pairs: To acquire a fine correspondence for each point in the contour $C^i_{\text{Source}}$, we search for the closest point in the contour $C^j_{\text{Target}}$. Note that we consider only the points which belong to $C^i_{\text{Source}}$ or $C^j_{\text{Target}}$ instead of all the high confidence points. For simplicity, let us consider the interpolation of two neighboring contours $C^i_{\text{Source}} \subseteq \Omega_s$ and $C^j_{\text{Target}} \subseteq \Omega_s$. The source contour $C^i_{\text{Source}}$ is generated by the $i$th desired image, and the target contour $C^j_{\text{Target}}$ is generated by the $j$th desired image. The desired source image and the target image are generated by the corresponding view (for details refer to Section 3.2).

2. Interpolate the point-pairs between two contour curves: With regard to the source contour curve, we start with the starting point and continue to its neighbors recursively, until the algorithm reaches the end point. We execute the process simultaneously on the target contour curve. To interpolate point-pairs between two contour curves, we adopt the method mentioned in Section 3.2. Because $l$ is adaptive according to $\gamma_{\max}$ during the process of the interpolation, the method can speed up the reconstruction. After interpolating all the contour curves of the sparse point cloud, the sparse point cloud turns into a quasi-dense point cloud (Fig. 3).

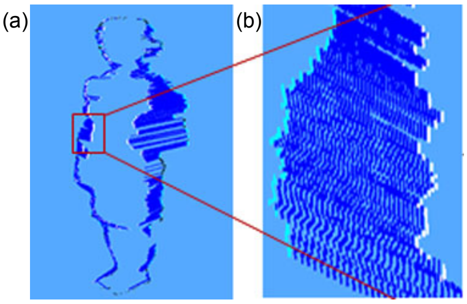

**Fig. 3 The result of the interpolation between spatio contours which belong to the same point cloud (a) and a zoom of the rectangular area (b)**
The contour of white points is the source contour curve; the contour of green points is the target contour curve; the blue points are the expansion points (References to color refer to the online version of this figure)

3. Obtain quasi-dense point cloud with color information: The quasi-dense point cloud is projected onto the corresponding multi-view image to obtain a point cloud with color information (Huang *et al.*, 2013). We briefly describe the method as follows:

First, we project the high confidence point onto the corresponding desired image to obtain the color information of the high confidence point.

Second, we project the expansion point, which is generated by interpolating the corresponding high confidence point, onto the same desired image and then obtain the expansion point with color information. The reason is that the expansion point can be visible in the same view, which generates the corresponding desired image.

4. Remove the outliers: To remove the outliers, the approach in Huang *et al.* (2014a) is adopted. In addition, it is important to distinguish the high confidence point or expansion point from the quasi-dense point cloud during the process of removing the outliers. The reason is that if the 3D point is the high confidence point, then we will not need to determine the location relationship between the 3D point and the point cloud.

Finally, we obtain the color and quasi-dense point cloud sequence.

### 3.4 Model-level interpolation

In this subsection, we interpolate the quasi-dense point cloud sequence at the model level. We express the interpolation of two shapes as a process where one shape deforms to maximize its similarity to another shape. We enhance the inter-frame consistency of the silhouette image sequence by interpolating the distance field image, and then reconstruct the point cloud according to the interpolation silhouette image. Therefore, we can obtain a more accurate interpolation model between two quasi-dense point clouds.

In a multi-video sequence, the interval between successive frames is very short. In other words, there is a temporal-contour consistency between neighboring frames similar to the inter-frame consistency which is used in many fields (for example, motion estimation technology uses time redundancy to improve the coding efficiency in the video compression field). An example of temporal-contour consistency between frames $t$ and $t$+1, which belong to the same viewpoint, is shown in Fig. 4. Within the area framed by the rectangle, the shape of the arm is similar. This phenomenon makes it possible to interpolate the neighboring point cloud.

We adopt the shape tracking approach that is based on interpolating neighboring quasi-dense point clouds. Following this idea, we complete the interpolation between the source point cloud and the target point cloud. The process consists of two steps: searching for a corresponding point-pair, and interpolating the point-pair between two quasi-dense point clouds.

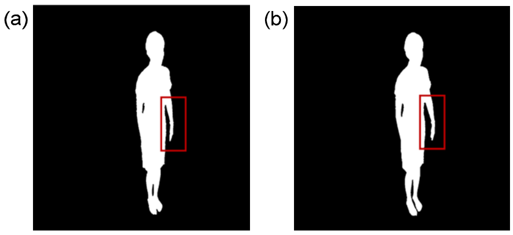

**Fig. 4 The silhouette at frames *t* (a) and *t*+1 (b)**

Consider two quasi-dense point clouds $\Omega_{\text{Source}}$ and $\Omega_{\text{Target}}$ ($\Omega_{\text{Source}}$ stands for source point cloud, and $\Omega_{\text{Target}}$ stands for target point cloud), each consisting of $M$ contour curves corresponding to $M$ camera views.

1. Search for a corresponding point-pair: Assume $C^i_{\text{Source}} \subseteq \Omega_{\text{Source}}$ ($i$=0, 1, …, $n_3$−1) and $C^j_{\text{Target}} \subseteq \Omega_{\text{Target}}$ ($j$=0, 1, …, $n_4$−1), where $n_3$ and $n_4$ are the numbers of points in $C^i_{\text{Source}}$ and $C^j_{\text{Target}}$, respectively.

To obtain a point-pair that belongs to the neighboring point cloud, we adopt the approach mentioned in Section 3.2. Note that it considers only the same camera view and the neighboring point cloud instead of all the high confidence points.

2. Interpolate point-pairs between two neighboring models: The transport can be performed by computing the location of the interpolation contour curve $C_{\text{Middle}}$. To express the idea mathematically, we interpolate the two contour curves through

$$C_{\text{Middle}} = g(C^i_{\text{Source}}, C^j_{\text{Target}}, \tau), \tag{3}$$

where $\tau$ is the progress of transport. We choose the straight motion path. The wrap function $g(\cdot)$ is a simple linear interpolation.

We traverse the two contour curves. With regard to the source contour curve, we start with the starting point and continue to the closest point recursively, until the algorithm reaches the end point. We execute the process simultaneously on the target contour curve, as shown in Fig. 5 (for details refer to Section 3.2).

After interpolating all the point-pairs, the

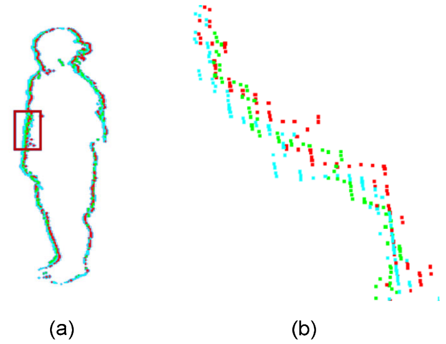

**Fig. 5 The interpolation contour between spatio contours which belong to the neighboring point cloud**
(a) The interpolation contour curves between two contour curves; (b) The zoom on the rectangular area in (a) (the source contour consists of the red points, the target contour consists of the blue points, and the interpolation contour consists of the green points); (c) The interpolation model (References to color refer to the online version of this figure)

interpolation point cloud is obtained. From a transport point of view, an interpolation point cloud is expressed as a set of multiple weighted data points. Because there are multiple point clouds that come from both the source and target models, the resulting point cloud has some outliers because of numerical instabilities. To remove the outliers, the resulting point cloud sequence is clipped.

Finally, we obtain the time-varying and spatio-temporal-coherence point cloud sequence.

## 4 Experimental results and analysis

We conducted experiments to demonstrate the performance of our multiscale-contour-based interpolation framework on the publicly available datasets 'Cheongsam' (Liu *et al.*, 2010), 'Redskirt' (Liu *et al.*, 2010), 'Redshirt' (Liu *et al.*, 2010), and 'Lady Dance' (http://4drepository.inrialpes.fr/public/datasets).

'Cheongsam' is a video of real-life performances and has 20 views. The image (video) sequence was collected at a frame rate of 25 frames/s. The resolution of images used for reconstruction was 1024×768. It is a short sequence (20 frames) with various types of actions such as standing, turning, and squatting. 'Redskirt' and 'Redshirt' are similar to 'Cheongsam'. Frames 111, 118, and 128 of 'Cheongsam' show standing, turning, and squatting, respectively (Fig. 6).

'Lady Dance' is a dataset of a 3D photography collection and has 8 images from 8 viewpoints. We selected 6 viewpoints to illustrate our method.

All the experiments were run on a desktop computer. The hardware consisted of a 3.16 GHz Intel® Core Duo E8500 CPU, an ATI Radeon HD 3450 graphics card, and a 12 GB memory. The exact camera parameters were known a priori. Thus, we could assess the performance of our framework in approximately ideal conditions.

### 4.1 Point-level interpolation

To assess the quality of point-level interpolation, we interpolated the contour curve of frame 112 on the first view. The results for different values of parameters $\lambda_{\max}$ and $\gamma_{\max}$ are shown in Fig. 7.

The numbers of 3D points for different values of parameters $\lambda_{\max}$ and $\gamma_{\max}$ are listed in Table 1. The results illustrate the performance of point-level interpolation. As $\lambda_{\max}$ increases, the denseness of the contour curve increases (Fig. 7 and Table 1). Therefore, the coherence of the contour is improved, which lays the foundation for the exactness of the point-pair. With a decrease in $\gamma_{\max}$, we can obtain a similar result.

### 4.2 Contour-level interpolation

To assess the efficiency of contour-level interpolation, we interpolated the sparse point cloud. The results are shown in Fig. 8 and Table 2.

The quality of the model on the dataset 'Lady Dance' was poor (Fig. 8). The reason is that the consistency between neighboring views was poor because we selected only 6 views from the dataset. The consistency will rise if the number of views was increased. The quality of the model on the datasets 'Cheongsam' and 'Redshirt' confirms this conclusion. Each of the two datasets has 20 images from 20 viewpoints.

The reconstruction times of frame 0 of 'Lady Dance', frame 0 of 'Redshirt', and frame 118 of 'Cheongsam' for different values of parameters $\lambda_{\max}$ and $\gamma_{\max}$ are listed in Table 2. This illustrates the result of contour-level interpolation. The reconstruction time will decrease as the view number decreases.

Parameters $\lambda_{\max}$ and $\gamma_{\max}$ together determine the denseness of the point cloud (Fig. 8 and Table 2). When $\lambda_{\max}$ increases or $\gamma_{\max}$ decreases, the higher denseness of a quasi-dense point cloud is obtained along with an increase in reconstruction time. The

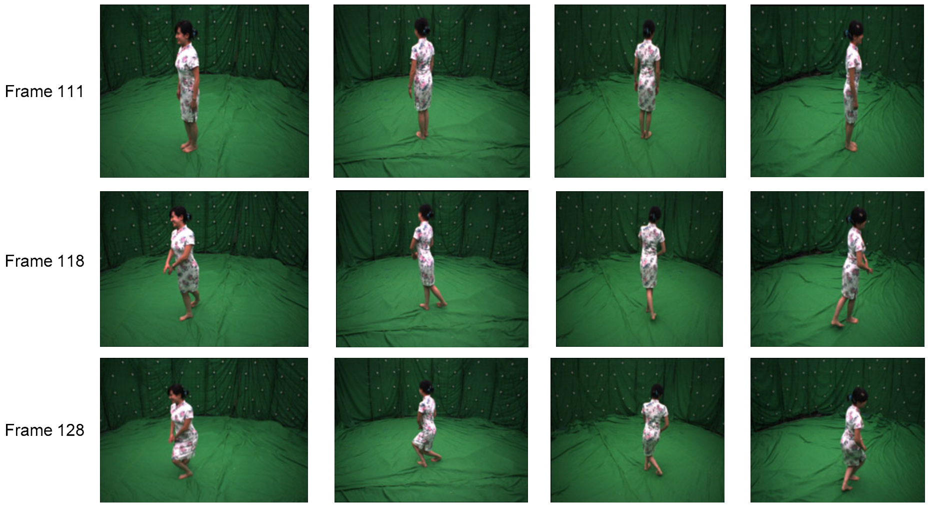

**Fig. 6 Samples of ‘Cheongsam’: frame 111 (top row), frame 118 (middle row), and frame 128 (bottom row)**

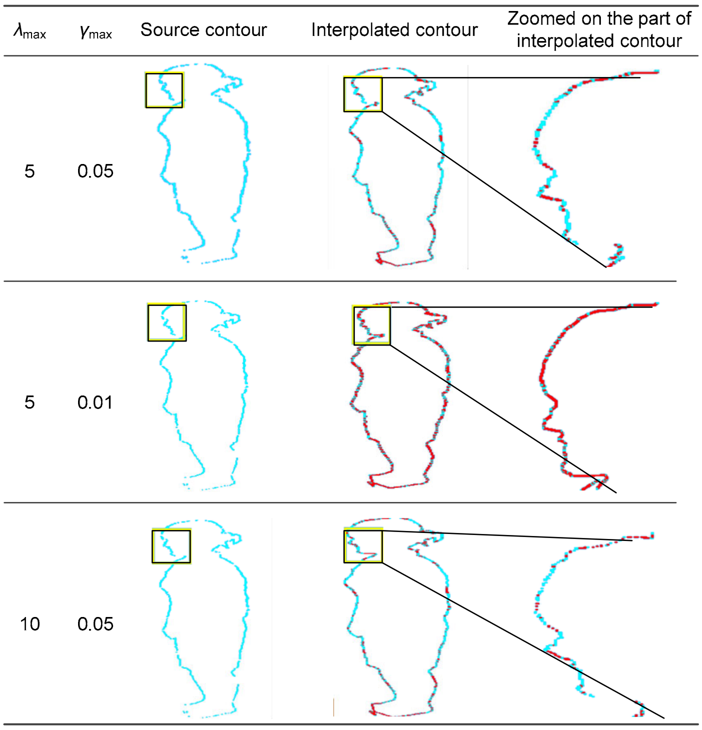

**Fig. 7 Point-level interpolation for different values of $\lambda_{\max}$ and $\gamma_{\max}$**
Third column: source contour; fourth column: overlap between the interpolation contour (red) and source contour (green); fifth column: zoom on the yellow-framed rectangular area (References to color refer to the online version of this figure)

**Table 1 The numbers of 3D points for different values of $\lambda_{\max}$ and $\gamma_{\max}$**

| $\lambda_{\max}$ | $\gamma_{\max}$ | Number of high confidence points | Number of expansion points |
|---|---|---|---|
| 5 | 0.05 | 1005 | 865 |
| 5 | 0.01 | 1005 | 6732 |
| 10 | 0.05 | 1005 | 1329 |

number of interpolations differs between point-pairs $(P^i_{\text{Source}}, P^j_{\text{Target}})$ (Section 3.3). In other words, it is adaptive.

### 4.3 Model-level interpolation

To assess the quality of model-level interpolation, we show the samples from the point cloud

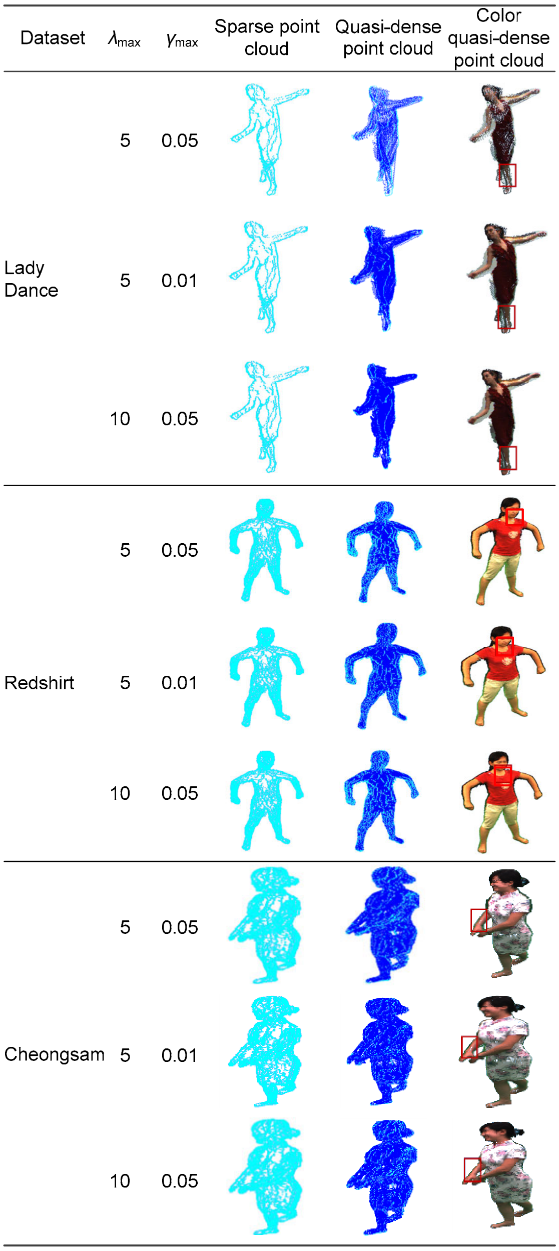

**Fig. 8 Interpolation of the sparse point cloud**

Fourth column: sparse point cloud which consists of the high confidence points (green); fifth column: quasi-dense point cloud which consists of the high confidence points (green) and the expansion points (blue); sixth column: quasi-dense point cloud with color information. The denseness is different in the red-framed rectangular area (References to color refer the online version of this figure)

sequence in Fig. 9. The point cloud transports from frames 111 to 112, 118 to 119, and 128 to 129. Note that we set $\lambda_{\max}$=5, $\gamma_{\max}$=0.01, and $\tau$=0.5. Generally speaking, as $\lambda_{\max}$ increases, the temporal resolution and temporal coherence of the model sequence will increase.

To test the performance of our framework on different video sequences, we conducted experiments on the dataset ‘Redskirt’. The dataset is a short sequence showing medium speed dance moves, and thus offers a good opportunity to verify the tracking stability. The transportation process between neighboring models is illustrated in Fig. 10.

The source shape transports gradually to maximize its similarity to the target shape (Figs. 9 and 10). Note that the result from frames 45 to 46 appears similar to that from frames 47 to 48. The reason is that these four frames are neighboring. However, the rectangular area shows that the result is slightly different. This reflects the performance of our framework.

Note that we show mainly the experimental results of dataset ‘Cheongsam’ to explain our ideas. We show only part of the experimental results from ‘Redskirt’, ‘Redshirt’, and ‘Lady Dance’.

We then analyze the average reconstruction time to assess the efficiency of our proposed framework.

First, we provide the average reconstruction time between neighboring frames of the dataset ‘Cheongsam’. A comparison of the result with those of three state-of-the-art methods (Liu *et al.*, 2010; Bilir and Yemez, 2012; Allain *et al.*, 2015) is listed in Table 3. The results support the intuitiveness of our method efficiency between neighboring frames.

Second, we analyze the average reconstruction time using the sequence ‘Cheongsam’ (Fig. 11). The result is 54.59 s per frame, while the reconstruction of a single frame would take about 110 s per frame using the SFS approach. Compared with reconstructing the model for each time instance, we have reduced the average reconstruction time. Note that the frame index is regenerated after interpolating the quasi-dense point cloud sequence.

A comparison with state-of-the-art methods (Liu *et al.*, 2010; Bilir and Yemez, 2012; Allain *et al.*, 2015) is shown in Table 4, which shows quantitative evaluation of the average reconstruction time per frame. Although the method of Bilir and Yemez (2012) showed a good result, our method still sped up the reconstruction. In summary, the proposed framework shows competitive performance on neighboring frames and through the sequence.

**Table 2 Reconstruction time for a single frame**

| Dataset | Number of views | $\lambda_{\max}$ | $\gamma_{\max}$ | Number of high confidence points | Number of expansion points | Time (s) |
|---|---|---|---|---|---|---|
| Lady Dance | 6 | 5 | 0.05 | 3064 | 50 517 | 11.75 |
| | 6 | 5 | 0.01 | 3064 | 58 818 | 12.35 |
| | 6 | 10 | 0.05 | 3064 | 82 975 | 12.78 |
| Redshirt | 20 | 5 | 0.05 | 33 291 | 270 098 | 84.68 |
| | 20 | 5 | 0.01 | 33 291 | 380 192 | 89.48 |
| | 20 | 10 | 0.05 | 33 291 | 400 265 | 92.36 |
| Cheongsam | 20 | 5 | 0.05 | 21 265 | 238 537 | 77.54 |
| | 20 | 5 | 0.01 | 21 265 | 271 576 | 85.24 |
| | 20 | 10 | 0.05 | 21 265 | 301 190 | 90.51 |

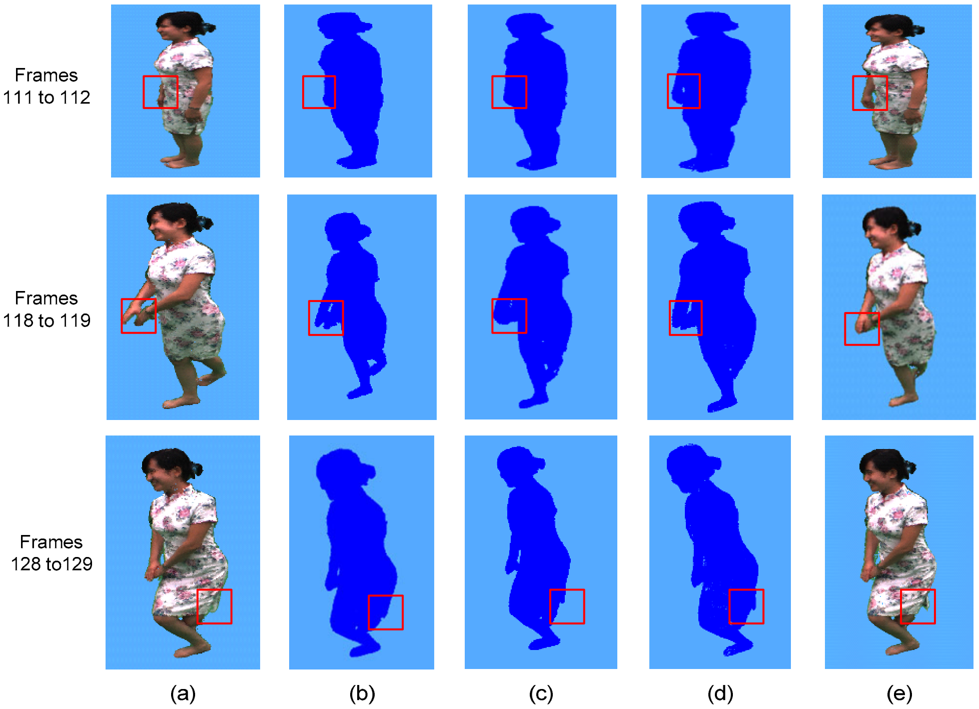

**Fig. 9 Point cloud transporting from frames 111 to 112, 118 to 119, and 128 to 129**
(a) Source quasi-dense point cloud; (b) Interpolation point cloud between (a) and (c); (c) Point cloud based on the interpolation silhouette image; (d) Interpolation point cloud between (c) and (e); (e) Target quasi-dense point cloud

To obtain a quantitative analysis of the quality of reconstruction, an experiment on datasets 'Cheongsam', 'Redskirt', and 'Lady Dance' was performed. Table 5 lists the accuracy and completeness of the final results with respect to the ground truth model. Note that we set $\lambda_{\max}$=5, $\gamma_{\max}$=0.01, and $\tau$=0.5. The index of the model is shown in parentheses in the table.

In our experiment, we interpolated the high confidence point, and then expanded the resulting model. An alternative approach would be interpolating the model consisting of the expansion point and high confidence point to speed up the reconstruction. However, this alternative approach would lead to more noise, e.g., more outliers.

The following are some tips in relation to the parameters:

1. To speed up reconstruction, the search range should be limited using the $x$ coordinate of the contour pixel $p(x, y)$. In our experiment, the search range was set to $[x-50, x+50]$.

2. The final interpolation number, which is determined by the maximum resolution $\gamma_{\max}$, is adaptive. To speed up the reconstruction and improve the accuracy, the Euclidean distance $\gamma_{\max}$ between two points was set to [0.05, 0.50] in our experiment.

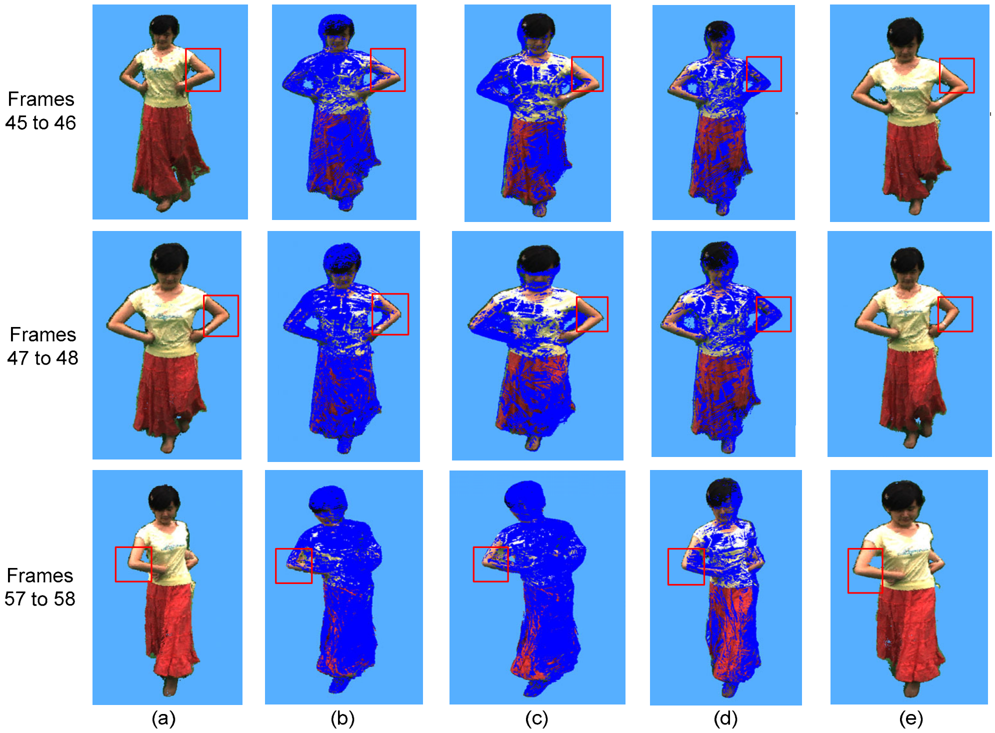

**Fig. 10 Point cloud transporting from frames 45 to 46, 47 to 48, and 57 to 58**
(a) Source quasi-dense model; (b) Overlap between the interpolation quasi-dense model (blue) and the source model; (c) Overlap between the quasi-dense model based on interpolation silhouette (blue) and the source model; (d) Overlap between the interpolation quasi-dense model (blue) and the target model; (e) Target quasi-dense model (References to color refer to the online version of this figure)

**Table 3 Average reconstruction time per frame between neighboring frames using different methods**

| Method | Reconstruction time (s) | | |
|---|---|---|---|
| | Frames 111 to 112 | Frames 118 to 119 | Frames 128 to 129 |
| Liu *et al.* (2010) | 140.32 | 138.61 | 141.94 |
| Bilir and Yemez (2012) | 67.45 | 66.43 | 71.45 |
| Allain *et al.* (2015) | 80.73 | 78.72 | 82.75 |
| Our method | 59.26 | 47.69 | 49.19 |

**Table 4 Average reconstruction time per frame through the sequence**

| Dataset | Method | Time (s) |
|---|---|---|
| Cheongsam | Liu *et al.* (2010) | 142.68 |
| | Bilir and Yemez (2012) | 67.45 |
| | Allain *et al.* (2015) | 80.85 |
| | Our method | 52.59 |
| Redskirt | Liu *et al.* (2010) | 145.45 |
| | Bilir and Yemez (2012) | 68.45 |
| | Allain *et al.* (2015) | 80.78 |
| | Our method | 56.54 |

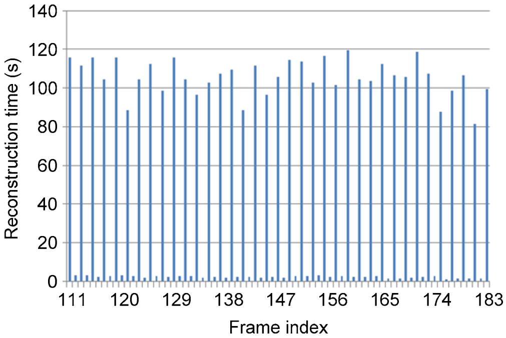

**Fig. 11 Reconstruction time of individual frames for dataset 'Cheongsam'**

3. A higher frame rate point cloud sequence can be obtained by setting the parameter $\tau$.

Because $\lambda_{\max}$ and $\gamma_{\max}$ have a strong impact on quality as well as speed, and determine the denseness of the point cloud, to obtain the balance between speed and denseness, an appropriate value should be selected according to the different scenes and requirements.

**Table 5 Accuracy and completeness results for different datasets and methods**

| Method | Accuracy | | | Completeness | | |
|---|---|---|---|---|---|---|
| | Cheongsam (111) | Redskirt (45) | Lady Dance (00) | Cheongsam (111) | Redskirt (45) | Lady Dance (00) |
| Liu *et al.* (2010) | 0.65 | 0.70 | 0.79 | 98.1% | 97.9% | 86.2% |
| Bilir and Yemez (2012) | 0.72 | 0.79 | 0.85 | 97.8% | 98.3% | 86.8% |
| Allain *et al.* (2015) | 0.59 | 0.64 | 0.77 | 99.0% | 99.2% | 87.1% |
| Our method | 0.95 | 1.03 | 1.18 | 94.5% | 93.8% | 83.0% |

The index of the model is shown in parentheses in the table

## 5 Conclusions

We have proposed an interpolation framework to construct the shape of an object. The interpolation is performed within a frame, between neighboring frames temporally, and between neighboring frames captured by different cameras.

Our approach focused on improving the reconstruction speed. Numerical experiments demonstrated the effectiveness of our approach. We have obtained a higher frame rate, spatio-temporal-coherence, and quasi-dense point cloud sequence with color information. The time-varying quasi-dense point cloud representations of the shape of dynamic 3D objects can be tracked quickly thanks to the multiscale-contour-based interpolation and spatio-temporal-contour consistency. The major contributions of this paper are twofold:

1. Speeding up reconstruction by use of the multiscale-contour-based framework. According to the different levels of the contour consistency, we integrated the multiscale spatio-temporal-contour consistency into a framework and made good use of these consistencies to reduce the reconstruction time.

2. Improving the exactness of point-pairs. We searched for the corresponding point for each 3D point in the spatial domain. By improving the exactness of point-pairs, a point cloud with higher accuracy can be obtained.

Our ultimate goal is to apply the framework to generate a free-viewpoint video. Therefore, in future work we will interpolate the pixel pairs between the color images for more realistic 3D modeling in real scenes.

## References

Ahmed, N., Junejo, I.N., 2013. A system for 3D video acquisition and spatio-temporally coherent 3D animation reconstruction using multiple RGB-D cameras. *Int. J. Signal Process. Image Process. Patt. Recogn.*, **6**(2):113-128.

Allain, B., Franco, J.S., Boye, R.E., 2015. An efficient volumetric framework for shape tracking. IEEE Conf. on Computer Vision and Pattern Recognition, p.268-276. http://dx.doi.org/10.1109/CVPR.2015.7298623

Arita, D., Taniguchi, R., 2001. RPV-II: a stream-based real-time parallel vision system and its application to real-time volume reconstruction. 2nd Int. Workshop on Computer Vision Systems, p.174-189. http://dx.doi.org/10.1007/3-540-48222-9_12

Baumgart, B.G., 1974. Geometric Modeling for Computer Vision. PhD Thesis, Stanford University, Stanford, USA.

Bilir, S.C., Yemez, Y., 2012. Non-rigid 3D shape tracking from multiview video. *Comput. Vis. Image Understand.*, **116**(11): 1121-1134. http://dx.doi.org/10.1016/j.cviu.2012.07.001

Borovikov, E., Sussman, A., Davis, L., 2003. A high performance multi-perspective vision studio. 17th Annual Int. Conf. on Supercomputing, p.348-357. http://dx.doi.org/10.1145/782814.782862

Cheung, G.K.M., Kanade, T., Bouguet, J.Y., 2000. A real time system for robust 3D voxel reconstruction of human motions. IEEE Conf. on Computer Vision and Pattern Recognition, p.714-720. http://dx.doi.org/10.1109/CVPR.2000.854944

Cheung, G.K.M., Baker, S., Kanade, T., 2003. Visual hull alignment and refinement across time: a 3D reconstruction algorithm combining shape-from-silhouette with stereo. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, p.375-382. http://dx.doi.org/10.1109/CVPR.2003.1211493

Díaz-Más, L., Muñoz-Salinas, R., Madrid-Cuevas, F.J., *et al.*, 2010. Shape from silhouette using Dempster-Shafer theory. *Patt. Recog.*, **43**(6):2119-2131. http://dx.doi.org/10.1016/j.patcog.2010.01.001

Duckworth, T., Roberts, D.J., 2011. Accelerated polyhedral visual hulls using OpenCL. IEEE Virtual Reality Conf., p.203-204. http://dx.doi.org/10.1109/VR.2011.5759469

Franco, J.S., Boyer, E., 2009. Efficient polyhedral modeling from silhouettes. *IEEE Trans. Patt. Anal. Mach. Intell.*, **31**(3):414-427. http://dx.doi.org/10.1109/TPAMI.2008.104

Furukawa, Y., Ponce, J., 2009. Carved visual hulls for image-based modeling. *Int. J. Comput. Vis.*, **81**(1):53-67.

http://dx.doi.org/10.1007/s11263-008-0134-8

Furukawa, Y., Ponce, J., 2010. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Patt. Anal. Mach. Intell.*, **32**(8):1362-1376. http://dx.doi.org/10.1109/TPAMI.2009.161

Haro, G., 2012. Shape from silhouette consensus. *Patt. Recogn.*, **45**(9):3231-3244. http://dx.doi.org/10.1016/j.patcog.2012.02.029

Hasler, N., Rosenhahn, B., Thormahlen, T., *et al.*, 2009. Markerless motion capture with unsynchronized moving cameras. IEEE Conf. on Computer Vision and Pattern Recognition, p.224-231. http://dx.doi.org/10.1109/CVPR.2009.5206859

Hauswiesner, S., Khlebnikov, R., Steinberger, M., *et al.*, 2012. Multi-GPU image-based visual hull rendering. 12th Eurographics Symp. on Parallel Graphics and Visualization, p.119-128.

Hofmann, M.H., Davrila, M.G., 2009. Multi-view 3D human pose estimation combining single-frame recovery, temporal integration and model adaptation. IEEE Conf. on Computer Vision and Pattern Recognition, p.2214-2221. http://dx.doi.org/10.1109/CVPR.2009.5206508

Huang, C.H., Lu, D.M., Diao, C.Y., 2013. Accelerated visual hulls of complex objects using contribution weights. Proc. 7th Int. Conf. on Image and Graphics, p.685-689. http://dx.doi.org/10.1109/ICIG.2013.139

Huang, C.H., Lu, D.M., Diao, C.Y., 2014a. A point cloud representation using plane-space-local-area-color-consistency. *J. Comput.-Aided Des. Comput. Graph.*, **26**(8):1297-1303 (in Chinese).

Huang, C.H., Boyer, E., Navab, N., *et al.*, 2014b. Human shape and pose tracking using keyframes. IEEE Conf. on Computer Vision and Pattern Recognition, p.3446-3453. http://dx.doi.org/10.1109/CVPR.2014.440

Kanaujia, A., Haering, N., Taylor, G., *et al.*, 2011. 3D human pose and shape estimation from multi-view imagery. IEEE Computer Vision and Pattern Recognition Workshops, p.49-56. http://dx.doi.org/10.1109/CVPRW.2011.5981821

Kim, D., Dahyot, R., 2012. Bayesian shape from silhouettes. Int. Workshop on Multimedia Understanding Through Semantics, Computation, and Learning, p.78-89. http://dx.doi.org/10.1007/978-3-642-32436-9_7

Laurentini, A., 1994. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Patt. Anal. Mach. Intell.*, **16**(2):150-162. http://dx.doi.org/10.1109/34.273735

Li, K., Dai, Q.H., Xu, W.L., 2011. Markerless shape and motion capture from multiview video sequences. *IEEE Trans. Circ. Syst. Video Technol.*, **21**(3):320-334. http://dx.doi.org/10.1109/TCSVT.2011.2106251

Liu, Y.B., Dai, Q.H., Xu, W.L., 2010. A point-cloud-based multiview stereo algorithm for free-view-point video. *IEEE Trans. Vis. Comput. Graph.*, **16**(3):407-418. http://dx.doi.org/10.1109/TVCG.2009.88

Matsuyama, T., Wu, X.J., Takai, T., *et al.*, 2004. Real-time dynamic 3-D object shape reconstruction and high-fidelity texture mapping for 3-D video. *IEEE Trans. Circ. Syst. Video Technol.*, **14**(3):357-369. http://dx.doi.org/10.1109/TCSVT.2004.823396

Matusik, W., Buehler, C., Raskar, R., *et al.*, 2000. Image-based visual hulls. ACM Special Interest Group on Computer Graphics, p.369-374. http://dx.doi.org/10.1145/344779.344951

Nakajima, H., Makihara, Y., Hsu, H., *et al.*, 2012. Point cloud transport. 21st Int. Conf. on Pattern Recognition, p.3803-3806.

Nakazawa, M., Mitsugami, I., Makihara, Y., *et al.*, 2012. Dynamic scene reconstruction using asynchronous multiple Kinects. 21st Int. Conf. on Pattern Recognition, p.469-472.

Perez, J.M., Aledo, P.G., Sanchez, P.P., 2012. Real-time voxel-based visual hull reconstruction. *Microprocess. Microsyst.*, **36**(5):439-447. http://dx.doi.org/10.1016/j.micpro.2012.05.003

Raeesi N., M.R., Wu, Q.M.J., 2010. A complete visual hull representation using bounding edges. 11th Pacific-Rim Conf. on Multimedia, p.171-182. http://dx.doi.org/10.1007/978-3-642-15702-8_16

Taneja, A., Ballan, L., Pollefeys, M., 2011. Modeling dynamic scenes recorded with freely moving cameras. 10th Asian Conf. on Computer Vision, p.613-626.

Vlasic, D., Peers, P., Baran, I., *et al.*, 2009. Dynamic shape capture using multi-view photometric stereo. *ACM Trans. Graph.*, **28**(5):174. http://dx.doi.org/10.1145/1618452.1618520

Wang, S.Y., Yu, H.M., 2012. Convex relaxation for a 3D spatiotemporal segmentation model using the primal-dual method. *J. Zhejiang Univ.-Sci. C (Comput. & Electron.)*, **13**(6):428-439. http://dx.doi.org/10.1631/jzus.C1100331

Wu, X.J., Takizawa, O., Matsuyama, T., 2006. Parallel pipeline volume intersection for real-time 3D shape reconstruction on a PC cluster. IEEE Int. Conf. on Computer Vision Systems, p.1-4. http://dx.doi.org/10.1109/ICVS.2006.49

Xia, D., Li, D.H., Li, Q.G., 2011. A novel approach for computing exact visual hull from silhouettes. *Optik*, **122**(24):2220-2226. http://dx.doi.org/10.1016/j.ijleo.2011.02.013

Zhang, Z., Seah, H.S., Quah, C.K., *et al.*, 2011. A multiple camera system with real-time volume reconstruction for articulated skeleton pose tracking. 17th Int. Multimedia Modeling Conf., p.182-192. http://dx.doi.org/10.1007/978-3-642-17832-0_18