Noname manuscript No.
(will be inserted by the editor)

Volumetric Stereo and Silhouette Fusion for Image-Based Modeling

Peng Song · Xiaojun Wu · Michael Yu Wang

Received: date / Accepted: date

Abstract This paper presents a volumetric stereo and silhouette fusion algorithm for acquiring high quality models from multiple calibrated photographs. Our method is based on computing and merging depth maps. Different to previous methods of this category, the silhouette information is also applied in our algorithm to recover the shape information on the textureless and occluded areas. The proposed algorithm starts by computing visual hull using a volumetric method in which a novel projection test method is proposed for visual hull octree construction. Then, the depth map of each image is estimated by an expansion-based approach that returns a 3D point cloud with outliers and redundant information. After generating an oriented point cloud from stereo by rejecting outlier, reducing scale, and estimating surface normal for the depth maps, another oriented point cloud from silhouette is added by carving the visual hull octree structure using the point cloud from stereo to restore the textureless and occluded surfaces. Finally, Poisson Surface Reconstruction approach is applied to convert the oriented point cloud both from stereo and silhouette into a complete and accurate triangulated mesh model. The proposed approach has been implemented and the performance of the approach is demonstrated on several real datasets, along with qualitative comparisons with the state-of-the-art image-based modeling techniques according to the Middlebury benchmark.

Keywords Multi-view stereo · Depth map · Oriented point cloud · Visual hull

P. Song
Division of Control and Mechatronics Engineering, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen 518055, China
E-mail: songpenghit@163.com

X. Wu
Division of Control and Mechatronics Engineering, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen 518055, China
E-mail: wuxj@hitsz.edu.cn

M. Y. Wang
Department of Mechanical and Automation Engineering, Chinese University of Hong Kong, Shatin, N.T., Hong Kong, China
E-mail: yuwang@mae.cuhk.edu.cn

1 Introduction

At present the art of computing complex and high quality 3D models has large and wide applications in computer graphics, medical imaging, 3D animation, electronic games, etc. In practice, image-based modeling technique is an efficient and convenient method to acquire models of real word object. According to the information it uses, image-based modeling can be categorized into three classes: shape from shading, shape from silhouette, and shape from stereo. Shape from shading methods [1] are based on the diffusing properties of Lambertian surfaces. They require controlled environments where the illumination of the object space and the object reflectance must be known. In the second one [2] [3], 3D object shape is constructed by intersection of the visual cones formed by back-projecting the silhouettes in the corresponding images. The reconstructed 3D object shape is not guaranteed to be the same as the original object since concave surface regions can never be distinguished using silhouette information alone. Methods of shape from stereo seek to reconstruct a depth map for each input view using information contained in the object texture. Evidently, if the object has no texture or if its information is too weak, the method will fail. In this case, another type of information such as silhouette, shading or radiance can be employed to reconstruct accurate and complete models.

This paper proposes an algorithm to reconstruct 3D object surface from multiple calibrated images by using both stereo and silhouette information. Specifically, the proposed reconstruction algorithm can be decomposed into four steps.

In the first step, visual hulls are computed by a volumetric method in which a novel projection test method is proposed for visual hull octree construction. Then, the depth map of each image is estimated from multi-view stereo by an expansion-based approach and an oriented point cloud denoted as point cloud from stereo (PCST) is computed from the depth maps through outlier rejection, point cloud size reduction and surface normal estimation. In the third step, the visual hull octree structure is carved by the PCST to generate another oriented point cloud on the visual hull denoted as point cloud from silhouette (PCSL) to recover the shape information from silhouettes on those parts of the object surface which cannot be captured by texture information. At last, these two point clouds are merged to generate a more complete point cloud on object surface denoted as point cloud from stereo and silhouette (PCSTSL) and Poisson Surface Reconstruction (PSR) [4] approach is applied to reconstruct an accurate and complete surface model from the PCSTSL since this approach can robustly recover the fine details from a set of noisy, non-uniform points.

Compared with traditional image-based modeling approaches, the most obvious unique of our proposed algorithm lies in the visual hull computation, expansion-based depth map estimation, and volumetric stereo and silhouette fusion. In particular, the benefits of our approach are as follows:

- A novel volumetric approach constructs high quality visual hull mesh from silhouettes.
- An expansion-based depth map estimation algorithm outputs dense and accurate depth map quickly.
- A volumetric stereo and silhouette fusion approach applies the silhouette information to amend the missing shape information from multi-view stereo in our reconstruction framework.

The paper is organized as follows. In Section 2, we present a brief review of several related works. In Section 3, a novel volumetric visual hull computation method is addressed in detail. In Section 4, an expansion-based approach is applied for depth map estimation and an oriented PCST is generated by cleaning, downsampling and surface normal estimation for the point cloud merged from the depth maps. In Section 5, the visual hull octree structure is carved by the estimated PCST to generate a PCSL and the PSR approach is applied to reconstruct a complete model from the shape information both from stereo and silhouette. In Section 6 we present experimental results on several datasets, along with qualitative comparisons with several state-of-the-art imaged-based modeling algorithms. Finally, in Section 7, we draw some conclusions and give future directions about this work.

2 Related Work

Although there are many approaches to reconstruct an accurate model of a 3D object from a sequence of calibrated images, these approaches can be mainly categorized into four classes according to the taxonomy of Seitz et al. [5]: 3D volumetric approaches [6] [7] [8], surface evolution approaches [9] [10] [11] , feature extraction and expansion techniques [12] [13] [14] and depth map based methods [15] [16] [17] [18]. In practice, depth map based methods are not only easy to implement but also can reconstruct very accurate surface model. Generally, these methods involve two separate stages. First, a depth map is computed for each viewpoint using binocular stereo. Second, the depth maps are merged to produce a 3D model. In these methods, the estimation of the depth maps is crucial to the quality of the final reconstructed 3D model. Our method falls into the fourth category. However, the proposed approach is different from the previous works of this category in many aspects of which the most important one is that the geometric constraint associated with the silhouettes is incorporated into the reconstruction process. In what follows we discuss five papers that are most closely related to this paper, with an emphasis on the differences compared to our method.

The inspiration for the approach presented in this paper is the work of Hernandez et al. [10]. They recover a complete model by deforming a mesh, initialized as the visual hull, to find a minimum cost surface in a cost volume which is merged from the depth maps for each viewpoint. In the deformation process, they also incorporate an additional silhouette terms to fuse silhouettes with stereo for reconstruction. Since their approach is based on iterative local refinement via snake deformation model, it is susceptible to local minima and unstability due to many parameters need to be tuned. In their algorithm, depth maps are computed by backprojecting the ray for each pixel into the visual hull and then reprojecting the depth interval onto neighboring views where window-based correlation is performed. The work presented here improves the depth map estimation approach by introducing an expansion-based approach. Furthermore, our approach uses an oriented point cloud based approach to combine the shape information both from texture and silhouette into a full 3D model.

Another related work has been reported by Goesele et al. [16]. They use a two-step technique. They first use robust window-based matching to compute reliable depth estimates. Then a volumetric method is applied to merge them into a single surface representation. Although their method is simple to implement, their models suffer from a large number of holes and very long processing time. In contrast, our algorithm is very efficient and can reconstruct complete object surface estimates by using both texture and silhouette information.

Recent work by Furukawa et al. [12] proposes a novel algorithm for calibrated multi-view stereo. The algorithm starts by computing a dense set of small rectangular patches covering the surfaces visible in the images and then converts the resulting patch model into an initial mesh model by PSR approach or iterative snapping. Finally, an optional final refinement algorithm is applied to refine the initial mesh to achieve even higher accuracy. In their work, they computes a dense set of small rectangular patches by a match, expand and filter procedure. In contrast, we compute an oriented point cloud from multi-view stereo by a depth map estimation, point cloud cleaning and downsampling, and surface normal estimation procedure.

Our work is similar to that of Bradley et al. [17] who propose to reconstruct an accurate model from multi-view stereo in two steps: binocular stereo matching on image pairs and surface reconstruction from depth maps. The binocular stereo algorithm creates depth maps from pairs of adjacent viewpoints and makes use of scaled window matching to improve the density and precision of depth estimates. And the surface reconstruction step creates a triangular mesh from the depth maps by a downsampling, cleaning and meshing procedure. Although we process the depth maps by similar procedures, we use different methods to compute depth maps from multi-view stereo. Our algorithm computes depth maps by an expansion-based approach, while Bradley et al. just use the basic binocular stereo matching method to compute depth map for each image.

Space carving [9] is a technique that starts from a volume containing the scene and greedily carves out non-photoconsistent voxels from that volume until all remaining visible voxels are consistent. Since it uses a discrete representation of the surface but does not enforce any smoothness constraint on the surface, the reconstructed results are often quite noisy. Our approach also carves the visual hull octree structure, however, the goal is to generate point cloud from silhouette to amend the missing shape information from stereo which is different from the space carving approach. Furthermore, our algorithm outputs complete and accurate shape estimate represented by triangulated mesh, rather than a voxel based representation.

3 Shape from Silhouette

The visual hull [19] is the maximal shape consistent with an object's silhouettes as seen from any viewpoint in a given region which can be constructed by intersecting the cones generated by back projecting the object silhouettes of a given set of views, shown in Fig. 1. Different approaches for the construction of the visual hull have been developed, such as volumetric method [2], polyhedral method [20] and marching intersection method [21]. The proposed visual hull computation method belongs to volumetric method which is based on the polygonization of octree structure by using marching cubes algorithm [22].

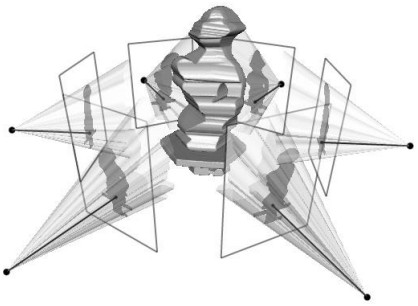


Fig. 1 The intersection of silhouette cones defines an approximate geometric representation of an object called the visual hull [10].

The proposed visual hull computation method can be decomposed into three steps. Firstly, a 3D bounding box of an object is computed by an optimization method. Taking the 3D bounding box as a root node, an octree of the object is reconstructed from the silhouettes through recursive subdivision and projection tests in which a novel projection test method is proposed to determine whether a voxel is locating outside, on or inside the visual hull. Finally, the visual hull mesh is extracted from the octree structure by using marching cubes algorithm. An isosurface function that corresponds to the visual hull surface is also defined in order to extract smooth visual hull mesh.

3.1 3D Bounding Box Estimation

To build visual hull octree structure, an initial bounding box is needed as a root node. In practice, an accurate 3D bounding box can improve the precision of the result mesh. Since we do not dispose of any 3D information, we calculate the 3D bounding box only from a set of silhouettes and the projection matrices. This can be done by considering the 2D bounding boxes of each silhouette and then back projecting these 2D bounding boxes. Then the bounding box of the object can be computed by an optimization method for each of the 6 variables defining the bounding box, which are the maximum and minimum of x, y, z [23].

3.2 Visual Hull Octree Construction

An octree of an object can be reconstructed from the silhouettes through recursive subdivision and projection tests. The process begins with a single voxel, which is the 3D bounding box of the object. Each voxel is projected onto all the images and tested against the silhouettes. The test result classifies the voxel as being inside, outside or on the boundary of the visual hull (see Fig. 2). Among the three types of voxels, only voxels on the boundary of the visual hull contain

the potential visual hull boundary and are subject to further subdivision until the maximum allowed number of subdivision is reached. Therefore, the projection test method is crucial for visual hull octree construction. In this section, a novel projection test method is proposed for visual hull octree construction. Before explaining our method, we present several projection test methods in previous work on visual hull computation.

- R. Szeliski's projection test method [2]: A cube projected into an image plane will in general form a six-sided polygon. Szeliski uses a coarser test based on the hexagon's bounding box, which may sometimes fail to detect a true inclusion or exclusion. As the goal of his algorithm is to obtain a quick and rough model of an object in close to real time, this method is adaptable.
- Y. Yemez et al.'s projection test method [3]: Yemez et al. solve this problem by oversampling the edges of an voxel up to the maximum octree level such that the voxel type decision is based on the state (in or out) of the intermediate sampled points along the edges and the corners. That is: (1) If all these points of the voxel are out of the visual hull, the voxel's type is out; (2) If all these points of the voxel are in the visual hull, the voxel's type is in; (3) Else, the voxel's type is on.
- M. Potmesil's projection test method [24]: This method first computes the exact projection of a voxel and then does intersection test between the projection of the voxel and the silhouettes to determine the type of the voxel.

In order to build visual hull octree structure and extract smooth mesh from the octree, we define an isosurface function that corresponds to the visual hull surface. The visual hull isosurface function for a given 3D point v is

$$f_{iso}(v) = \max_i D_i(P_i \times v), \quad i = 1,2,\cdots,N \tag{1}$$

where D_i is the distance transform [25] to the contour of silhouette i, negative inside and positive outside the silhouette. In fact, for each 3D point, its isosurface function value represents the 3D distance between the 3D point and the visual hull surface, negative inside and positive outside the visual hull.

To evaluate a given voxel, we project it into all the silhouettes to assign it one of three available types. Our projection test method (see Algorithm 1) is similar to Potmesil's one [24] which first computes the exact projection of a voxel and then does intersection test between the projection of the voxel and the silhouettes to determine the type of the voxel. However, our methods are different in two aspects. Firstly, the proposed projection test method takes advantage of the isofunction value of a voxel's 8 corners to decrease computation since when part of the 8 corners are out of visual hull there is no need to project the voxel to all the silhouette images to know its type. Secondly, we use a simpler method to compute the exact projection of a voxel. Potmesil computes the projection of a voxel by determining the relative position between the voxel and the projection center of an image and then getting the projection of the cube based on a lookup table. While our method first computes the projections of 8 corners of a voxel, then calculates the convex hull of them which is the same as the projection of the voxel according to the basic formulation of the convex hull of a 3D polygon projected on a 2D plane.

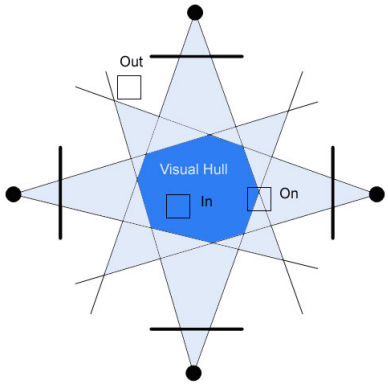


Fig. 2 Three types of a voxel in octree-based visual hull reconstruction: on, in, and out of the visual hull.

Input: A given voxel and all the calibrated silhouette images
Output: The type of the voxel
Calculate the isofunction value of 8 corners of the voxel;
if *the 8 corners of the voxel are out of the visual hull* **then**
  Project the voxel to all the images;
  if *in one image the projection of the voxel is completely out of the silhouette* **then**
    The voxel's type is out;
  else
    Its type is on;
else if *the 8 corners of the voxel are in the the visual hull.* **then**
  Project the voxel to all the images;
  if *in all the images the projection of the voxel is completely in the silhouettes* **then**
    The voxel's type is in;
  else
    Its type is on;
else
  The voxel's type is on;

Algorithm 1: The proposed projection test algorithm.

In practice, we use the gift wrapping algorithm [26] to compute the convex hull of the projections of a voxel's 8 corners. Once the exact projection of a voxel on each silhouette image has been computed, our approach needs to determine the relative position of the projection of the voxel to all the silhouettes. Since the projection of the voxel is a convex polygon, our approach evaluates the relative posi-

Fig. 3 Four color images of the Soldier Terra Cotta Warrior sequence.

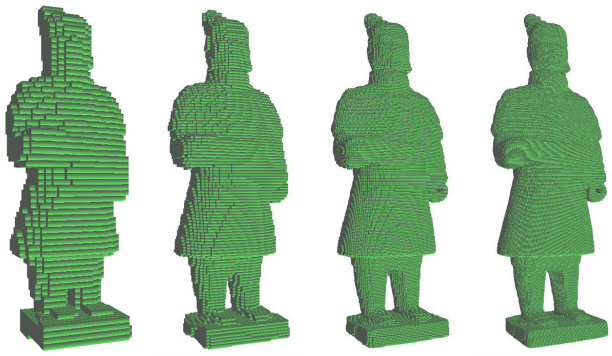


Fig. 4 Visual hull octree structure of the Soldier object. From left to right, the level of detail ranges from 6 to 9 depth levels.

tion of the projection to the silhouette by a scan processor that tracks only the right and the left edges from the top to the bottom of the projection. The evaluation criterion is: (1) If all the pixels in the polygon are inside the object regions, the polygon is in the silhouette; (2) If all the pixels in the polygon are outside the object regions, the polygon is out of the silhouette; (3) Else, the polygon intersects with the silhouette. The octree models of a Soldier Terra Cotta Warrior (see Fig. 3) with different resolutions constructed by our approach are presented in Fig. 4.

3.3 Visual Hull Mesh Extraction by Marching Cubes Algorithm

Once the visual hull octree has been constructed, marching cubes algorithm is applied to extract the visual hull mesh. To do this, the voxel occupancy of a leaf node is encoded into 8 values using the isofunction value of its eight vertices. Since these isofunction values can represent the 3D distance to the visual hull surface, the mesh extracted from the octree structure is very smooth. In Fig. 5 we present two views of the visual hull mesh of the Soldier object and we can appreciate the quality of the visual hull model.

The visual hull will be used in our image-based modeling algorithm in two aspects. On one hand, in order to generate depth maps from multi-view stereo, the depth interval by backprojecting the ray for each pixel into the visual hull

Fig. 5 Two views of the visual hull mesh of the Soldier object extracted from 8-level octree structure.

mesh defines the search range for each pixel, described in Section 4. On the other hand, the visual hull octree structure is carved by a PCST in order to generate a PCSL, described in Section 5.

4 Shape from Multi-View Stereo

The shape information represented by an oriented point cloud is computed from multi-view stereo in this section. Firstly, depth maps are estimated from multi-view stereo efficiently by an expansion-based method. Since the 3D point cloud merged from the depth maps contains outliers and redundant information, the second step is to reject the outliers and downsample the point cloud. Finally, the surface normal of each point in the point cloud is estimated from the positions of the neighbors and the viewing direction of each 3D point is employed to select the orientation of estimated surface normal.

4.1 Expansion-Based Depth Map Estimation

The proposed expansion-based depth map estimation approach is an improvement to the Hernandez et al.'s greedy depth map generation approach [10]. Therefore, we first give a short explanation to the greedy approach. The inputs of the approach are a sequence of calibrated images $I = \{I_0, I_1, .., I_{n-1}\}$ and the visual hull of an object. For each image I_i, the approach selects k neighboring views against which to correlate I_i using robust window matching. For each pixel p in I_i, the approach computes the depth interval from the visual hull of the object which is the backprojected ray of p inside the visual hull. Then reproject the depth interval into selected neighboring views and compute the normalized cross-correlation (NCC) value between an $m \times m$ window centered on p and the corresponding windows centered on the projections in each of the image. For a given

depth interval, its projection into the different images are all related by the epipolar constraint by which all the correlation curves generated by different views can be related into a single coordinate system [27]. Once the correlation curves are computed, the best candidate depth is chosen from them if its NCC value is larger than some threshold $thres1$ for at least two views in the k neighboring views. Note that for each pixel p in I_i, the best candidate depth is chosen to be the value of depth that maximizes NCC value, or none if no valid depth is found. After the best candidate depth is selected, the position of the 3D point corresponding to the pixel p can be computed easily by triangulation method. A detailed description of Hernandez et al.'s greedy depth map estimation approach can be seen in [10].

A drawback of this greedy approach is the computation time since searching the depth value for each pixel from the depth interval defined by the visual hull has a large redundancy of computation. Hernandez et al. speed up the greedy approach by partitioning an image into 3 different resolution layers, computing the depth interval from the visual hull for the lowest resolution layer and from the precedent layer for consecutive layers. In practice, the improvement is about 5 or 6 times faster for well textured images.

The key insight of our approach is to expand from the already recovered shape information in order to obtain shape from multi-view stereo quickly. Specifically, our approach first partitions each image into lots of small windows with fixed size $M \times M$, then computes a depth value for the center pixel of each window using greedy approach. If we find a depth value and its confident value is larger than a threshold $thres2$, the depth value is taken as a reference depth value for the window. In practice, if we compute the 3D positions for all the reference depth values, we can obtain a sparse point cloud with many outliers although only the depth values with high confident value are selected (see Fig. 6 (a) and (b)). Therefore, a median-rejection method is applied for all the reference depth values of an image to reject the obvious outliers (see Fig. 6 (c) and (d)). Since our approach only selects the depth value with a high confident value, there will be many windows without reference depth value, especially for the surface area with little or no texture. Therefore, we compute a reference value for the window without it from its 3×3 neighboring windows, i.e., for a window without a reference depth value, if the number of neighboring windows with it in the 3×3 neighborhood is more than a fixed number a (in all our experimental result, $a = 4$), compute a reference depth value for the window as the median of the depth values of its neighboring windows. This process iterates for 5 times for all the experimental results. In Fig. 6 (e), we can see that the point cloud is more dense after this step. For all the windows of an image, our approach only computes depth value for the pixels of a window with reference depth value. Since each image is a picture of an object and we can expect it to be locally continuous, if the surface is correctly seen and if there is no occlusion, the depth value of the pixels in each window will not be very different from its reference depth value. Therefore, we search the depth values for all the pixels in the window from a depth interval with fixed length d centered at the reference depth along the optical ray. The depth map of an image of the dinoRing sequence provided by the Middlebury benchmark [33] is demonstrated in Fig. 6 (f) which shows that our approach can compute a dense depth map. Our expansion-based depth map estimation algorithm is also described in Algorithm 2.

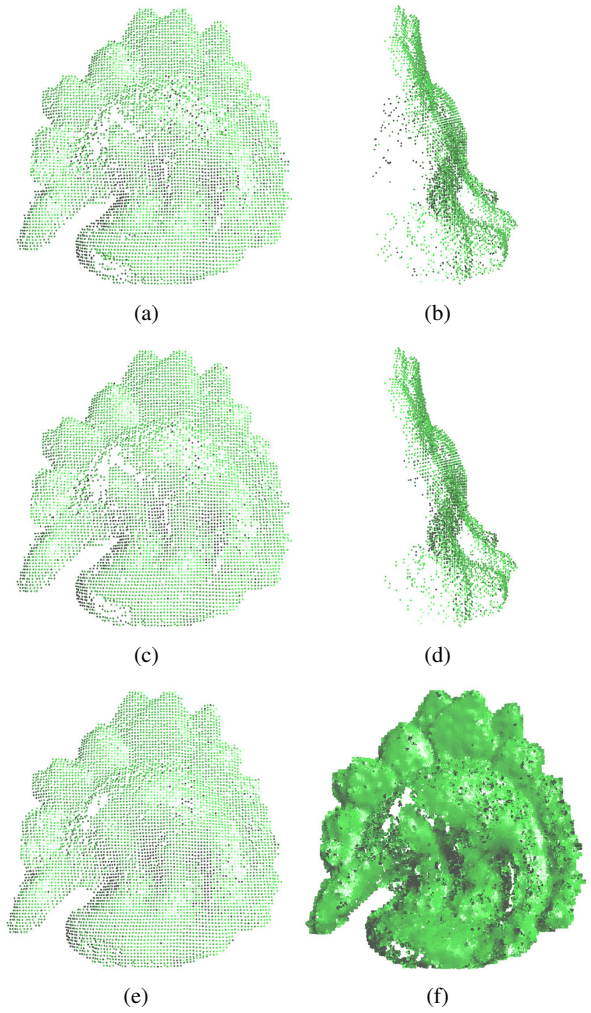


Fig. 6 Expansion-based depth map estimation steps for one image of the dinoRing sequence. (a) and (b) Two views of a sparse point cloud merged from reference depth values. (c) and (d) Two views of the sparse point cloud after median-rejection process. (e) The sparse point cloud after expanding from neighboring windows. (f) The estimated depth maps. From left to right, the number of 3D point in each point cloud is 5868, 5691, 6021 and 128455.

In implementation, the size of the expanding window $M \times M$ depends on the resolution of input image sequence. Typically, for input images with 2000×3000 pixels, the size of the expanding window is 21×21. And the value of $thres2$

```
Input: Calibrated image sequence and the visual hull
Output: Depth map of each image
for each image in imageList do
    for each expanding window do
        Compute a reference depth value by greedy approach;
    end
    Reject the outliers by median-rejection method;
    for the window without reference depth value do
        Compute a reference depth value from its 3×3
        neighboring windows.
    end
    for each expanding window do
        Search the depth value for each pixel in the window
        from the depth interval defined by the reference depth;
    end
end
```

Algorithm 2: The proposed expansion-based depth map estimation algorithm

depends only on how well of the texture of the object. For a well textured object, $thres2 = 3.0$, while for an object with little texture, $thres2 = 2.5$. The value of d depends on both the size of the reconstruction object and the size of the expanding window. Generally, the value of d is from 1% to 2% of the size of the reconstruction object.

Since the depth interval defined by the reference depth is much shorter than the depth interval defined by visual hull, the computation time of the proposed approach is dominated by the reference depth computation step. Typically, the improvement of computation time for an image partitioned into 5×5 windows is around 10 times faster than the greedy approach. The output of this algorithm is depth map for each image. We just merge these depth maps into a point cloud which contains outliers and redundant information. For each 3D point in the point cloud, its confidence value and viewing direction are stored for post-processing.

4.2 Point Cloud Cleaning and Downsampling

The previous subsection outputs a point cloud on the object surface which contains many outliers and redundant information. On one hand, there are many outliers in the point cloud generated for miscorrelation. On the other hand, the point cloud also contains large amounts of redundant information due to duplicate reconstructions of parts of the geometry from multiple views. Therefore, we should reject the outliers and downsample the point cloud before surface reconstruction.

The outliers of the point cloud are rejected by a two-step approach. Firstly, the visual hull of a reconstruction object is incorporated as a constraint to reject 3D points out of the visual hull. Then, we build a voting octee from the estimated point cloud and select a threshold to eliminate miscorrelations. Given a set of point samples S and a maximum

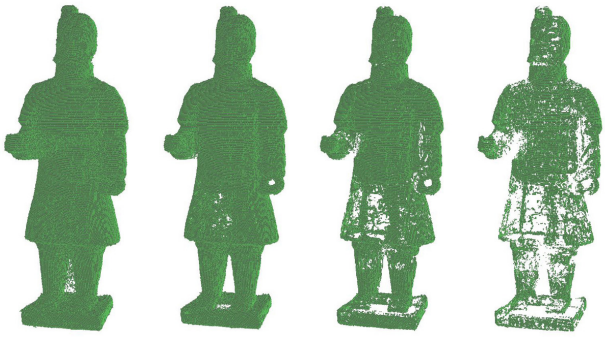


Fig. 7 Voting octree for the Soldier sequence with 10 levels of depth and different thresholds. From left to right, the threshold value is 0, 5, 10, 20.

tree depth VT_d, the voting octree is the minimal octree with the property that every point sample falls into a leaf node at depth VT_d. And we build a voting octree for the point cloud which contains, for each voxel, the sum of the individual correlation scores contained in that voxel. This volume can be seen as a volume of surface probability where a voxel with a high score is very probable to contain the real object surface. Therefore, we threshold the voting octree by eliminating the voxels that have relatively lower score values than a threshold value $thres3$ to add robustness to the correlation approach. For input images with 2000×3000 pixels, typical resolutions of the voting octree are between 10 and 11 levels. The different views of voting octree for the Soldier sequence after binarization with different thresholds are presented in Fig. 7.

To downsample the 3D point cloud, for each node at the maximum depth of the voting octree, we extract the point with largest confidence value in the corresponding voxel. Due to the loss of image space when taking photographs (the object image being smaller than the image size), loss of resolution caused by the size of the correlation window and by the maximum camera baseline, the resolution of a 10-level voting octree is already very high for input images with 2000×3000 pixels. Therefore, for a 10-level voting octree using the point with largest confidence value instead of all the points in a voxel will not reduce the accuracy of the estimated depth maps but decrease the size of the point cloud significantly. These extracted 3D points construct a new point cloud on the object surface with few outliers and smaller scale.

4.3 Surface Normal Estimation

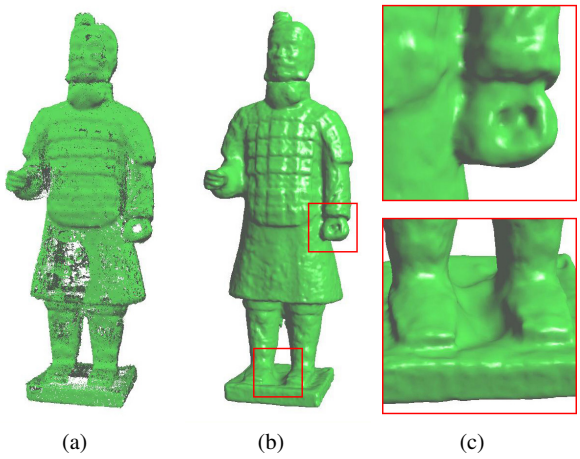
After the outliers are rejected and the size of the point cloud are reduced, we need to estimate exact surface normal for every point. The Principal Component Analysis (PCA) approach [28] is applied to do this work. In this approach, for a 3D point p_i in the point cloud the normal is given as u_i, the

eigenvector associated with the smallest eigenvalue of the covariance matrix of the k-nearest-neighborhood of p_i. In practice, we choose two parameters to define the neighborhood of a given point which are a fixed radius R and a point number N and use a KD-Tree to efficiently compute the k-neighborhood queries. To compute a neighborhood for p_i, firstly, we compute the number of points in a ball with the radius R centering at p_i. If the number is larger than or equal to N, the neighborhood is the ball with radius R. Otherwise, we enlarge the radius until the number of points in the enlarged ball is more than N. And the enlarged ball is taken as neighborhood to compute normal alternatively.

In most cases, the determination of surface normal orientations is not an easy task. However, in our case, we can select the orientation of the surface normal according to the dot product result, ζ, between the viewing direction c_i of the point and the surface normal u_i [17]. If ζ is larger than zero, the direction of surface normal is the same as c_i. Otherwise, the direction is as opposed to c_i.

$$n_i = \begin{cases} u_i & \text{if } u_i \cdot c_i > 0 \\ -u_i & \text{Otherwise} \end{cases} \quad (2)$$

The output of this section is an oriented point cloud on the object surface with few outliers and relatively small scale denoted as PCST. In Fig. 8 (a), an oriented PCST for the Soldier sequence is demonstrated from which we can see most shape information including some minute details such as the face has been correctly recovered. However, there are still a few surface areas of the Soldier object without estimated points since these areas are with no texture or little texture or cannot be seen from any view. On the other hand, one limitation of the PSR method is that it will connect two disconnect regions when there are no samples between these two regions. If we use the PSR approach to triangulate the PCST directly, the surface area with no or little estimated 3D points cannot be correctly recovered, illustrated in Fig. 8 (b) and (c). We can see that this method cannot reconstruct correct structures at the left hand or the support platform of the Soldier object (illustrated by red squares). In the next section, we will apply the silhouette information to solve this problem.



(a) (b) (c)

Fig. 8 Reconstruct the Soldier surface model from the PCST by using PSR method. (a) An oriented PCST. (b) A model reconstructed from the PCST using PSR method. (c) Two zoomed portions of the reconstructed model which are not correctly recovered.

5 Volumetric Stereo and Silhouette Fusion

In practice, shape information of well-textured objects with simple topology can be effectively captured from multi-view stereo. However, it is hard to recover complete surface models for textureless objects or objects with complex topology using texture information only. In this case, silhouette information can be applied to amend the missing shape information from multi-view stereo. A volumetric stereo and silhouette fusion approach is proposed in this section to recover accurate and complete surface estimates. Firstly, another oriented point cloud on the visual hull denoted as PCSL is generated in order to recover the shape information from silhouettes for the surface areas which cannot be captured by texture information. Secondly, the shape information from stereo and silhouette are combined by merging two point clouds (PCST and PCSL) and PSR approach is applied to convert the oriented point cloud both from stereo and silhouette into a triangulated mesh model.

The point cloud from multi-view stereo does not have or has little points in the area of the object surface with no texture or little texture or the area that cannot be seen from any view. In most case, the shape information of these areas can be captured by the silhouette information. To generate an oriented point cloud from silhouette, the key new idea is that we classify the voxels of visual hull octree structure into three types according to their relative position to the points of PCST: (1) Type 1: the voxel contains a 3D point or 3D points of PCST; (2) Type 2: the voxel intervenes the line between a 3D point of PCST and the optical center of the point's reference image; (3) Type 3: all the remaining voxels. In fact, most of these remaining voxels locate at textureless and occluded surface areas (see Fig. 9 (a)). As an oriented point cloud need to be computed at these areas, we extract the vertices and normals of the visual hull mesh in these remaining voxels to construct a PCSL (see Fig. 9 (b)). Our PCSL computation approach is described in Algorithm 3.

Since the voxel carving scheme is prone to errors [29], although most points of PCSL locate at the the textureless and occluded surface areas, there are still a few points on the well-textured surface area which will be taken as outliers if they are far away from the real object surface. However, as the PSR approach can create very smooth surfaces that robustly approximate noisy data, this is not a serious prob-

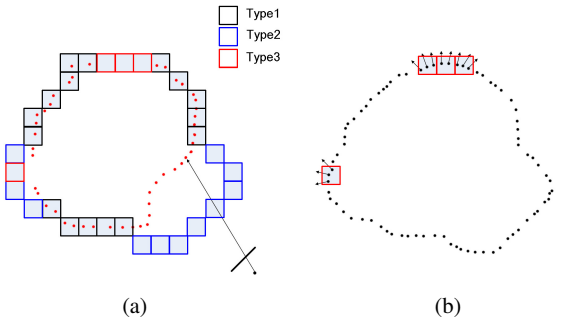


Fig. 9 Description of the algorithm to compute PCSL. (a) Classify the voxels of visual hull octree structure into three types (the red points represent a PCST). (b) Extract the vertices and normals of visual hull mesh in the remaining voxels (the black points represent the visual hull mesh vertices).

Input: Visual hull octree structure, visual hull mesh, and point cloud from stereo
Output: Point cloud from silhouette
for *each voxel of the visual hull octree structure* **do**
  for *each point in the point cloud from stereo* **do**
    if *the point is in the voxel* **then**
      The voxel's type is 1;
      Break;
    else if *the voxel intervenes the line between the point and the optical center of the point's reference image.* **then**
      The voxel's type is 2;
      Break;
    else
      The voxel's type is 3;
  end
end
for *each voxel whose type is 3* **do**
  Extract the vertices and normals of the visual hull mesh in this voxel;
end

Algorithm 3: The proposed PCSL computation algorithm

lem. In practice, we can control the number of these outliers by setting an appropriate level of visual hull octree structure to be carved by a PCST denoted as VH_d. For a PCST computed from 10-level voting octree, VH_d is between 7 to 9 which is tradeoff between computation time and the quality of reconstructed results.

Once the PCSL is computed, it is added to the PCST computed in the previous section to generate a more complete point cloud on the object surface denoted as PCSTSL. And we use freely available package [30] of the PSR method to convert the oriented PCSTSL into a triangulated mesh model. For the Soldier object, a PCSL and a PCSTSL are demonstrated in Fig. 10 (a) and (b) from which we can see that the PCSL can effectively capture the missing shape information from stereo such as the left hand and the support platform of the object although it is not as dense as the

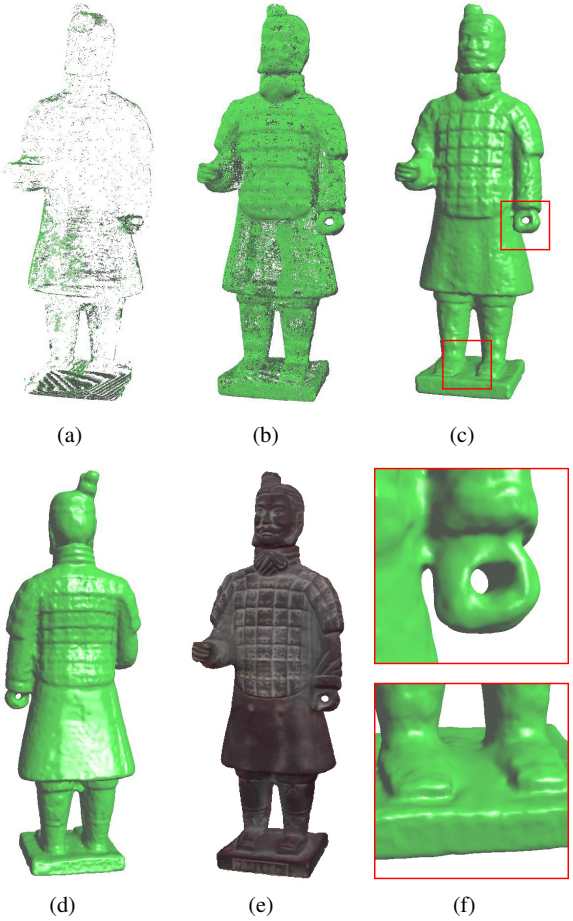


Fig. 10 Volumetric fusion steps for the Soldier sequence. (a) PCSL. (b) PCSTSL. (c) and (d) Two views of the reconstructed model. (e) Textured surface model. (f) Two zoomed portions of the reconstructed model.

PCST. Two views of the reconstructed model of the Soldier object are demonstrated in Fig. 10 (c) and (d). In Fig. 10 (f), two zoomed portions of the reconstructed model which are the left hand and the support platform of the object are presented. And we can appreciate that the previous mentioned problem is solved by adding a PCSL and these two structures are correctly recovered.

6 Experimental Results

To evaluate the contributions of our approach, we demonstrate the reconstructions of several real world objects. Firstly, we apply our approach to reconstruct the Captain Terra Cotta Warrior dataset acquired in our lab with an electronic turntable and a fixed camera. Secondly, the BigHead sequence which is courtesy of [31] and the Skull sequence which is provided by [32] are also recovered, along with comparisons with several state-of-the-art image-based modeling techniques. Finally, we quantitatively evaluate our ap-

Table 1 Characteristics of the datasets used in our experiments.

dataset	*Ima*	*Resolution*	*PSize*	*Vertices*	*Time*
Soldier	36	3088×2056	1253749	150000	74.8
Captain	36	3088×2056	1412236	150000	72.0
BigHead	36	2008×3040	1331808	150000	84.5
Skull	16	1900×1700	578854	180000	67.7
templeRing	47	640×480	732824	316228	20.8
dinoRing	48	640×480	924281	644625	25.2
dinoSparseRing	16	640×480	309504	281146	7.9

proach using the Middlebury benchmark [33] which consists of two objects, a temple and a dinosaur.

Table 1 lists the number of input images (*Ima*), their approximate size (*Resolution*), the number of points of an oriented point cloud as input of PSR approach (*PSize*), the number of vertices of a final model (*Vertices*), and running time in minutes (*Time*) for each dataset. Computation times are dominated by depth maps generation from multi-view stereo step. A typical computation time to compute depth maps from 36 images of 6 Mpixels is about one hour on a Duo E7400 2.80GHz computer, while visual hull computation and Poisson surface reconstruction from mixed point cloud only cost a few minutes.

Table 2 lists running parameters of the proposed algorithm for each dataset, including the number of selected correlation images k, the threshold to select a best candidate depth $thres1$, the size of NCC matching window $m \times m$, the size of expanding window $M \times M$, the threshold to select a reference depth value $thers2$, the depth interval length d, the voting octree level VT_d and its threshold $thres3$, the neighborhood to compute surface normal R,N, and the level of visual hull octree structure to be carved VH_d. See the corresponding sections for details about the meaning and setting method for these parameters.

6.1 Captain Sequence

The Captain object (see Fig. 11) is a 228mm tall, gray, strongly diffuse Terra Cotta Warrior with lots of details and complex topology. The object is placed on a turntable and a sequence of images is taken with a fixed angle step which is 10 degrees. Therefore, the Captain sequence contains 36 images which all have a resolution of 3088×2056 pixels. Our approach requires accurate segmentation of each image into silhouette. To facilitate silhouette segmentation, we use a monochrome background in the setup of image acquisition (see Fig. 3 and Fig. 11). So it is easy to find the object silhouette using standard background subtraction method. Furthermore, our approach also requires that each input image is accurately calibrated and we use an approach similar to [10] to calibrate our system.

A complete reconstruction process for the Captain object is presented in Fig. 12. To generate depth map from

Fig. 11 Four color images of the Captain Terra Cotta Warrior sequence.

multi-view stereo, the correlation for a given pixel is computed with 4 neighboring views for a typical sequence of 36 images. Then a 10-level voting octree is built from the depth maps and many outliers generated for miscorrelation are eliminated by using the visual hull constraint and thresholding the voting octree ($thres3 = 4.0$). The neighborhood that is used for estimating surface normal for each 3D point is defined by $R = 2mm$, $N = 200$. In order to compute a PCSL, a 8-level visual hull octree structure is carved by a PCST. See Table 2 for more details of the reconstruction parameters for the Captain sequence. The reconstructed model (see Fig. 12 (e) and (f)) shows that the proposed approach can not only recover minute details such as the details on the face and the armor, but also capture correct topology of a real object using both stereo and silhouette information.

6.2 BigHead Sequence

The BigHead object (see Fig. 13) is a very well textured object which is quite suitable for stereo reconstruction. However, since the object support table is lack of texture which cannot be reconstructed from multi-view stereo, this part can only be recovered using silhouette information.

Complete reconstruction steps are demonstrated in Fig. 14. Since the body of the BigHead object is well textured, a quite uniform PCST can be computed in this area which can be seen in Fig. 14 (b). Fig. 14 (c) shows that most points in the PCSL are at the support table of the object which is the only textureless part of the object. And the reconstructed model of the BigHead object by PSR method from the PCSTSL (see Fig. 14 (d)) is demonstrated in Fig. 14 (e).

In Fig. 15 we present the comparison between the surface model reconstructed by the proposed algorithm and the model reconstructed by Hernandez et al. [10]. Since Hernandez et al. always use a regularization term in their snake-based approach, some fine details might lose for this reason. In Fig. 15 (c) we can see that our approach can reover more accurate details than their method at the face of the object (illustrated in red square).

Table 2 Running parameters for the datasets used in our experiments. See text for more details.

dataset	k	*thres1*	$m \times m$	$M \times M$	*thres2*	*d(mm)*	VT_d	*thres3*	*R(mm)*	N	VH_d
Soldier	4	0.6	11×11	21×21	3.0	3.0	10	6.0	2.0	200	9
Captain	4	0.6	11×11	21×21	3.0	3.0	10	4.0	2.0	200	8
BigHead	4	0.6	11×11	21×21	3.0	3.0	10	8.0	1.6	200	8
Skull	2	0.6	7×7	21×21	3.0	3.0	10	3.0	1.0	100	7
templeRing	4	0.6	5×5	5×5	3.0	3.0	9	3.9	1.0	100	
dinoRing	4	0.5	5×5	5×5	2.5	4.0	9	4.0	1.8	180	
dinoSparseRing	2	0.5	5×5	5×5	2.5	4.0	9	3.8	1.8	180	

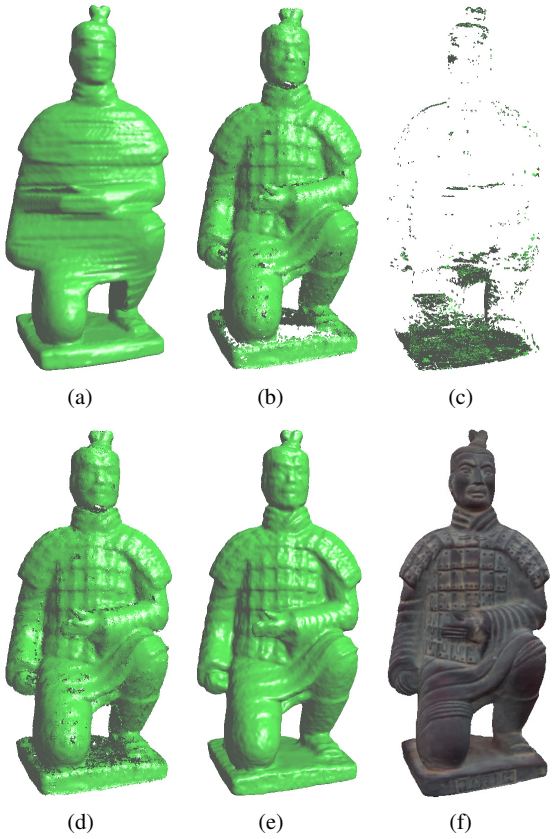


Fig. 12 The reconstruction steps for the Captain sequence. From (a) to (f), visual hull, PCST, PCSL, PCSTSL, reconstructed model before and after texturing mapping.

Fig. 13 Four color images of the BigHead sequence.

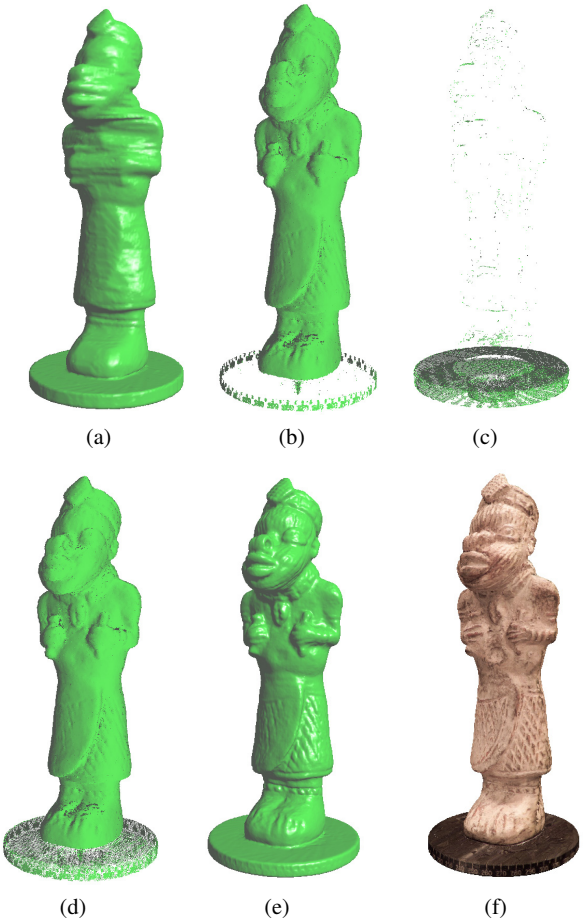


Fig. 14 The reconstruction steps for the BigHead sequence. From (a) to (f), visual hull, PCST, PCSL, PCSTSL, reconstructed model before and after texturing mapping.

6.3 Skull Sequence

The Skull object (see Fig. 16) is a plaster cast of a human skull. The Skull dataset contains 16 images captured on a ring around the object plus an additional 8 images captured on a sparser ring at higher elevation angles. A more detailed description of the Skull data set can be found in [32]. In our case, although the visual hull is computed from all the 24 silhouettes, we just use the 16 images captured on a ring to compute depth maps from multi-view stereo. The recon-

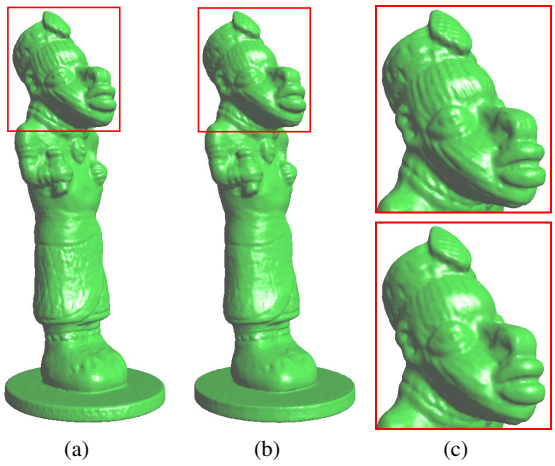


Fig. 15 Comparison with a snake-based method by Hernandez et al. (2004) for the BigHead sequence. (a) The reconstructed model using our method (150000 vertices). (b) The reconstructed model using Hernandez et al.'s method (118344 vertices). (c) Two zoomed images, the top one is ours and the bottom one is theirs.

Fig. 16 Two color images of the Skull sequence.

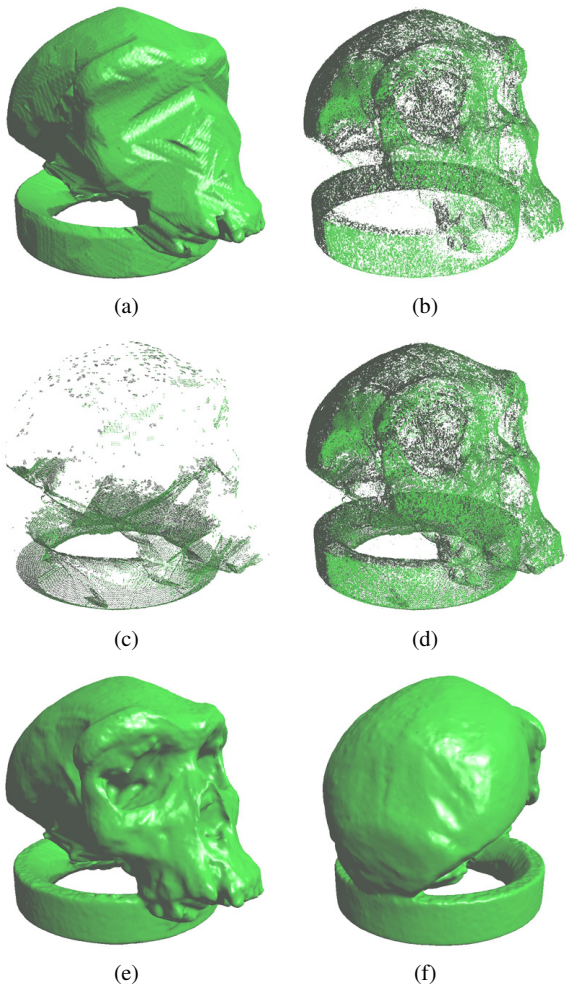


Fig. 17 The reconstruction steps for the Skull sequence. From (a) to (f), visual hull, PCST, PCSL, PCSTSL, two views of the reconstructed model.

struction parameters for the Skull sequence can be seen in Table 2.

In Fig. 17 we present complete reconstruction steps of the Skull model. Starting from the visual hull (Fig. 17 (a)), an oriented PCST is generated from multi-view stereo and an oriented PCSL is computed by carving the visual hull octree structure using the PCST. We can see in Fig. 17 (b) and (c) that the PCSL can effectively capture the missing shape information from stereo such as the inside border of the support and the bottom of the Skull object. After the PCST and the PCSL are mixed to generate a PCSTSL (Fig. 17 (d)), the PSR method is applied to convert the oriented point cloud into a triangulated mesh model (Fig. 17 (e) and (f)) from which we can appreciate the high quality of the recovered surface such as the face and sutures of the object.

In Fig. 18 (a) and (b) we present the comparison between the surface model reconstructed by the proposed algorithm and the model reconstructed by Furukawa et al. [34]. One limitations of their multi-view stereo method is that some concavities may be too deep to be carved away by the graph cuts. We can see in Fig. 18 (c) that the nose of the Skull object is not correctly recovered for its limitations. And we can see in Fig. 18 (a) that our method not only recovers the deep concavities such as the nose of the Skull object, but also outperforms Furukawa et al.'s method in reconstructing a small concave structure near the nose of the object.

In Fig. 18 (a) and (d) we present the comparison between the surface model reconstructed by the proposed algorithm and the model reconstructed by Goesele et al. [16]. We can see in Fig. 18 (d) that there are many holes in textureless areas of the Skull model reconstructed by Goesele et al.. However, our method recovers a complete and accurate model since we use shape information both from stereo and silhouette.

6.4 TempleRing Sequence

The temple and dino datasets (see Fig. 19) are provided by the Middlebury benchmark [33]. The images of the datasets were captured by using the Stanford spherical gantry and a CCD camera with a resolution of 640×480 pixels attached

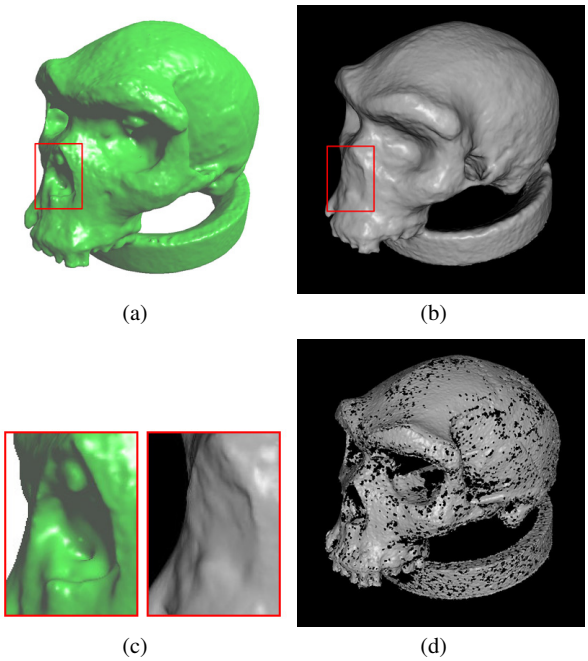


Fig. 18 Comparisons with other image-based modeling techniques for the Skull sequence. (a) The reconstructed model using our method. (b) The reconstructed model using a method by Furukawa et al. (2006). (c) Two zoomed portions of the Skull models recovered by our and Furukawa et al.'s methods. (d) The reconstructed model using a method by Goesele et al. (2006).

Fig. 19 Two objects of the Middlebury benchmark, temple and dino.

to the tip of the gantry arm. From the resulting images, three datasets were created for each object, corresponding to a full hemisphere, a single ring around the object, and a sparsely sampled ring.

Our method is well-suited for viewpoints arranged in a ring setup since the selecting correlation views is straightforward. Thus we perform our evaluation on the templeRing, dinoRing, and dinoSparseRing datasets. The reconstruction parameters for three datasets are presented in Table 2. And the evaluation results of our algorithm for the three datasets are shown in Table 3 which are evaluated on the accuracy (Acc) and completeness (Comp) of the final result with respect to a ground truth model, as well as processing time (Time). We highlight the best performing algorithm for each metric. Please note that the computation

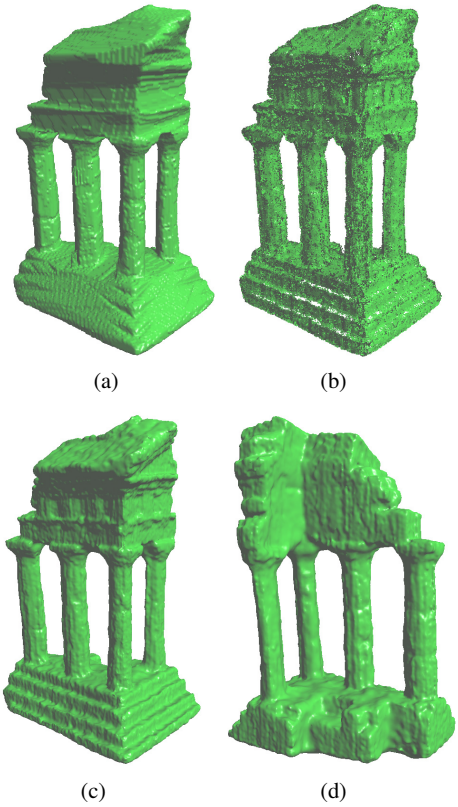


Fig. 20 The reconstruction steps for the templeRing sequence. From (a) to (d), visual hull, oriented PCST, and two views of the reconstructed model.

time of our approach presented in Table 3 is a standard time which is a little different from the time presented in Table 1. And the format of computation time in Table 3 is as hour:minute:second. According to Table 3, we can see that our approach is comparable to current state-of-the-art techniques, especially for the dinoRing dataset. Additionally, our approach is also one of the most efficient methods among non-GPU methods for the three datasets and our computation time is very close to that of Bradley et al.'s method [17].

The temple object is a plaster reproduction of an ancient temple which contains lots of geometric structure and texture. As it is very well textured, a quite uniform and dense point cloud can be computed using our shape from stereo approach (see Fig. 20 (b)). Since most of the shape information can be captured from multi-view stereo, there is no need to generate a PCSL. Fig. 20 (c) and (d) present two views of the reconstructed temple model which shows that our method is able to reconstruct objects of non-trivial topology and with fine details.

Table 3 Evaluation results for the Middlebury benchmark datasets.

	templeRing			dinoRing			dinoSparseRing		
	Acc	*Comp*	*Time*	*Acc*	*Comp*	*Time*	*Acc*	*Comp*	*Time*
Proposed approach	0.61	98.3%	19:26	0.38	99.4%	23:31	0.54	95.5%	7:24
Furukawa(2008)	0.47	99.6%	3:31:01	0.28	99.8%	5:00:34	0.37	99.2%	2:31:37
Furukawa(2007) [12]	0.55	99.1%	6:02:40	0.33	99.6%	9:04:00	0.42	99.2%	3:44:00
Furukawa(2006) [34]	0.58	98.5%	10:00:00	0.42	98.8%	15:00:00	0.58	96.9%	10:00:00
Bradley [17]	0.57	98.1%	11:21	0.39	97.6%	23:25	0.38	94.7%	7:06
Vu [35]	0.45	99.8%	1:33	0.53	99.7%	1:26			
Hernandez [10]	0.52	99.5%	2:00:00	0.45	97.9%	2:06:00	0.60	98.5%	1:46:00
Goesele [17]	0.61	86.2%	34:00:00	0.46	57.8%	41:56:00	0.56	26.0%	14:03:12

6.5 DinoRing Sequence

The dino object is a white, strong diffuse plaster dinosaur model without obvious texture. However, our shape from multi-view stereo algorithm still reconstructs geometry of most portion of the surface (see Fig. 21 (b)). Fig. 21 (c) and (d) shows our final reconstruction result of the dinoRing dataset and we can see that details such as the foot of the dinosaur have been ideally reconstructed. As our expansion-based depth map estimation algorithm outputs dense and accurate depth map efficiently which is crucial to the final model, at the moment these results are submitted, the accuracy measurement of the dinoRing dataset ranks top 3 and the completeness measurement ranks top 6. Furthermore, the computation time of our approach is also very competitive among non-GPU methods. See Table 3 for more details.

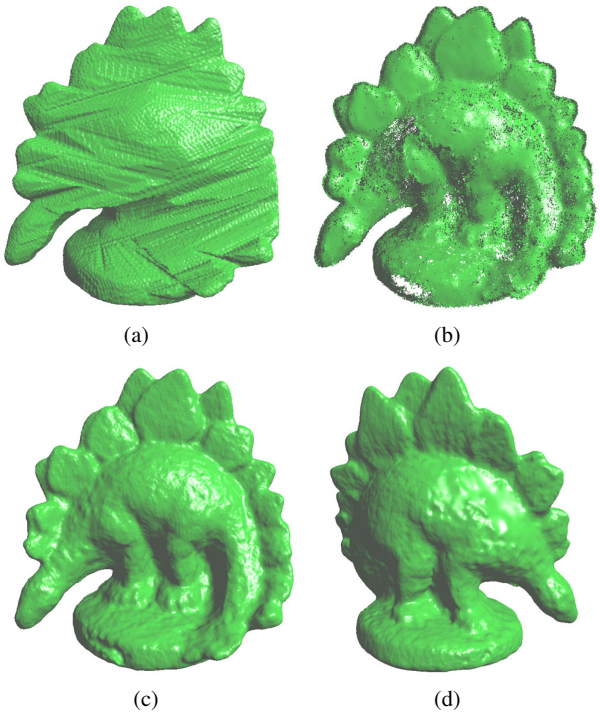


Fig. 21 The reconstruction steps for the dinoRing sequence. From (a) to (d), visual hull, oriented PCST, and two views of the reconstructed model.

6.6 DinoSparseRing Sequence

Fig. 22 shows that reconstruction process for the dinoSparseRing sequence. As this dataset just contains 16 images while our algorithm requires that each surface point is seen in at least three views (a reference view and at least two neighboring views), the recovered point cloud is much sparser than that of the dinoRing sequence which is shown in Fig. 22 (b). Fig. 22 (c) and (d) shows the final model.

7 Conclusions

We have developed a novel algorithm for 3D real object surface model reconstruction from a sequence of calibrated images by fusing the shape information obtained from multi-view stereo and silhouette. The algorithm first computes an oriented point cloud on the object surface from calibrated images using texture information. Then, the silhouette information is applied to recover the textureless and occluded surface areas. After the merging of the shape information obtained from texture and silhouette, the PSR approach is applied to recover a complete and accurate triangulated mesh model. The experimental results with several real datasets demonstrate that the proposed approach can produce complete and accurate surface models. According to the evaluation results of the Middlebury benchmark, the proposed approach is comparable to the state-of-the-art image-based modeling techniques.

Future work should mainly focus on improving the accuracy of the proposed approach which can be realized by depth map subpixel optimization and depth map cleaning. On one hand, a subpixel optimization step can be applied to improve the accuracy of the estimated depth maps. On the other hand, more effective approaches can be applied to clean the depth maps without rejecting the correct shape information such as using the visibility constraint [12], disparity gradient limit constraint [36] and so on.

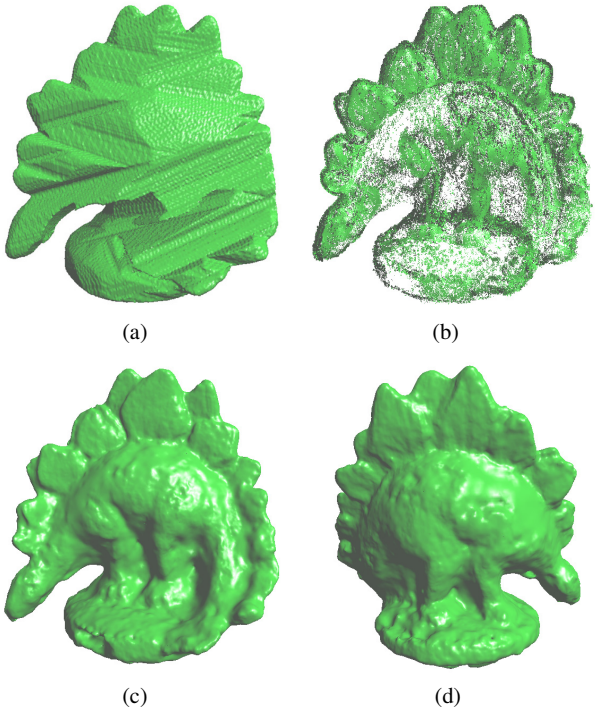


Fig. 22 The reconstruction steps for the dinoSparseRing sequence. From (a) to (d), visual hull, oriented PCST, and two views of the reconstructed model.

8 ACKNOWLEDGMENTS

This project is partially supported by Natural Science Foundation of China (NSFC50805031) and Science & Technology Research Projects of Shenzhen (No. JC200903120184A, ZYC200903230062A), Foundation of the State Key Lab of Digital Manufacturing Equipment & Technology (No. DMETKF2009013).

References

1. Prados, E., Faugeras, O.: Perspective shape from shading and viscosity solutions. ICCV 2, 826-831 (2003)
2. Szeliski, R.: Rapid octree construction from range sequences. Comput. Vis. Graph. Image Process 58(1), 23-32 (1993)
3. Yemez, Y., Schmitt, F.: 3D reconstruction of real objects with high resolution shape and texture. Image Vis. Comput. 22(13), 1137-1153 (2004)
4. Kazhdan, M., Bolithp, M., Hoppe, H.: Poisson surface reconstruction. Eurographics Symposium on Geometry Processing, 61-70 (2006)
5. Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szeliski, R.: A comparison and evaluation of multi-view stereo reconstrucion algorithms. CVPR 1, 519-526 (2006)
6. Vogiatzis, G., Torr, P., Cipolla, R.: Multi-view stereo via volumetric graph-cuts. CVPR, 391-398 (2005)
7. Sinha, S.N., Mordohai, P., Pollefeys, M.: Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh. ICCV, (2007)
8. Vogiatzis, G., Hernandez, C., Torr, P.H.S., Cipolla, R.: Multi-view stereo via volumetric graph-cuts and occlusion robust photo-consistency. IEEE Trans. on PAMI 29(12), 2241-2246 (2007)
9. Kutulakos, K., Seitz, S.M.: A theory of shape by space carving. IJCV 38(3), 199-218 (2000)
10. Hernandez, C., Schmitt, F.: Silhouette and stereo fusion for 3d object modeling. Computer Vision and Image Understanding 96(3), 367-392 (2004)
11. Pons, J., Keriven, R., Faugeras, O.: Modelling dynamic scenes by registering multi-view image sequences. CVPR, 822-827 (2005)
12. Furukawa, Y., Ponce, J.: Accurate, Dense, and Robust Multi-View Stereopsis. CVPR, (2007)
13. Habbecke, M., Kobbelt, L.: A surface-growing approach to multi-view stereo reconstruction. CVPR, (2007)
14. Goesele, M., Snavely, N., Curless, B., Hoppe, H., Seitz, S.M.: Multi-view stereo for community photo collections. ICCV, (2007)
15. Narayanan, P., Rander, P., Kanade, T.: Constructing virtual worlds using dense stereo. ICCV, 3-10 (1998)
16. Goesele, M., Curless, B., Seitz, S.M.: Multi-view stereo revisited. CVPR, 2402-2409 (2006)
17. Bradley, D., Boubekeur, T., Heidrich, W.: Accurate multi-view reconstruction using robust binocular stereo and surface meshing. CVPR, (2008)
18. Liu, Y., Cao, X., Dai, Q., Xu, W.: Continuous depth estimation for multi-view stereo. CVPR, 2121-2128, (2009)
19. Laurentini, A.: The Visual Hull Concept for Silhouette Based Image Understanding. IEEE Trans. on PAMI 16(2), 150-162 (1994)
20. Matusik, W., Buehler, C., McMillan, L.: Polyhedral Visual Hulls for Real-Time Rendering. Proc. 12th Eurographics Workshop on Rendering, 115-125 (2001)
21. Tarini, M., Callieri, M., Montani, C., Rocchini, C., Olsson, K., Persson, T.: Marching Intersections: An Efficient Approach to Shape-from-Silhouette. In Proceedings of VMV, 283-290 (2002)
22. Lorensen, W.E., Cline, H.E.: Marching Cubes: A High Resolution 3d Surface Construction Algorithm. SIGGRAPH 21, 163-169 (1987)
23. Song, P., Wu, X., Wang, M.Y.: A Robust and Accurate Method for Visual Hull Computation. IEEE International Conference on Information and Automation, 784-789 (2009)
24. Potmesil, M.: Generating Octree Models of 3d Objects from Their Silhouettes in a Sequence of Images. Computer Vision, Graphics, and Image Processing vol.40, 1-29 (1987)
25. Borgefors, G.: Distance Transformations in Digital Images. Computer Vision, Graphics, and Image Processing vol.34, 344-371 (1986)
26. Jarvis, R.A.: On the Identification of the Convex Hull of a Finite Set of Points in the Plane. Information Processing Letters vol.2, 18-21 (1973)
27. Hernandez, C., Schmitt, F.: Multi-stereo 3d object reconstruction. 3DPVT, 159-166 (2002)
28. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points. SIGGRAPH, 71-78 (1992)
29. Yemez, Y., Wetherilt, C.J.: A volumetric fusion technique for surface reconstruction from silhouette and range data. Computer Vision and Image Understanding, 105(1):30-41 (2007)
30. Poisson Surface Reconstruction package. `http://www.cs.jhu.edu/~misha/Code/PoissonRecon/`
31. BigHead sequence. `http://www.tsi.enst.fr/3dmodels/`
32. 3D photography datasets. `http://www-cvr.ai.uiuc.edu/~yfurukaw/research/mview.`
33. Dino and Temple datasets. `http://vision.middlebury.edu/mview/`
34. Furukawa, Y., Ponce, J.: Carved Visual Hulls for Imaged-Based Modeling. ECCV vol.1, 564-577 (2006)
35. Vu, H., Keriven, R., Labatut, P., Pons, J.P.: Towards high-resolution large-scale multi-view stereo. CVPR, (2009)
36. Lhuillier, M., Quan, L.: Match propagation for image-based modeling and rendering. IEEE Trans. on PAMI 24(8), 1140-1146 (2002)