IPSJ Journal Vol. 49 No. 4 1234–1252 (Apr. 2008)

Regular Paper

Multi-viewpoint Silhouette Extraction with 3D Context-aware Error Detection, Correction, and Shadow Suppression

Shohei Nobuhara,[†1] Yoshiyuki Tsuda,[†1,∗1]
Iku Ohama[†1,∗2] and Takashi Matsuyama[†1]

This paper presents a novel approach for simultaneous silhouette extraction from multi-viewpoint images. The main contribution of this paper is a new algorithm for 1) 3D context aware error detection and correction of 2D multi-viewpoint silhouette extraction and 2) 3D context aware classification of cast shadow regions. Our method takes both monocular image segmentation and background subtraction of each viewpoint as its inputs, but does not assume they are correct. Inaccurate segmentation and background subtraction are corrected through our iterative method based on inter-viewpoint checking. Some experiments quantitatively demonstrate advantages against previous approaches.

1. Introduction

This paper is aimed at the problem of a simultaneous silhouette extraction of multi-viewpoint images for 3D shape reconstruction. In recent years, the importance of silhouette extraction has increased since many 3D shape reconstruction algorithms [1–8] utilize the visual hull of the object given by the "shape from silhouette" (or SFS in short) method [9]. These algorithms use visual hulls as their initial estimate, and then refine them based on multi-viewpoint textures and/or other reconstruction cues.

However, the multi-viewpoint environment involves new difficulties which are not present in single viewpoint situation. The first difficulty lies with the "AND" operation of the SFS process. In the SFS process, each 3D point in the scene is labeled as a part of the object if its projection lies in the silhouette on all the camera views. This indicates that only one view having errors in its silhouette can easily spoil the quality of the visual hull even if all the other silhouettes are perfect.

The second difficulty comes from the increased chance of having similar colors in the object and the background. In silhouette extraction for 3D shape reconstruction, it is reasonable and widely assumed that we can use the background images of every viewpoint and can apply "background subtraction" individually. Here the most fundamental difficulty in the monocular silhouette extraction is ambiguity due to the comparison between the similar foreground and background colors. In the multi-viewpoint environment this problem can be more difficult than the single viewpoint situation. This is because each 3D point on the object surface can be captured from different viewpoints and should have reasonable differences between each background image. Hence if the object has a color which is completely different from the background of one viewpoint but is similar to that of another view, extraction of silhouettes can be difficult and only one error on a single view corrupts the visual hull as we mentioned before. This problem can be seen in segmentation based approaches as well. For example, suppose we try to obtain a segmentation of the image shown in the top row of Figure 1. The top row of Figures 1(a) and (b) show the same object captured from different viewpoints. The goal is to find a parameter which gives reasonably large segments without overlapping the boundary between the object and the background. The bottom row of Figure 1 shows the enlarged images of the rectangle areas in the two views shown in the top row of Figure 1. Here we can observe that circular areas of (a) have no differences in pixel values between the object and the background while these areas are not observed as a part of the object boundary in (b). This suggests that it is hard to determine the exact object/background boundary based on the viewpoint of (a) alone. Hence we assume that we cannot obtain the perfect segmentation which separates the object and the background without overlapping based on monocular image segmentation techniques.

To overcome these difficulties, we propose a new algorithm which (1) does not rely on the exact segmentation given by single viewpoints individually and (2) does not rely on clear differences between the object and background colors at

†1 Kyoto University
∗1 Presently with SANYO Electric Co., Ltd
∗2 Presently with Panasonic Corporation

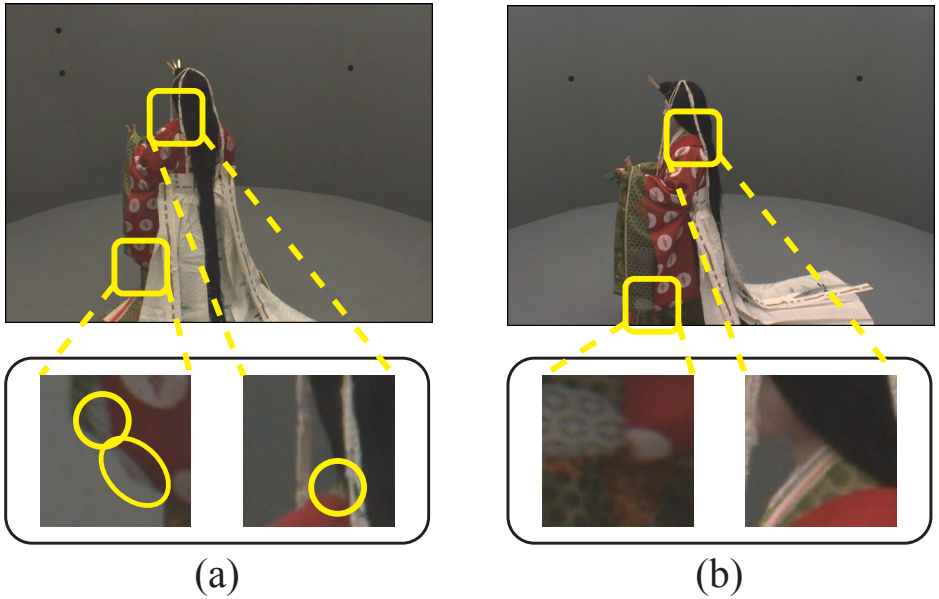


Fig. 1 Examples of object regions with the same colors of the background. (a) and (b) show images of an object captured from different viewpoints. The circular areas in (a) have no differences in pixel values between the object and the background region while these areas are not observed as a part of the object boundary in (b).

every viewpoint. The key idea of our method is a scheme to detect and correct errors on silhouette boundaries based on an inter-viewpoint checking of consistencies on silhouettes. We call this 3D geometry based consistency *3D context*. Our algorithm uses results of single viewpoint segmentation and background subtraction which contain errors as its initial inputs, and refines them to achieve better silhouettes. In addition to this contribution, we also introduce a shadow suppression scheme which utilizes a calibrated multi-viewpoint environment as well.

The rest of this paper is organized as follows. We first review related studies in Section 2, then we describe our algorithm in Section 3, and evaluate it against other approaches in Section 4. We conclude our paper and discuss possible future work in Section 5.

2. Related Work

For multi-viewpoint silhouette extraction, various approaches which utilize multi-viewpoint environment have been proposed so far [10] [11] [12] [13]. For example, Zeng and Quan proposed a method which uses an inter-viewpoint constraint and 2D segmentation of captured images [12]. Their method realized silhouette extraction without using background images. Goldlücke and Magnor proposed a method which realized simultaneous 2D segmentation and 3D reconstruction based on graph-cuts [13]. However they does not have any explicit error detection and correction schemes based on 3D geometry while our method does.

While many studies have been done for shadow suppression on the silhouette estimation so far, most of them focus on the monocular environment. For example, one of the most simple and effective approaches is intensity and chromaticity based categorization of shadow regions [14]. In this approach, each pixel is categorized as shadow region if it is similar to the background pixel in chromaticity but different in intensity. This approach provides significant improvement in comparison with a simple background subtraction using a certain threshold. However, this 2D, pixel-wise silhouette extraction does not consider the 3D context of each pixel. They can suppress real cast shadows on the floor, but they also suppress self shadow regions on the object surface and darkly textured regions like black hair. On the other hand, our algorithm utilizes knowledge about the 3D environment and can eliminate the mis-suppression of shadow-like regions on the object surface.

3. Algorithm

Before describing the algorithm we first clarify the inputs and assumptions of our algorithm. The inputs of our algorithm are:

- multi-viewpoint images of the object,
- multi-viewpoint background images (images of the scene without the object), and
- camera calibration parameters.

Together with these inputs we also assume the following:

- The monocular segmentation of each image can include mis-segmented pixels

even with a simple background due to the color ambiguity, as shown in Figure 1.
- The monocular background subtraction of each image can include error pixels, as described in Section 1.
- We know the 3D geometry of the floor plane and the cast shadows of the object appear only on it.

The last floor assumption is reasonable for 3D shape reconstruction based on the "shape from silhouette" (SFS) algorithm [9]. This is because conventional voxel-based SFS implementations require a certain bounding box which encages the object, and it is adequate to use the studio floor as the bottom of the bounding box.

Using these inputs and assumptions, the goal is to obtain the object silhouettes for every viewpoint which are "consistent" with each other. Through this paper, we call the silhouettes consistent if they satisfy the following three constraints based on the 3D geometry, 2D color, and background subtraction:

- **Intersection constraint (IC) :** Projection of the visual hull which is computed from silhouettes on every viewpoint should be equal to the silhouette on each viewpoint.
- **Projection constraint (PC) :** Projection of the visual hull should have an outline which matches with apparent edges of captured image from each viewpoint.
- **Background subtraction constraint (BC) :** The sum of differences of pixel values between the background and captured image in the projection of the visual hull should be greater than a certain threshold.

Note the first two constraints were originally proposed by Zeng and Quan [12]. To obtain silhouettes which satisfy them, we model the observed images by decomposing them into (1) object regions, (2) cast shadow regions, and (3) background regions. We denote this by $\mathsf{Img} \rightarrow \mathsf{Obj} + \mathsf{Cs} + \mathsf{Bg}$. Based on this modelling, we introduce a two-step approach as illustrated in Figure 2. The first step extracts $\mathsf{Obj} + \mathsf{Cs}$ from Img and the second step extracts Obj from $\mathsf{Obj} + \mathsf{Cs}$.

In what follows, we denote the number of cameras by N, the i-th camera by C_i, the silhouette on C_i at t-th iteration step by $\mathrm{Sil}_i(t)$, the segmented image on C_i at t-th iteration step by $\mathrm{Seg}_i(t)$, the visual hull computed with silhouettes at t-th step by $\mathrm{VH}(t)$, the projection of $\mathrm{VH}(t)$ on C_i by $\mathrm{Svh}_i(t)$.

3.1 Silhouettes and shadow extraction with error detection and correction

First, we introduce an algorithm which extracts both object regions and cast shadow regions ($\mathsf{Obj} + \mathsf{Cs}$) from multi-viewpoint images (Img). To achieve this extraction, we utilize two constraints IC and PC. As described in [12], these two constraints produce a set of silhouettes such that each of them is NOT the background region Bg. That is, silhouettes given by these constraints cannot suppress shadows cast by the object itself because cast shadows on the floor also satisfy IC and PC. In other words, IC and PC enable us to extract $\mathsf{Obj} + \mathsf{Cs}$ from Img. We employ an iterative process as shown in Figure 2 in which we carve a silhouettes per segments so that they satisfy these constraints. The key point of this algorithm is an error detection and correction scheme which utilizes the multi-viewpoint environment. Note here that our iterative method starts with initial segmentations with errors, but errors are corrected through the iterations at the "error detection" part (described later).

3.1.1 Segmentation

In this algorithm we utilize a monocular image segmentation which can re-segment its segments into smaller ones without changing the original outlines of the segments as illustrated in Figure 3. Preserving the original outline of each segment on its re-segmentation is a key point to assure the convergence of the algorithm as discussed later. As a method which satisfies this requirement we introduce the following algorithm.

Step 1. Do pixel labeling to group neighboring pixels with the same value.
Step 2. Apply hierarchical clustering to merge neighboring pixel groups according to the similarity of their colors. Here we introduce a threshold to suppress image noise and merge two neighboring pixel groups if the difference of the average color is smaller than this threshold.
Step 3. Use a clustering result at a certain level of the hierarchy as the initial segmentation, and use the result in the lower level of the hierarchy to obtain the re-segmentation of each segment.

While this is not the only one nor the best method which satisfies the requirement, the experiments show that our method can estimate the object silhouettes even

with this simple algorithm. In the experiments we used a threshold value of 5 at Step 2.

3.1.2 Silhouette carving

In each iteration step, we carve a silhouette so that it satisfies PC. Here we define that a silhouette satisfies PC if it is composed by a set of segments. So we define our carving algorithm as follows:

- For each segment of the captured image,
  - If any portion of the segment is located outside the silhouette, carve the whole segment region from the silhouette.

This operation returns the maximum set of segments each of which is included in the silhouette completely. We denote this operation by $\mathrm{Carve}(\mathrm{Sil}_j(t), \mathrm{Seg}_j(t))$, where $\mathrm{Sil}_j(t)$ and $\mathrm{Seg}_j(t)$ denote the silhouette and segmented image of Camera j at t-th iteration respectively. Figure 4 illustrates this operation. Note here that this operation is done for all segments, and the order of selection does not affect the result.

3.1.3 3D Context aware error detection and correction

Suppose we have a set of silhouettes which satisfies IC. If we carve a silhouette Sil_j of camera C_j so that it satisfies PC, the visual hull will also be carved and its projection on each viewpoint will be equal to or smaller than the original silhouette which satisfies IC. For example, if we carve the black area of CAM_1 in Figure 5, the corresponding volume of the visual hull is also carved, and it is observed as the removal of the gray area on CAM_2. That is, carving one of the intersection consistent silhouettes makes it projection consistent, but breaks intersection consistency between other silhouettes. Here, if the changes between the projection of the visual hull $\mathrm{Svh}_i(t)$ and the original silhouette $\mathrm{Sil}_i(t)$ are *acceptable* on the other viewpoints, we take the projections of visual hull as new silhouettes. We use BC to determine if it is acceptable or not (described below). This process makes the silhouettes satisfy IC again, and PC on Sil_j as well.

If the changes are not acceptable in terms of BC, it is clearly correct to regard that the last carving of the silhouette is wrong, *i.e.*, segments carved in the last $\mathrm{Carve}(\mathrm{Sil}_j(t), \mathrm{Seg}_j(t))$ operation are too large and they contain not only the background but also object regions. This indicates that we need re-segmentation of segments in question into smaller segments as described in Section 3.1.1, and we can retry to check if re-carving is acceptable or not.

For error detection we define the function

$$\mathrm{IsAcceptable}(\mathrm{Sil}_i(t), \mathrm{Svh}_i(t)) = \begin{cases} \mathit{true} & \sum_p |\mathrm{Img}(p) - \mathrm{Bg}(p)| < \mathrm{threshold}, \\ \mathit{false} & \text{otherwise}, \end{cases} \quad (1)$$

where p is a pixel such that $p \in \left(\mathrm{Sil}_i(t) \cap \overline{\mathrm{Svh}_i(t)}\right)$. That is, p denotes a pixel which belongs to $\mathrm{Sil}_i(t)$ but not to $\mathrm{Svh}_i(t)$. $\mathrm{Img}(p)$ and $\mathrm{Bg}(p)$ denote the intensities of the captured and background images at p respectively. This function verifies the changes between the projection of visual hull $\mathrm{Svh}_i(t)$ and the original silhouette $\mathrm{Sil}_i(t)$ if it satisfies BC or not.

3.1.4 Iterative algorithm

Using the constraints and functions defined above, we introduce the following algorithm which extracts Obj + Cs from Img on every viewpoint as illustrated in Figure 2.

Step 0 Let all the silhouettes on every viewpoint be equal to the entire region of the image. Then we compute the visual hull with these silhouettes and project it onto each viewpoint. We call these projections $\mathrm{Sil}_i(0), i = 1, \ldots, N$ (Figure 6). Here, we have a set of multi-viewpoint silhouettes which satisfies IC, and start iterating with $t = 1$.

Step 1 Choose the next camera C_i, which is not chosen in the previous $N - 1$ selections.

- **Step t.1** Let $\mathrm{Sil}_i(t) := \mathrm{Carve}(\mathrm{Sil}_i(t-1), \mathrm{Seg}_i(t-1))$. $\mathrm{Carve}(\cdot)$ produces silhouette which satisfies PC. Figures 7 and 4 illustrate this operation in $t = 1$ and $t > 1$. Then, in other viewpoints $C_j (j \neq i)$, use previous silhouettes as: $\mathrm{Sil}_j(t) := \mathrm{Sil}_j(t-1), j \neq i$.
- **Step t.2** Compute the visual hull $\mathrm{VH}(t)$ using silhouettes $\mathrm{Sil}_k(t)$ and its projections $\mathrm{Svh}_k(t)$ where $k = 1, \ldots, N$.
- **Step t.3** If there are no differences between $\mathrm{Svh}_k(t)$ and $\mathrm{Sil}_k(t), k = 1, \ldots, N$, quit the iteration and go to **Step 2**. Here, each $\mathrm{Sil}_k(t)$ satisfies PC, IC, and BC.
- **Step t.4** Evaluate the differences between $\mathrm{Svh}_k(t)$ and $\mathrm{Sil}_k(t)$ by $\mathrm{IsAcceptable}(\cdot)$, where $k = 1, \ldots, N$. If the number of views on which the

function returns *true* is greater than a certain threshold, let $\mathrm{Sil}_k(t) := \mathrm{Svh}_k(t), k = 1, \ldots, N$ and go to Step 1 with incrementing the iteration counter $t := t + 1$. Otherwise, re-segment $\mathrm{Seg}_i(t)$ and go to **Step *t*.1** with $t := t + 1$.

Step 2 Output $\mathrm{Sil}_k(t), k = 1, \ldots, N$ as the silhouettes including cast shadow regions. These silhouettes satisfy PC, IC, and BC.

This algorithm has a nested, double loop structure. The outer loop selects a camera C_i from $C_1, \ldots, C_N$, and the inner loop carves the silhouette of C_i, *i.e.* $\mathrm{Sil}_i(t)$ by PC and the other silhouettes $\mathrm{Sil}_j, j \neq i$ by IC while updating the segmentation of C_i, *i.e.* $\mathrm{Seg}_i(t)$. Note here that t serves as a global counter and is not initialized on quitting the inner loop in order to describe the update history of the silhouettes and segmentations uniquely. This iterative algorithm can remove Bg regions in theory. We use the resultant Obj + Cs as the input of the next algorithm.

3.2 3D context aware cast shadow removal

Let Scs_j denote the silhouette on C_j given by the algorithm described in the previous section. As described above, Scs_j includes both Obj and Cs regions. The goal of the algorithm we introduce in this section is the removal of Cs from $\mathrm{Scs}_j, j = 1, \ldots, N$ which appear on the floor as assumed in the beginning of Section 3. The key point here is a 3D geometry based constraint on the removal described by GeometricCheck(f, C) in Section 3.2.2.

Suppose we have the visual hull Vcs computed from $\mathrm{Scs}_j, j = 1, \ldots, N$. In this computation, we assume that we use a bounding box whose bottom is equal to the floor of the capturing studio as described above. Let the floor plane be $z = 0$. Our algorithm categorizes the surface of Vcs into Obj region, Cs region, or part of the floor. In this categorization process, our algorithm needs to explore the surface of Vcs. So we use a triangular surface mesh model as the data structure of Vcs for simplicity. Let f denote a triangle of Vcs, F_{Obj} the set of triangles categorized as Obj, F_{Cs} the set of triangles categorized as Cs, and F_{floor} the set of triangles categorized as part of the floor.

3.2.1 Silhouette generation

Once we can categorize triangles into these types, we can obtain a silhouette of viewpoint C_j which contains Obj only as follows:

(1) Initialize the silhouette and depth buffer of C_j. Let all pixels in the silhouette buffer be background pixels, and all pixels in the depth buffer be ∞.

(2) For each triangle f in V_{Cs},

2.1 Project f into C_j, and denote the corresponding area in C_j by P.

2.2 If the depth of f is smaller than those of the corresponding depth buffer area, update the depth buffer by the depth of f. Otherwise, go to the next triangle.

2.3 If f is in F_{Obj}, label the pixels in P as silhouette pixels. Otherwise, label them as background pixels.

3.2.2 Cast shadow model

We use the following criteria to identify a triangle f as Cs.

Geometric criterion: Cast shadow regions should neighbor the floor plane. That is, one of the neighboring triangles should be categorized as part of the floor. In addition to this, removal of cast shadow regions on the V_{Cs} surface should not affect object regions of silhouettes.

Photometric criterion: Cast shadow regions should have similar chromaticity and darker intensity in comparison to that of the background image on *visible* cameras.

Here, *visible* cameras are cameras that can observe the triangle in question. For example, visible cameras of f in Figure 8 are CAM_1 and CAM_2 since CAM_3 cannot observe f because of self-occlusion. Note that the above photometric criterion assumes a white (non-colored) lighting environment as used in [14].

In this algorithm, we assume that we have knowledge of where the object casts its shadows by the camera calibration processed beforehand. We use the floor of the studio, $z = 0$ plane, for simplicity as described above. The geometric criterion states that (1) each triangle in F_{Cs} should neighbor another triangle in F_{Cs} or F_{floor}, and (2) silhouettes produced by the algorithm described above should satisfy intersection consistency. Figure 9(a) and (b) illustrate intersection consistent and non-consistent cases. In Figure 9(a), if we remove the region of F_{Cs} on each camera, the silhouettes on visible cameras CAM_1 and CAM_2 will be carved. This carving on the silhouettes will carve the visual hull V_{cs} as well, but all the silhouettes satisfy the intersection constraint since the carving of

V_{Cs} cannot be observed from cameras CAM_3 and CAM_4 which cannot observe F_{Cs}. On the other hand, if we remove the region of F_{Cs} in Figure 9(b), the change in V_{Cs} caused by the carving of silhouettes in CAM_1 and CAM_2 will be observed from cameras CAM_3 and CAM_4 which cannot observe F_{Cs}. That is, the projection of carved V_{Cs} is not consistent with the silhouettes in CAM_3 and CAM_4, and silhouettes cannot satisfy the intersection constraint. We can detect this situation by using the following simple algorithm.

(1) Let f be a triangle of interest.

(2) For each camera C_i which can observe f, project all triangles in V_{Cs} onto it. Let p denote the pixel on which f is projected. In general, at least two triangles should be projected on p.

(3) If only f and triangles in F_{floor} are projected on p, that is, the visual cone from the projection center of C_i to f crosses only the floor plane, the removal of f does not affect the intersection constraint as shown in Figure 9(a). Otherwise, the removal will affect as shown in Figure 9(b).

We denote this 3D geometry based checking by GeometricCheck(f, C) where C denotes the set of all cameras $C_1, \ldots, C_N$. It returns true if the removal of f does not affect the intersection constraint, and otherwise it returns false.

The photometric criterion utilizes a pixel-wise shadow classifier which is the same as those of [14, 15]. For each camera that can observe a triangle f in it, we check if the pixel corresponding to f is classified as cast shadow or not. We refer to this photometric criterion by PhotometricCheck(v, C) defined as follows:

(1) Let f be the triangle of interest, and the number of cameras n be 0 on which f satisfies the photometric criterion defined above.

(2) For each camera C_i that can observe f, project f onto it. Let us denote the projected area by P and a pixel in P by p. We refer to the R, G, B values at p of the captured and background images on c by $\mathrm{Img}_i(p)$ and $\mathrm{Bg}_i(p)$ where $i = R, G, B$.

- If all $p \in P$ satisfies the following (2) or (3), let $n := n + 1$.

$$0 \leq \textstyle\sum_i (\mathrm{Bg}_i(p) - \mathrm{Img}_i(p)) \leq T_1 \tag{2}$$

$$\begin{cases} T_1 \leq \sum_i (\mathrm{Bg}_i(p) - \mathrm{Img}_i(p)) \leq T_2 \\ \dfrac{\sum_i \mathrm{Img}_i(p)\mathrm{Bg}_i(p)}{\sqrt{\sum_i \mathrm{Img}_i(p)^2 \sum_i \mathrm{Bg}_i(p)^2}} \geq T_3 \end{cases} \tag{3}$$

where T_1, T_2, T_3 are certain thresholds. Equation (3) represents the photometric criterion defined above. We also use Equation (2) to let p be shadow region if it is darker up to T_1. This is because chromaticity will be more sensitive than intensity for low values.

(3) If f satisfies the photometric criterion on more than one camera, that is, $n > 1$, we categorize f as a part of cast shadow region and let PhotometricCheck(f, C) return true. Otherwise, let PhotometricCheck(v, C) return false.

3.2.3 Algorithm

Using the criteria described above, we define an algorithm which removes shadow regions from silhouettes given by the algorithm in Section 3.1. The key point of this algorithm is the order of checking triangles.We traverse the mesh surface from the triangle which is a part of the floor, *i.e.*, $f \in F_{floor}$. Since we assumed that all cast shadows should be neighboring the floor regions F_{floor}, we can avoid checking the triangles which are a part of the self shadow or darkly textured regions not located around the floor.

Step 1 Compute V_{Cs} as polygonal surface model. We used the discrete marching cubes method [16].

Step 2 Let F_{floor} be the set of triangles such that each of them is on the plane $z = 0$. Using F_{floor}, we define F as the set of triangles such that each of them is not in F_{floor} and at least one neighboring triangle is in F_{floor} (Figure 10). We also initialize F_{Obj} and F_{Cs} as empty set.

Step 3 For a triangle $f \in F$,

Step 3.1 If f is classified as cast shadow, *i.e.* both GeometricCheck(f, C) and PhotometricCheck(f, C) return true, let $F := F + U - f$ and $F_{Cs} := F_{Cs} + f$ where U denotes set of triangles such that each of them is neighboring f and does not belong to F_{Obj}, F_{Cs} nor F_{floor}.

Step 3.2 Otherwise, let $F_{\mathsf{Obj}} := F_{\mathsf{Obj}} + f$ and $F := F - f$.

Step 4 If $F \neq \emptyset$, go back to **Step 3**. Otherwise, project V_{Cs} to each viewpoint and obtain final silhouettes by the algorithm described in Section 3.2.1.

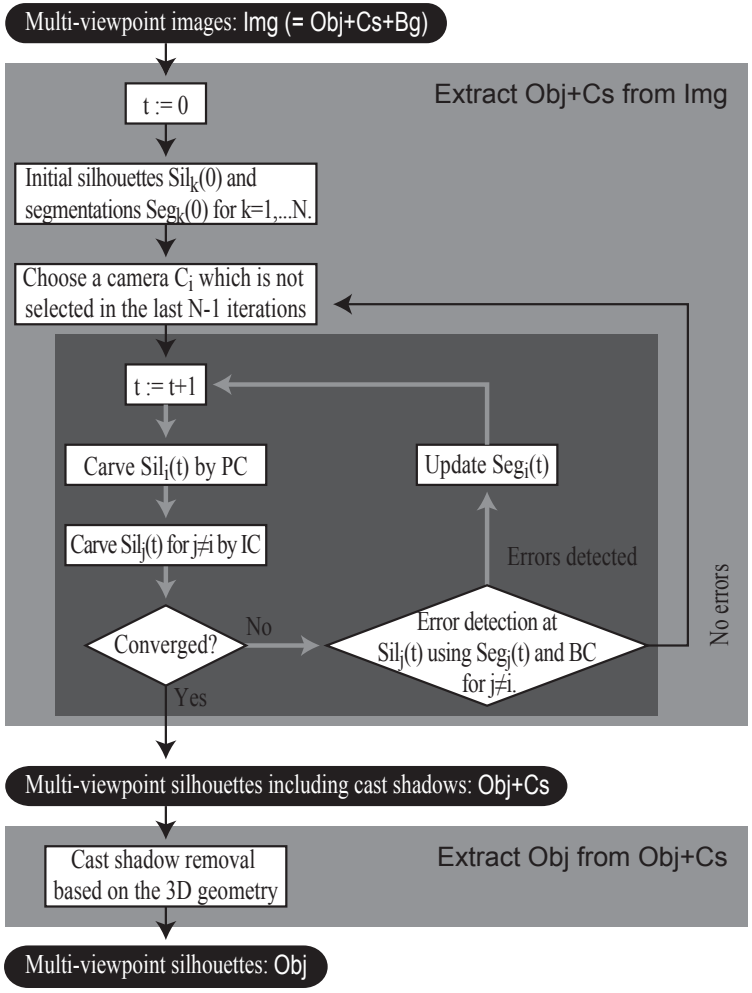


Fig. 2 Overview of the algorithm. The proposed algorithm consists of two parts. The first part extracts multi-viewpoint silhouettes including cast shadows from multi-viewpoint images. The second part removes cast shadows from the result of the first part. The first part has a nested loop structure and the darker area with arrows in light grey color indicates the inner loop.

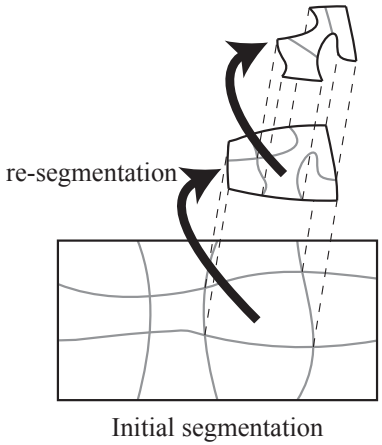


Fig. 3 Coarse-to-fine segmentation. The outline of the segment is preserved through the re-segmentation process.

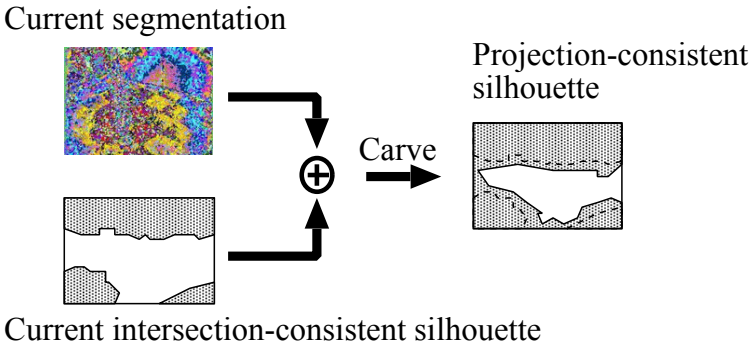


Fig. 4 Silhouette carving. The current intersection-consistent silhouette is carved so as to be the maximum set of segments each of which is included in the current silhouette completely.

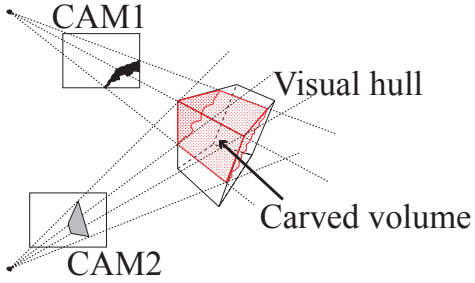


Fig. 5 Intersection consistency. Carving of the black region in the CAM_1 image introduces the removal of the dotted volume from the visual hull which can be seen as the removal of the grey region at CAM_2. Hence if we carve the black region in the CAM_1 image, we have to carve the grey region in the CAM_2 image to keep the intersection consistency.

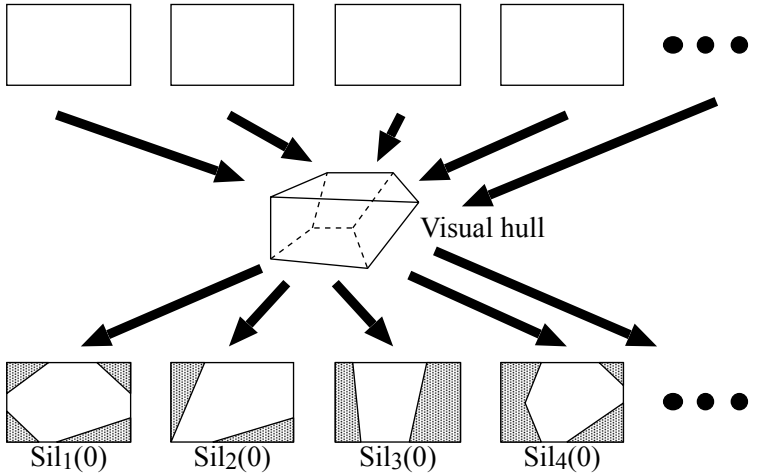


Fig. 6 Initial silhouettes. The initial silhouettes are given as the projection of the visual hull which is computed by assuming that all the silhouettes are equal to the entire region of the image.

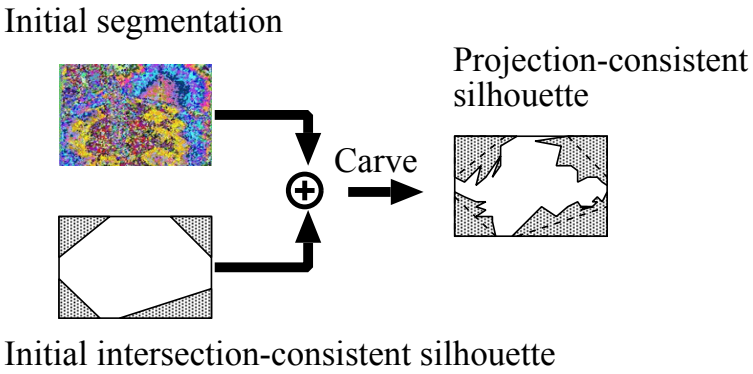


Fig. 7 First carving process at $t = 1$.

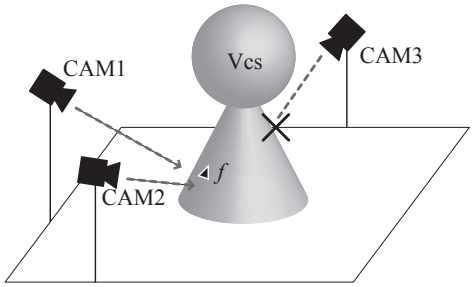
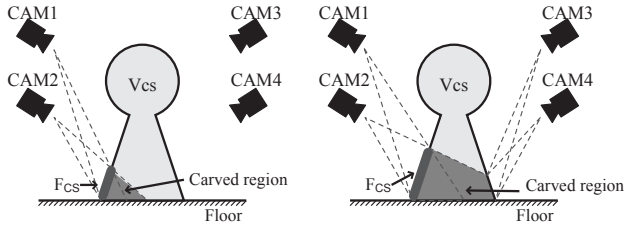


Fig. 8 Visibility checking. While CAM_1 and CAM_2 can observe the triangle f on the object 3D surface, CAM_3 cannot observe f because it is self-occluded.



(a) Intersection consistent (b) Not intersection consistent

Fig. 9 Intersection consistent removal of cast shadow region. Carving of the projection of F_{Cs} from the silhouettes of the cameras each of which can observe F_{Cs} (in this figure, CAM_1 and CAM_2) is equivalent to the removal of the darker volume from V_{Cs}. (a) If this removal is not observable from the other cameras (CAM_3 and CAM_4), it does not affect the intersection consistency. (b) On the other hand, if the removal of the darker volume is observable, it makes the silhouettes of CAM_3 and CAM_4 be intersection inconsistent.

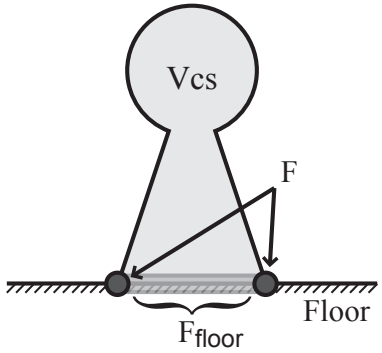


Fig. 10 Initial state of shadow suppression algorithm. V_{Cs} denotes the visual hull computed from the multi-viewpoint silhouettes including cast shadows. F_{floor} denotes the set of triangles on the V_{Cs} surface and on the floor. F is the set of triangles on the V_{Cs} each of which neighbors F_{floor}.

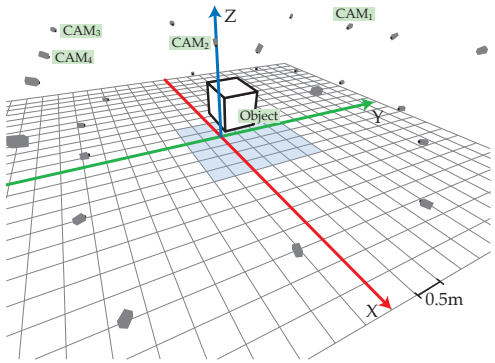


Fig. 11 Camera arrangement

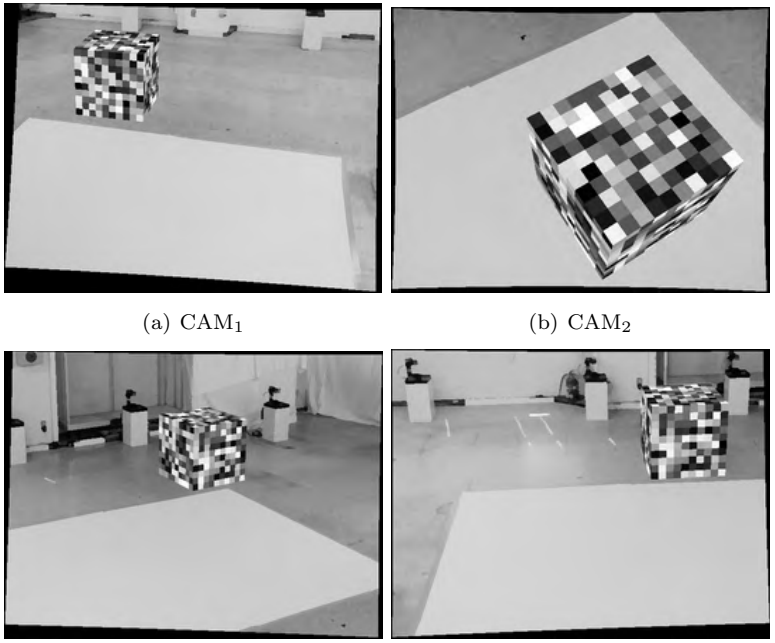
4. Experiments

To demonstrate the capability of our error detection and correction algorithm, we first apply the proposed algorithm to a synthesized dataset. Then, we quantitatively evaluate our method using real images against conventional approaches. Note here that the synthesized dataset does not include any artificial cast shadows to focus on the error detection and correction process.

4.1 Synthesized Images

Figure 12 shows projections of a synthesized object "cube" on 4 of 25 cameras arranged as shown in Figure 11. Figures 13 and 14 show the initial silhouette $\mathrm{Sil}_i(0)$ generated as illustrated by Figure 6, and the initial segmentation $\mathrm{Seg}_i(0)$ respectively. Note that mis-segmentations which cover both the object and background can be observed in Figure 14.

Figures 15 and 16 show the first carving process with $C_i = \mathrm{CAM}_1$ from $t = 0$ to $t = 7$. The white regions in Figure 15 indicate $\mathrm{Sil}_1(t)$. The blue regions in Figure 15 indicate carved areas at each iteration. For example, the blue region of Figure 15(a) indicates the area carved from the previous, *i.e.* the initial silhouette $\mathrm{Sil}_1(0)$ shown in Figure 13(a). We can observe that (1) the silhouette is carved too much according to the rough segmentation as shown in Figures 15(a) and 16(a), (2) then it is recovered as shown in Figures 15(b), (c) and (d). Through this error recovery, the segmentation is refined as shown in Figures 16 (b), (c) and (d) so as to satisfy the condition IsAcceptable(·) on the other viewpoints. In this experiment the algorithm iterated the inner loop of Figure 2 from $t = 1$ to $t = 7$ and then quitted the inner loop to choose the next camera. Figures 17 to 22 show silhouettes and segmentations through carving and error recovery processes from $t = 8$ to $t = 15$ with $i = 2$, from $t = 16$ to $t = 27$ with $i = 3$, and from $t = 28$ to $t = 39$ with $i = 4$.

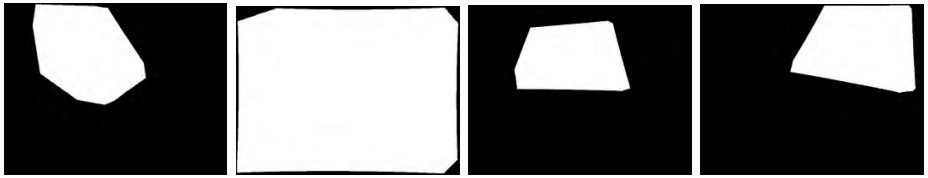


(a) CAM_1 (b) CAM_2

(c) CAM_3 (d) CAM_4

Fig. 12 Input synthesized foreground images

Figure 23 shows the final silhouettes estimated by the proposed algorithm. The algorithm iterated the inner loop of Figure 2 578 times in total while iterating the outer loop 92 times. The processing cost is approximately three days by an Intel Pentium4 running at 3GHz. As a comparison, Figure 24 shows silhouettes given by a pixel-wise simple background subtraction in which each pixel p is



(a) CAM_1 (b) CAM_2 (c) CAM_3 (d) CAM_4

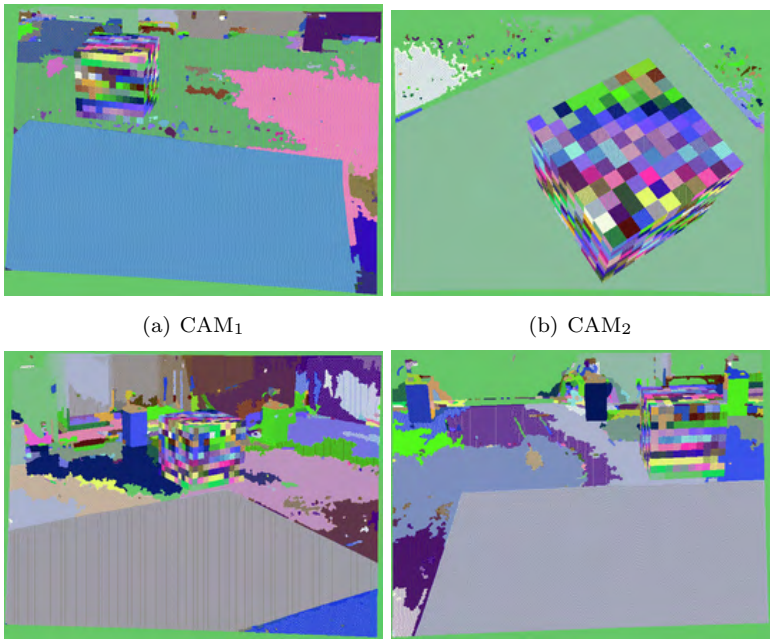
Fig. 13 Initial silhouettes $Sil_k(0), k = 1, \ldots, 4, t = 0$

categorized as silhouette if it satisfies

$$|\mathrm{Img}(p) - \mathrm{Bg}(p)| > T, \tag{4}$$

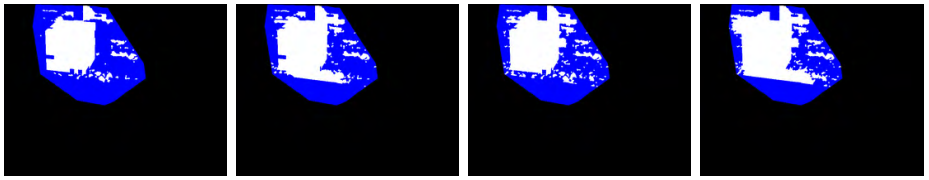
where $\mathrm{Img}(p)$ denotes the intensity of captured image at pixel p, $\mathrm{Bg}(p)$ that of the background image, T a certain threshold. For Figure 24 we used $T = 0$. Therefore the false-negative areas (pixels which are misclassified as background) in Figure 24 indicate that the apparent color of the object is exactly equal to that of the background (the bottom of the cube in Figure 24(b) for example). Compared with 24, our inter-viewpoint error correction algorithm can estimate the silhouette in such regions correctly even though they have no difference in pixel intensities.

We observed undetectable false-negative areas on another viewpoint as shown in Figure 25. As discussed in Section 5, this is because (1) these areas have no or small difference in pixel intensities between the background and (2) the carving of them does not affect the silhouettes on the other viewpoints due to the camera arrangement.



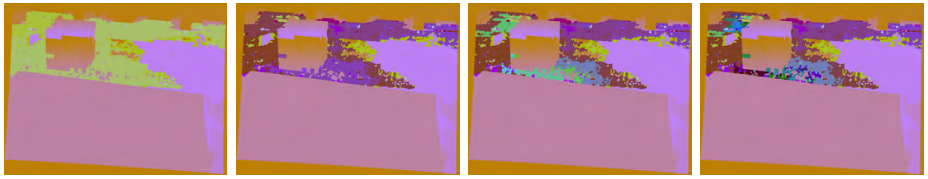
(a) CAM_1 (b) CAM_2 (c) CAM_3 (d) CAM_4

Fig. 14 Initial segmentations $Seg_k(0), k = 1, \ldots, 4, t = 0$



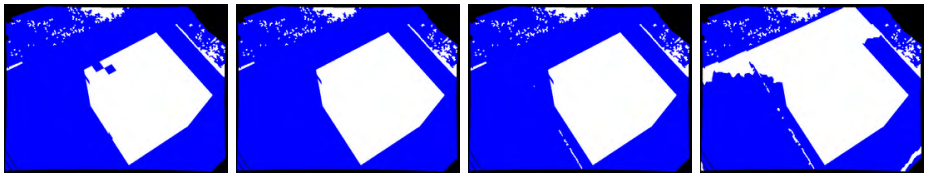
(a) $Sil_1(1)$ (b) $Sil_1(3)$ (c) $Sil_1(5)$ (d) $Sil_1(7)$

Fig. 15 Silhouette carving with error detection and correction of $CAM_i, i = 1, t = 1, \ldots, 7$



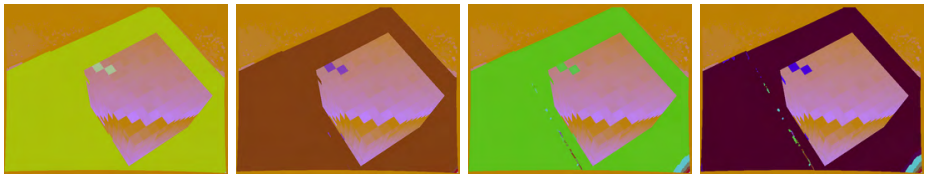
(a) $\mathrm{Seg}_1(1)$ (b) $\mathrm{Seg}_1(3)$ (c) $\mathrm{Seg}_1(5)$ (d) $\mathrm{Seg}_1(7)$

Fig. 16 Re-segmentation process of $\mathrm{CAM}_i, i = 1, t = 1, \ldots, 7$



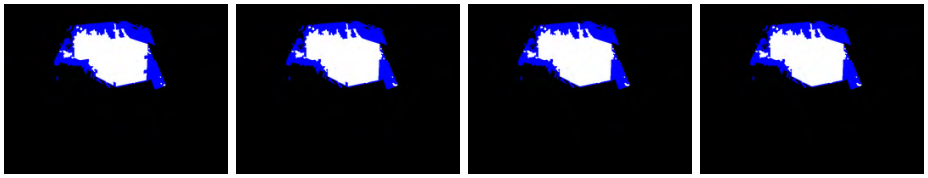
(a) $\mathrm{Sil}_2(8)$ (b) $\mathrm{Sil}_2(11)$ (c) $\mathrm{Sil}_2(13)$ (d) $\mathrm{Sil}_2(15)$

Fig. 17 Silhouette carving with error detection and correction of $\mathrm{CAM}_i, i = 2, t = 8, \ldots, 15$



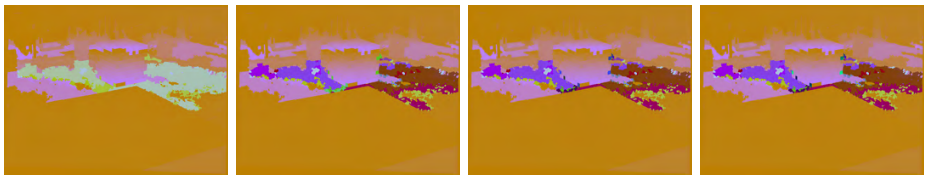
(a) $\mathrm{Seg}_2(8)$ (b) $\mathrm{Seg}_2(11)$ (c) $\mathrm{Seg}_2(13)$ (d) $\mathrm{Seg}_2(15)$

Fig. 18 Re-segmentation process of $\mathrm{CAM}_i, i = 2, t = 8, \ldots, 15$



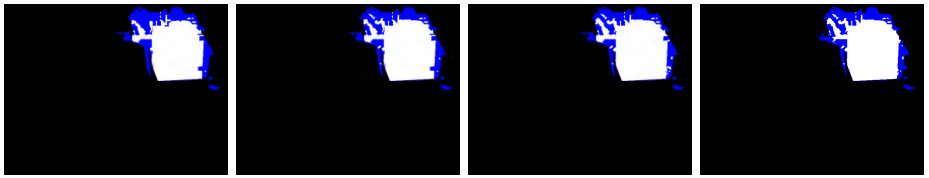
(a) $\mathrm{Sil}_3(16)$ (b) $\mathrm{Sil}_3(20)$ (c) $\mathrm{Sil}_3(22)$ (d) $\mathrm{Sil}_3(27)$

Fig. 19 Silhouette carving with error detection and correction of $\mathrm{CAM}_i, i = 3, t = 16, \ldots, 27$



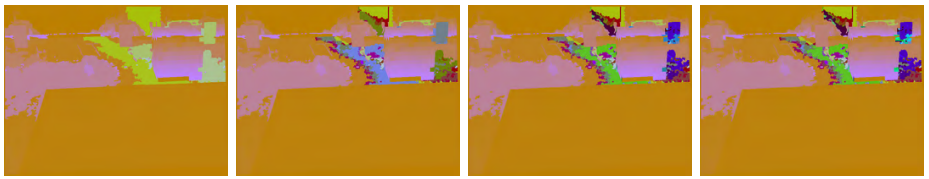
(a) $\mathrm{Seg}_3(16)$ (b) $\mathrm{Seg}_3(20)$ (c) $\mathrm{Seg}_3(22)$ (d) $\mathrm{Seg}_3(27)$

Fig. 20 Re-segmentation process of $\mathrm{CAM}_i, i = 3, t = 16, \ldots, 27$



(a) $\mathrm{Sil}_4(28)$ (b) $\mathrm{Sil}_4(31)$ (c) $\mathrm{Sil}_4(35)$ (d) $\mathrm{Sil}_4(39)$

Fig. 21 Silhouette carving with error detection and correction of $\mathrm{CAM}_i, i = 4, t = 28 \ldots, 39$



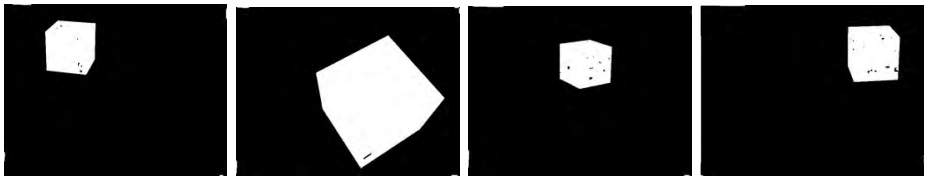
(a) $\mathrm{Seg}_4(28)$ (b) $\mathrm{Seg}_4(31)$ (c) $\mathrm{Seg}_4(35)$ (d) $\mathrm{Seg}_4(39)$

Fig. 22 Re-segmentation process of $\mathrm{CAM}_i, i = 4, t = 28 \ldots, 39$

(a) CAM_1 (b) CAM_2

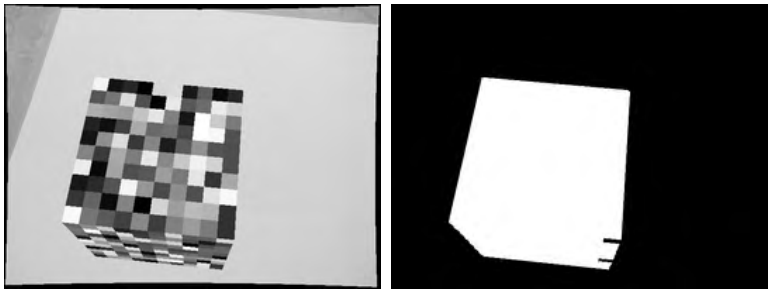
(c) CAM_3 (d) CAM_4

Fig. 23 Estimated silhouettes



(a) CAM_1 (b) CAM_2 (c) CAM_3 (d) CAM_4

Fig. 24 Silhouettes by simple background subtraction with $T = 0$



(a) Input image (b) Estimated silhouette

Fig. 25 Undetectable error (CAM_5)

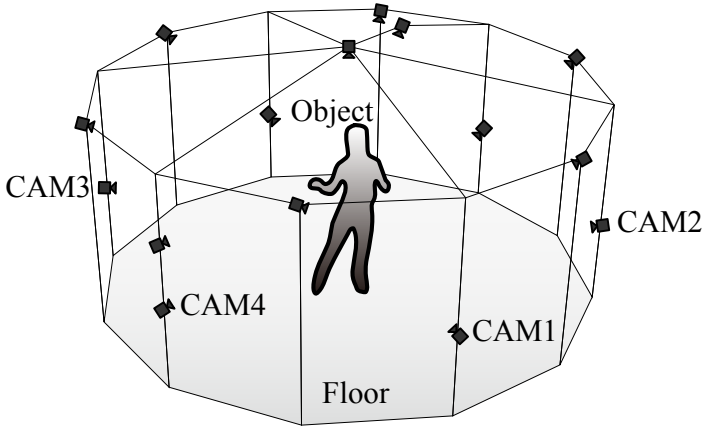


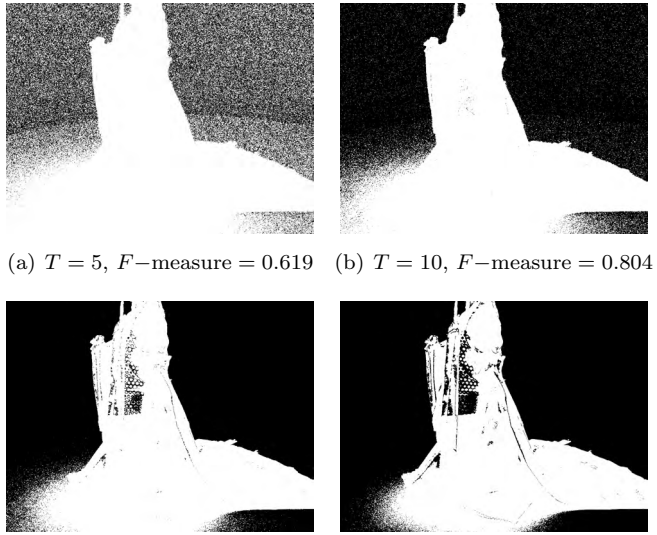
Fig. 26 Camera arrangement

4.2 Real Images

Figure 26 illustrates our camera arrangement. We use 13 XGA cameras on the wall and 2 XGA cameras on the ceiling. All cameras are calibrated beforehand. The top and second rows of Figure 28 show 4 of 15 input and background images captured by the cameras. We can observe that the object region contains some darkly textured regions, *e.g.*, long black hair.

Figure 27 shows the results of the naive background subtraction given by Eq (4) with F-measures defined below. It is clear that there is no magic threshold which produces accurate silhouettes without cast shadows. On the other hand, the third row of Figure 28 shows the results given by the pixel-wise shadow suppression algorithm by Horprasert *et al.* [14]. White and gray regions in this figure denote the object and shadow regions respectively. The results are much better than those of the naive approach, but some darkly textured regions, *e.g.* long black hair regions in CAM_2, are misclassified as shadow. It is hard to avoid this kind of misclassification. This is because each pixel in both cast shadow and darkly textured regions has no difference in pixel level.

The fourth row of Figure 28 shows the result of the algorithm described in Section 3.1. Compared with ground truth silhouettes given by hand (the bottom row of Figure 28), it is clear that the silhouettes include cast shadow regions



(a) $T = 5$, $F-$measure $= 0.619$ (b) $T = 10$, $F-$measure $= 0.804$

(c) $T = 20$, $F-$measure $= 0.869$ (d) $T = 30$, $F-$measure $= 0.869$

Fig. 27 Naive background subtraction (CAM_1)

around the object.

The fifth row of Figure 28 shows the final result of our method. The algorithm iterated the inner loop of Figure 2 398 times in total while iterating the outer loop 54 times to obtain this result. The processing cost is approximately three days by an Intel Pentium4 running at 3GHz with $T_1 = 3$, $T_2 = 200$, $T_3 = 0.99$. Compared with the third row of Figure 28, we can observe that our algorithm does not misclassify the regions which are misclassified as shadow regions by the pixel-wise method.

To obtain these silhouettes, we first extract silhouettes including cast shadow regions by the algorithm described in Section 3.1. Figure 29 shows the initial segmentation of captured images. We can observe that they include some segmentation errors, *i.e.* some segments include both object and background regions. For example, in CAM_4, the segment around the left shoulder of the woman includes the background regions as well. This mis-segmentation carves the object

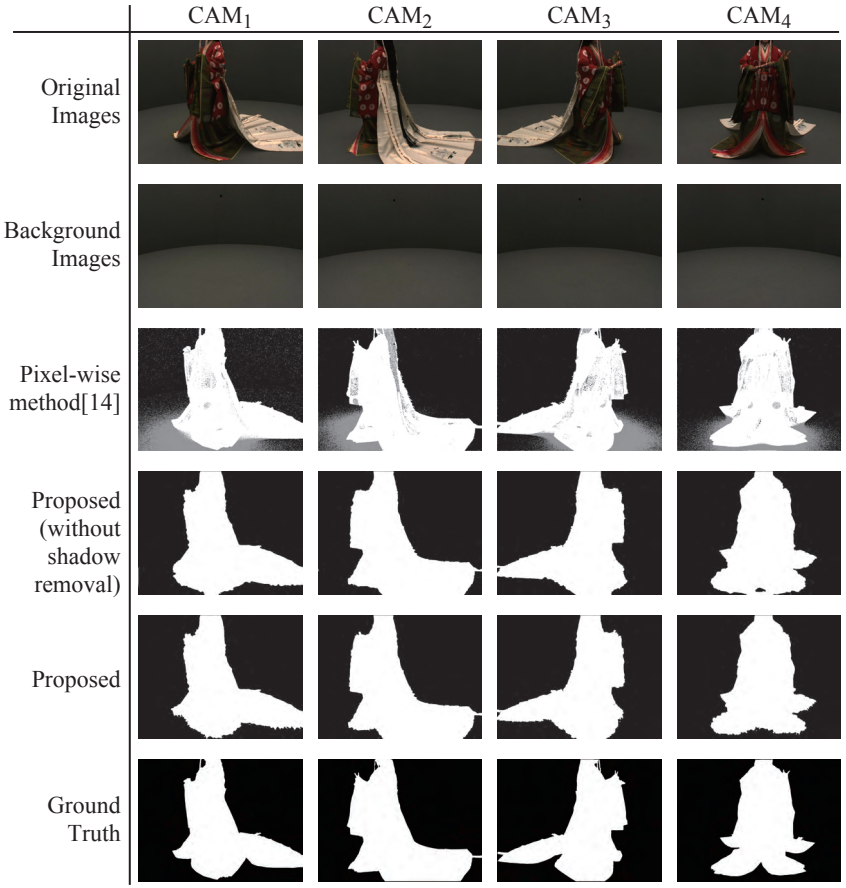
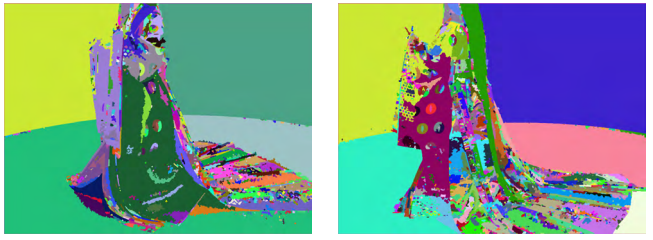
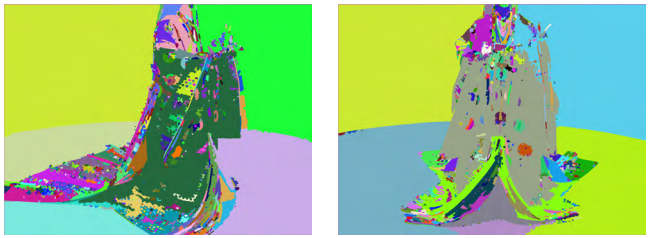


Fig. 28 Silhouettes estimation with a real multi-viewpoint image set.

region as shown in Figure 30(a). Gray areas in this figure denote regions which are carved in the previous iteration, but recovered since the carving of them violates the intersection consistency in other viewpoints. Figures 30 (b), (c), and (d) show how the recovery of mis-carved regions is performed. Figure 31 shows the re-segmentation process in this error correction. We can observe that the region which corresponds to the left shoulder of the woman is split into smaller



(a) CAM_1 (b) CAM_2

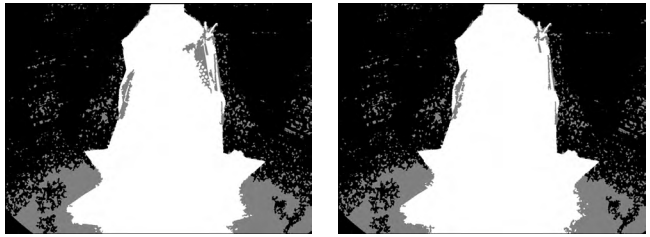


(c) CAM_3 (d) CAM_4

Fig. 29 Initial segmentations $\mathrm{Seg}_k(0), k = 1, \ldots, 4$

segments. This error detection and correction example indicates that the multi-viewpoint silhouette extraction algorithm by Zeng and Quan [12] will fail because it is straightforward and cannot correct mis-segmentations as shown in Figure 30(a).

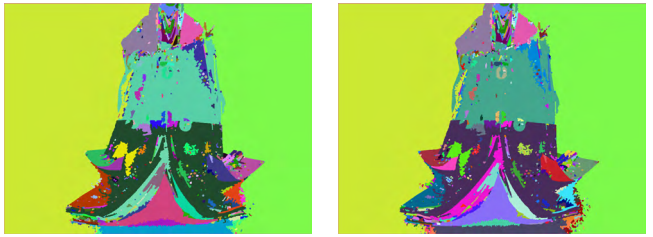
Figure 32 shows how our algorithm detects cast shadow regions. Here, gray areas denote regions classified as cast shadow. We can observe that our algorithm starts cast shadow detection from the triangles on the floor, and proceeds to their neighbors. Finally, we generate silhouettes by projecting the surface of the visual hull which is categorized as object region as described in Section 3.2. Figure 33 shows the final result composed with the input images. We can observe that our algorithm can suppress cast shadows as well as the pixel-wise method shown in the third row of Figure 28, and can preserve darkly textured areas, *e.g.* long black hair, as object region.



(a) $\mathrm{Sil}_4(57)$ (b) $\mathrm{Sil}_4(58)$

(c) $\mathrm{Sil}_4(61)$ (d) $\mathrm{Sil}_4(68)$

Fig. 30 Error correction at $\mathrm{CAM}_4, t = 57, \ldots, 68$



(a) $\mathrm{Seg}_4(57)$ (b) $\mathrm{Seg}_4(58)$

(c) $\mathrm{Seg}_4(61)$ (d) $\mathrm{Seg}_4(68)$

Fig. 31 Re-segmentation at CAM_4

Quantitative evaluation

To evaluate our method quantitatively, we categorize the pixels in a silhouette into the following four types:

True-positive: $S(p) = 1$ and $\hat{S}(p) = 1$,
True-negative: $S(p) = 0$ and $\hat{S}(p) = 0$,
False-positive: $S(p) = 1$ and $\hat{S}(p) = 0$,
False-negative: $S(p) = 0$ and $\hat{S}(p) = 1$,

where $S(p) = 1$ and $S(p) = 0$ denote that a pixel p is estimated as object and background respectively, and $\hat{S}(p) = 1$ and $\hat{S}(p) = 0$ denote that p is a part of object and background respectively in the ground truth image. Here, we use silhouettes given by hand as the ground truth (the bottom row of Figure 28). Using this classification, we evaluate our method by the F-measure defined as follows:

$$\text{Recall} = \frac{N_{\mathrm{TP}}}{N_{\mathrm{TP}}+N_{\mathrm{FN}}}, \tag{5}$$

$$\text{Precision} = \frac{N_{\mathrm{TP}}}{N_{\mathrm{TP}}+N_{\mathrm{FP}}}, \tag{6}$$

$$\text{F-measure} = \frac{2\times\text{Recall}\times\text{Precision}}{\text{Recall}+\text{Precision}}, \tag{7}$$

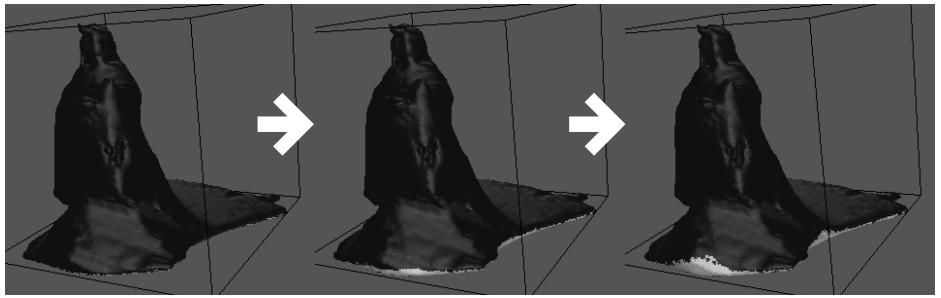
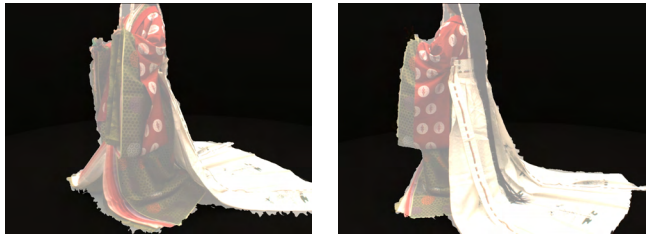


Fig. 32 Cast shadow detection



(a) CAM_1 (b) CAM_2

(c) CAM_3 (d) CAM_4

Fig. 33 Results of the proposed algorithm composed with the input images

	F-measure	Recall	Precision
Pixel-wise method by [14]	0.950	0.953	0.946
Our method without cast shadow removal	0.969	0.989	0.949
Our method	0.975	0.988	0.962

Table 1 Quantitative comparison

where N_{TP}, N_{FP}, and N_{FN} denote the number of true-positive, false-positive, and false-negative pixels in the estimated silhouette image respectively.

Table 1 shows F-measure, recall and precision of Horprasert's pixel-wise method [14] (the third row of Figure 28), our method without cast shadow removal (the fourth row of Figure 28) and our method (the fifth row of Figure 28) respectively. These values indicate that our algorithm outperforms not only the naive background subtraction algorithm but also the pixel-wise method.

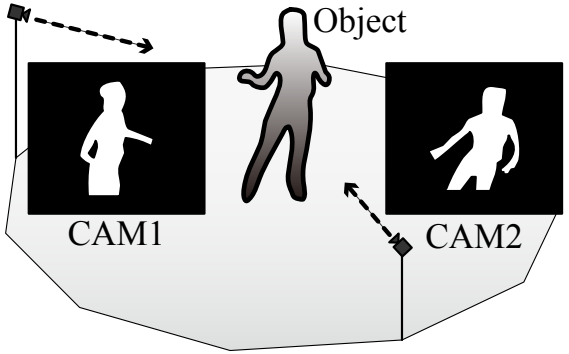


Fig. 34 Apparent hole on 2D silhouette

5. Discussions

5.1 Termination of the algorithm

Using the proposed algorithm in Section 3.1, carved regions can be recovered due to the error correction scheme as shown in Figure 30. However, it cannot go into an infinite loop since we re-segment the recovered regions so that the region is split into smaller segments up to pixel level (Figure 31) while preserving the outline of the original segment as described in Section 3.1.1. This one-way re-segmentation process makes the iteration converge if we omit errors at the single pixel level. The algorithm in Section 3.2 obviously terminates before it carves all the triangles since GeometricCheck$(\cdot)$ stops the carving which violates the intersection consistency IC.

5.2 Limitation on error detection

It is clear that at least one camera should observe the carved region to detect errors in the algorithm in Section 3.1. If carving of the silhouette on j-th camera at **Step** t**.2** does not affect other silhouettes at **Step** t**.4**, any carving is accepted. In this situation, our algorithm just carves the silhouette based on the current segmentation since we cannot use any information from other viewpoints.

5.3 Partial view case

In case that we cannot capture the whole of the object as shown in the top row of Figure 28, we cannot use a naive implementation of SFS method as follows –

"If we have N cameras, all the portions of the visual hull should be projected onto silhouette region at all of N cameras." We can use the following definition instead – "All the portions of the visual hull should be projected onto silhouette region for all of the *observable* cameras." Here, we define a camera to be *observable* if a portion of the visual hull is projected inside its imaging window.

5.4 Topology of silhouette region

Our algorithm carves silhouettes from outside, and it is clear that Carve(·) operation cannot carve holes inside the silhouette. That is, if the object has a real 3D hole, our algorithm cannot estimate it since the hole does not appear as a part of the outside contour of the 2D silhouette and the outside contours satisfy IC. However, we have a chance to carve if it is an apparent 2D hole and not observed as a hole by other cameras. In Figure 34, the 2D hole under the arm on CAM_1 is not a real 3D hole. So if the corresponding region is carved on CAM_2 by Carve(·) at iteration **Step** t**.2**, the hole can be carved at iteration **Step** t**.5**.

5.5 Mesh resolution in the cast shadow detection

The size of the triangles of the mesh surface in Section 3.2 plays an important role to control the accuracy of the cast shadow removal because larger triangles are more likely to violate GeometricCheck(·) while smaller triangles make the convergence slower. In the experiments, we used 1cm size triangles which are fine enough in practice due to the accuracy of the calibration while a coarse-to-fine strategy may speed up the algorithm.

6. Conclusion

In this paper, we proposed a novel error detection, correction and shadow suppression algorithm for multi-viewpoint silhouette extraction. The experiments demonstrated that (1) our algorithm outperforms both the naive background subtraction and monocular pixel-wise shadow suppression algorithms, and (2) our algorithm can correct errors which a conventional multi-viewpoint algorithm cannot recover.

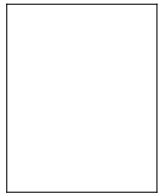
However the monocular segmentation algorithm we used is just a possible implementation which satisfies the one-way characteristic described in Section 3.1.1 and Section 5.1, and is not proven to be the only one nor the best. Further studies are required to evaluate how the choice of the monocular segmentation can affect the proposed algorithm, especially in terms of the speed of convergence and the quality of the result. In addition to this point, our algorithm does not employ temporal-consistency of silhouettes. So our future work will also concentrate on (1) extending for videos, and (2) integrating the both 2D multi-viewpoint silhouette estimation and accurate 3D shape reconstruction based on proposed constraints and texture matching between viewpoints. Development of an efficient computation scheme including coarse-to-fine and parallel processing on GPU is also left for future work since our algorithm is much slower than pixel-wise algorithms due to its iterative nature.

References

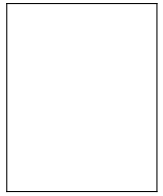
1) Cross, G. and Zisserman, A.: Surface reconstruction from multiple views using apparent contours and surface texture, *Proc.of NATO Advanced Research Workshop on Confluence of Computer Vision and Computer Graphics*, pp.25–47 (2000).
2) Fua, P. and Leclerc, Y.G.: Using 3-Dimensional Meshes To Combine Image-Based and Geometry-Based Constraints, *Proc.of ECCV*, pp.281–291 (1994).
3) Matsuyama, T., Wu, X., Takai, T. and Nobuhara, S.: Real-Time 3D Shape Reconstruction, Dynamic 3D Mesh Deformation and High Fidelity Visualization for 3D Video, *CVIU*, Vol.96, pp.393–434 (2004).
4) Cheung, K.M., Baker, S. and Kanade, T.: Visual Hull Alignment and Refinement Across Time: A 3D Reconstruction Algorithm Combining Shape-From-Silhouette with Stereo, *Proc.of CVPR*, pp.375–382 (2003).
5) Isidoro, J. and Sclaroff, S.: Stochastic Mesh-Based Multiview Reconstruction, *Proc. of 3DPVT*, pp.568–577 (2002).
6) Sinha, S.N. and Pollefeys, M.: Multi-view Reconstruction using Photo-consistency and Exact Silhouette Constraints: A Maximum-Flow Formulation, *Proc.of ICCV*, pp.349–356 (2005).
7) Starck, J., Hilton, A. and Miller, G.: Volumetric stereo with silhouette and feature constraints, *Proc.of BMVC* (2006).
8) Vogiatzis, G., Torr, P. H.S. and Cipolla, R.: Multi-View Stereo via Volumetric Graph-Cuts, *Proc.of CVPR*, pp.391–398 (2005).
9) Laurentini, A.: How far 3d shapes can be understood from 2d silhouettes, *PAMI*, Vol.17, No.2, pp.188–195 (1995).
10) Guillemaut, J.Y., Hilton, A., Starck, J., Kilner, J. and Grau, O.: A Bayesian Framework for Simultaneous Matting and 3D Reconstruction, *Proc.of 3DIM*, pp. 167–176 (2007).
11) Ivanov, Y., Bobick, A. and Liu, J.: Fast Lighting Independent Background Sub-

traction, *IJCV*, Vol.37, No.2, pp.199–207 (2000).
12) Zeng, G. and Quan, L.: Silhouette Extraction from Multiple Images of an Unknown Background, *Proc.of ACCV*, pp.628–633 (2004).
13) Goldlüecke, B. and Magnor, M.: Joint 3D-reconstruction and background separation in multiple views using graph cuts, *Proc.of CVPR*, pp.683–688 (2003).
14) Horprasert, T., Harwood, D. and Davis, L.S.: A Statistical Approach for Real-time Robust Background Subtraction and Shadow Detection, *ICCV Frame-Rate WS* (1999).
15) Han, H., Wang, Z., Liu, J., Li, Z., Li, B. and Han, Z.: Adaptive background modeling with shadow suppression, *Proc.of Intelligent Transportation Systems*, pp. 720–724 (2003).
16) Kenmochi, Y., Kotani, K. and Imiya, A.: Marching Cubes Method with Connectivity, *Proc.of ICIP*, Kobe, Japan, pp.361–365 (1999).

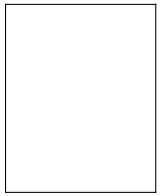
(Received November 28, 2007)
(Accepted July 17, 2007)



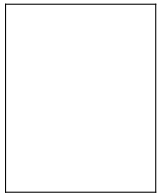
Shohei Nobuhara received his B.Sc. in Engeneering, M.Sc. and Ph.D. in Informatics from Kyoto University, Japan, in 2000, 2002, and 2005 respectively. From 2005 to 2007, he was a post-doctoral researcher at Kyoto University. Since 2007, he has been a research associate at Kyoto University. His research interest includes computer vision and 3D video. He is a member of IPSJ, IEICE, and IEEE.



Yoshiyuki Tsuda received his B.Sc. in Engeneering and M.Sc. in Informatics from Kyoto University, Japan, in 2007 and 2009 respectively. He is with SANYO Electric Co., Ltd. since 2009.



Iku Ohama received his B.Sc. in Engeneering and M.Sc. in Informatics from Kyoto University, Japan, in 2004 and 2006 respectively. He is with Panasonic Corporation since 2006.



Takashi Matsuyama received B. Eng., M. Eng., and D. Eng. degrees in electrical engineering from Kyoto University, Japan, in 1974, 1976, and 1980, respectively. He is currently a professor in the Department of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University.
His research interests include knowledge-based image understanding, computer vision, 3D video, and human-computer interaction. He wrote about 100 papers and books including two research monographs, A Structural Analysis of Complex Aerial Photographs, PLENUM, 1980 and SIGMA: A Knowledge-Based Aerial Image Understanding System, PLENUM, 1990. He won nine best paper awards from Japanese and international academic societies including the Marr Prize at ICCV'95. He is on the editorial board of Pattern Recognition Journal.
He was given Fellowships from International Association for Pattern Recognition, Information Processing Society Japan, and Institute for Electronics, Information, and Communication Engineers Japan.