3D Video Technologies: Capturing High-Fidelity Full 3D Shape, Motion, and Texture

Takeshi Takai* Shohei Nobuhara† Hiromasa Yoshimoto‡ Takashi Matsuyama§

Kyoto University

Figure 1: Visualized 3D Video for MR-PreViz.

Abstract

3D Video is a new media that records a dynamic event in the real world as is, i.e., high-fidelity full 3D shape, motion, and texture of an object. In this paper, we first introduce 3D video and the *MR-PreViz* (previsualization with Mixed-Reality techniques) project for film-making briefly, then present the scheme of 3D video generation and the utilization of 3D video into the MR-PreViz. We finally show the utility of our technologies by examples of the visualized scene in which two *samurais* are fighting a battle with swords in a traditional Japanese downtown, which is composed of 3D video and a virtual CG set.

CR Categories: H.5.1 [Information Interface and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities I.2.10 [Artificial Intelligence]: Vision and Scene Understanding—Motion, Shape, Texture

Keywords: 3D video, volume reconstruction, mesh deformation, texture mapping, film-making, previsualization, mixed-reality.

1 Introduction

1.1 3D Video

We are developing 3D video technologies for these several years [9, 10, 11, 12, 16, 19], which enable us to capture a dynamic event in the real world as is; it records time varying 3D object shape with high fidelity surface properties (i.e. color and texture). Its applications cover wide varieties of personal and social human activities: entertainment (e.g. 3D game and 3D TV), education (e.g. 3D animal picture books), sports (e.g. sport performance analysis), medicine (e.g. 3D surgery monitoring), culture (e.g. 3D archive of traditional dances) and so on.

*e-mail: takesi-t@vision.kuee.kyoto-u.ac.jp
†e-mail: nob@vision.kuee.kyoto-u.ac.jp
‡e-mail: yosimoto@vision.kuee.kyoto-u.ac.jp
§e-mail: tm@vision.kuee.kyoto-u.ac.jp

Several research groups developed real-time 3D shape reconstruction systems for 3D video and have opened up the new world of image media [13] [6] [18] [3] [4]. All these systems focus on capturing human body actions and share a group of distributed video cameras for real-time synchronized multi-viewpoint action observation. While the real-timeness of the earlier systems[13] [6] was confined to the synchronized multi-view video observation alone, the parallel volume intersection on a PC cluster has enabled the real-time 3D shape reconstruction [18] [3] [4].

As an example of the application of 3D video, we present a utilization of 3D video to the MR-PreViz project [1] that aims for supporting the previsualization process of film-making. Though computer graphics and motion capture system are utilized for the previsualization, one of the most advantageous points over ordinary computer graphics and motion capture systems is that the natural motions and appearance of objects can be representable. Figure 2 illustrates the basic scheme of 3D video generation. We present the summary of the major technologies in Section 2.

1.2 The MR-PreViz Project for Film-Making

In a creation of epic films, previsualization is becoming conventional in order to show directors' concepts, thoughts, idea, etc. to their film crews, and to make clear their own images with visualized computer graphics rather than a simple storyboard. The MR-PreViz project is aimed for supporting the previsualization process with mixed-reality (MR) techniques, which enable us to visualize a virtual scene into a real scene with geometrical and photometrical consistency. We show a basic flow of film-making as follows:

1. **Preproduction:**
   - Scenario.
   - Casting.
   - ***Previsualization.***
   - Preparation of filming.
2. **Production:**
   - Camerawork.
   - Lighting.
   - Filming.

Figure 2: Basic scheme of 3D Video Generation.

- Recording.

3. **Postproduction:**
   - Editing.
   - VFX.

In the project, we aspire to accomplish an *on-site* previsualization system, instead of most existing ones with simple computer graphics in a production office. With the MR-PreViz system, the director and the film crew can find out the camerawork and the composition that they desire through a trial and error process under a real set, scene, and lighting with consistently superimposed CG and/or 3D video, feeling the ambient atmosphere of the scene (Figure 3(a)). After the MR-PreViz process, the filming can be done efficiently by the director, the film crew, and the actors with the previsualized images (Figure 3(b)).

(a) MR-PreViz. (b) Filming.

Figure 3: Film-making with MR-PreViz [1].

This project is led by Prof. H. Tamura at Ritsumeikan Univ., and organized with Ritsumeikan Univ., NAIST, and Kyoto Univ. for five years (Oct. 2005 – Sept. 2010). Ritsumeikan Univ. organizes the project and develops an action editor, a 3D-space-layout tool, and a camerawork authoring tool, NAIST develops an MR system for outdoor scenes with geometrical and photometrical consistency, and we, Kyoto Univ., develop a method for capturing, visualizing and editing of 3D video over the coming four years. In this paper, we present visualization of 3D video for the MR-PreViz system in Section 3.

2 3D Video Generation

2.1 Studio Configuration

Figure 4 illustrates the configuration of our studio whose diameter is 6 m and height is 2.5 m, and the size of the area that we can reconstruct an object without defects is approximately 3 m × 3 m × 2 m in the center of the studio. We have a PC cluster system for capturing, which is composed by 15 PCs and one master PC. Each PC has one fixed-camera, and the cameras are connected to an external pulse generator for triggering. The specifications of the PC and the camera are following.

- **PC:** Pentium III 1 GHz × 2, 1 GB RAM
- **Camera:** Sony XCD-X710CR, XGA, 25 fps with an external triggering mode

The cameras are calibrated by the method in the OpenCV Library [2]. With this system, we can obtain a synchronized multi-viewpoint image at 25 fps. In order to click the shutter at high speeds (1/1000 sec) for capturing dynamic actions, we put a lot of fluorescent light tubes at the ceiling of the studio, which emit a natural light whose spectrum is almost equivalent to the sunlight. We also have a larger studio with 25 active-cameras and a PC cluster of 30 PCs, aiming to develop a method to generate 3D video of multiple objects with tracking in a wide area. In this paper, we capture multi-viewpoint images with the studio that has 15 cameras, and we generate 3D video with the PC cluster in the larger studio, because of the computational capability. The specification of the PC in the larger studio is Zeon 3.6 GHz × 2 and 2 GB RAM.

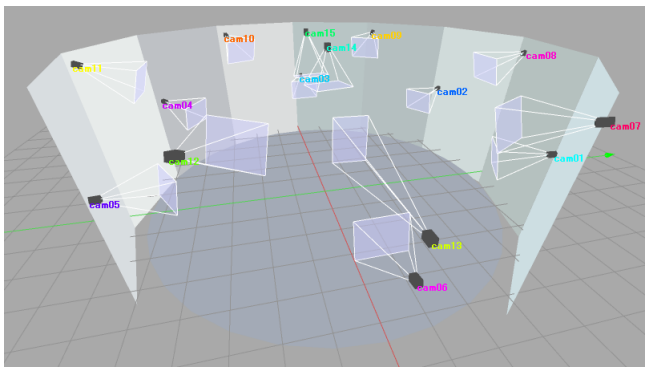


Figure 4: Studio configuration for capturing 3D video.

2.2 Real-time 3D Volume Reconstruction

We reconstruct a 3D volume of an object by the silhouette volume intersection method [8], which is based on the silhouette constraint that a 3D object is encased in the 3D frustum produced by back-projecting a 2D object silhouette on an image plane. With multi-view object images, therefore, an approximation of the 3D object shape can be obtained by intersecting such frusta (Figure 5). We have extended the method to reconstruct a volume from partial views of an object, which enables us to capture an image of

the object in closeup, and accordingly, we can obtain a high quality volume (The captured images are shown in Figures 10 and 11). We have also developed a real-time system for the method, which can reconstruct 30 volumes of objects per second at the resolution of 8 mm^3 with the PC cluster of 30 PCs that is described above. The details are presented in [19].

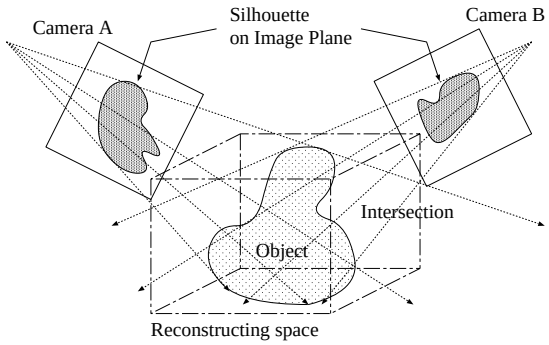


Figure 5: Silhouette volume intersection.

2.3 3D Shape Recovery with Deformable Mesh Model

A mesh that is reconstructed by the volume intersection and the discreet marching cubes [7] methods represents an approximate shape of an object, because the volume intersection method cannot reconstruct a concave portion of an object in principle, and the discreet marching cubes method only generates surfaces with a few discreet normal vectors. For high-accuracy recovery of an object, we have presented a method of a heterogeneous deformation model for 3D mesh and motion recovery [15]. In this paper, we apply the intra-frame deformation to each visual hull individually, since the current inter-frame deformation method cannot cope with global topological structure changes. In the following sections, we present a simple overview of the intra-frame deformation.

The procedure of the intra-frame deformation is follows:

Step 1. Compute the visibility of each vertex.

Step 2. Compute force $\boldsymbol{F}(v)$ that works on each vertex independently.

Step 3. Move each vertex by $\boldsymbol{F}(v)$.

Step 4. Terminate if all vertex motions are small enough. Otherwise go back to Step 1.

Force $\boldsymbol{F}(v)$ in Step 2 denotes a constraint of vertex v, which is defined as

$$\boldsymbol{F}(v) \equiv \alpha \boldsymbol{F}_i(v) + \beta \boldsymbol{F}_e(v) + \gamma \boldsymbol{F}_s(v), \tag{1}$$

where α, β, and γ ace coefficients for each force. The values are specified by heuristics as 0.3, 0.4, 0.3, respectively. $\boldsymbol{F}_i(v)$, $\boldsymbol{F}_e(v)$, and $\boldsymbol{F}_s(v)$ denote *internal*, *photometric*, and *silhouette preserving* forces, respectively. The details of the forces are as follows:

internal force, $\boldsymbol{F}_i(v)$: works to make the shape to keep smooth, and defined by

$$\boldsymbol{F}_i(v) \equiv \frac{\sum_j^n \boldsymbol{q}_{v_j} - \boldsymbol{q}_v}{n}, \tag{2}$$

where $\boldsymbol{q}_{v_j}$ denotes the neighboring vertices of v and n the number of neighbors. $\boldsymbol{F}_i(v)$ performs as tension between vertices and keeps them locally smooth.

Note that the utilities of this internal force is twofold:

1. make the mesh shrink, and
2. make the mesh smooth.

We need 1. in the intra-frame deformation since it starts with the visual hull which encases the real object shape. 2. on the other hand, stands for a popular smoothness heuristic employed in many vision algorithms such as the regularization and active contour models. The smoothing force works to prevent self-intersection since a self-intersecting surface includes protrusions and dents, which will be smoothed out before causing self-intersection.

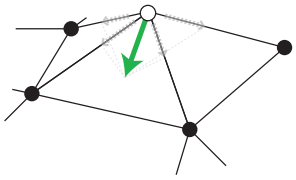


Figure 6: Internal force.

photometric force, $\boldsymbol{F}_e(v)$: works to make the shape to fit the captured image. We define the photometric force $\boldsymbol{F}_e(v)$ by

$$\boldsymbol{F}_e(v) \equiv \begin{cases} \nabla E_e(\boldsymbol{q}_v), & \text{if} \quad N(C_v) \geq 2, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

where C_v denotes a group of cameras that can observe vertex v, $N(C_v)$ the number of cameras in C_v, and $E_e(\boldsymbol{q}_v)$ the correlation of textures to be mapped around v (Figure 7(a)):

$$E_e(\boldsymbol{q}_v) \equiv \frac{1}{N(C_v) - 1} \sum_{c \in C_v \setminus c_m} \mathrm{NCC}\,(c, c_m)\,, \tag{4}$$

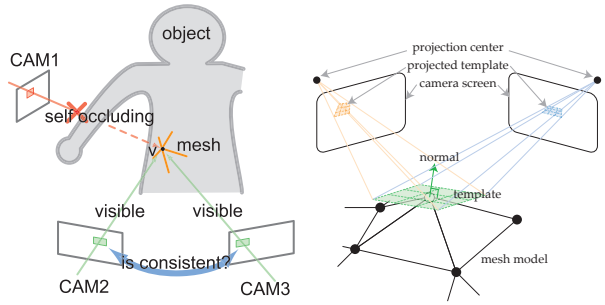
where c_m denotes the most facing camera in C_v, c a camera in C_v except c_m, $\mathrm{NCC}\,(c, c_m)$ the normalized cross correlation function between c and c_m given by:

$$\mathrm{NCC}\,(c, c_m) \equiv \frac{\iint_{w_v} (p_{w_v,c}\,(x,y) - \overline{p_{w_v,c}})\,(p_{w_v,c_m}\,(x,y) - \overline{p_{w_v,c_m}})\,dxdy}{\sqrt{\iint_{w_v} (p_{w_v,c}\,(x,y) - \overline{p_{w_v,c}})^2\,dxdy \iint_{w_v} (p_{w_v,c_m}\,(x,y) - \overline{p_{w_v,c_m}})^2\,dxdy}}, \tag{5}$$

where w_v denotes the template window around v (Figure 7(b)), $p_{w_v,c}$ the texture corresponding to w_v on the image captured by c, and $\overline{p_{w_v,c}}$ the average of the $p_{w_v,c}$. Note that the template window w_v for v is a rectangle plane tangent to v with a certain size, and is projected onto each camera c to obtain the texture $p_{w_v,c}$ (Figure 7(b)). The size of the template window in practice is determined by the distances between neighboring vertices and the resolution of captured images.

silhouette preserving force, $\boldsymbol{F}_s(v)$: works to make the shape to fit the contour of the silhouette. Figure 8(b) explains how this force at v is computed, where $S_{o,c}$ denotes the object silhouette observed by camera c, $S_{m,c}$ the 2D projection of the 3D mesh onto the image plane of camera c, and v' the projection of v onto the image plane of camera c.

1. For each c in C_v, compute the partial silhouette preserving force $\boldsymbol{f}_s(v, c)$ by the following method.
2. If v' is on the boundary of mesh silhouette $S_{m,c}$ between the background, and
   (a) v' is located outside of $S_{o,c}$ or



(a) Photometric constraint. (b) Template matching.

Figure 7: Photometric force.

(b) v' is located inside of $S_{o,c}$,

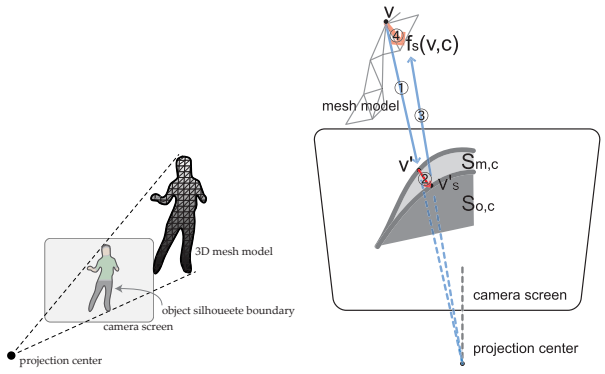
then compute the shortest vector from v' to $S_{o,c}$ (Figure 8(b) ②), i.e., v'_s and assign its corresponding vector to $\boldsymbol{f}_s(v,c)$ (Figure 8(b) ④, see below).

3. Otherwise, $\boldsymbol{f}_s(v,c) = 0$.

Here, we compute $\boldsymbol{f}_s(v,c)$ (Figure 8(b) ④) as follows:

$$\boldsymbol{f}_s(v,c) = (\boldsymbol{n}_v \cdot \boldsymbol{d}_{v,v'})\,\boldsymbol{n}_v, \tag{6}$$

where $\boldsymbol{n}_v$ denote the normal vector at v and $\boldsymbol{d}_{v,v'}$ the vector from v' to v'_s. Note here that $\boldsymbol{d}_{v,v'}$ is a 3D vector represented in the coordinate system of the mesh model.



(a) Silhouette constraint. (b) Silhouette preserving force.

Figure 8: Silhouette force.

For more details of the mesh deformation algorithm including the inter-frame deformation, see [15, 16].

2.4 High Fidelity Texture Mapping

In this section, we propose a novel texture mapping algorithm for the high fidelity visualization of 3D video. The problem we are going to solve here is how we can generate high fidelity object images from arbitrary viewpoints based on the 3D object shape with limited accuracy. Even though the mesh is deformed by the method that is described in Section 2.3, it contains errors according to the calibration errors and insufficiency of the deformation.

The input data for this algorithm are

- **A temporal series of 3D mesh data,**
  It was obtained by applying the intra-frame deformation to each visual hull individually.
- **A temporal series of multi-view video data, and**
- **Camera calibration data for all cameras.**

For high fidelity texture mapping, we first compute vertex color data, which we simply call it as *texture*, of each vertex in each camera. That is, what we call as *3D video* is a temporal sequence of a pair of the mesh and the texture. We then visualize 3D video in real time. In the following sections, we describe our texture mapping method, *the viewpoint dependent vertex-based method*, in detail.

2.4.1 Definitions

First of all, we define words and symbols as follows, where a bold face symbol denotes an array:

- **`vertex color:`** denotes color value of red (R), green (G), blue (B), and α (A) that denotes the visibility of the vertex with 0 (invisible) and 1 (visible).
- **mesh, $\boldsymbol{M}$**: denotes the deformed mesh, which has arrays of vertices and face indices.
- **group of cameras, $\boldsymbol{C}$**: Each camera has the intrinsic and extrinsic parameters and captured images.
- **array of vertex colors for rendering, $\boldsymbol{V}$**: This array is sent to the graphics processing unit (GPU) for rendering. E.g., $\boldsymbol{V}_v$ denotes the vertex color of vertex v.
- **texture, $\boldsymbol{T}$**: denotes the texture is an array of vertex colors from all the cameras. E.g., $\boldsymbol{T}_{v,c}$ denotes the vertex color of vertex v in camera c, and $\boldsymbol{T}_v$ an array of vertex colors of vertex v in $\boldsymbol{C}$.
- **viewpoint, $\vec{P}_{eye}$**: denotes a position from which we observe an object.
- **viewing direction, $\vec{V}_{eye}$**: denotes a unit vector of the direction from the viewpoint to a gazing point.
- **vector, $\vec{V}_c$**: denotes a unit vector of the direction of the optical axis of camera c.
- **vector, $\vec{V}_{eye\to v}$**: denotes a unit vector from the viewpoint to vertex v.
- **normal, $\vec{N}_v$**: denotes a surface normal of vertex v.
- **weighting factors, $\boldsymbol{w}$ and $\bar{\boldsymbol{w}}$**: denote arrays of weighting factors and normalized weighting factors, respectively. E.g., $\boldsymbol{w}_c$ denotes the weighting factor of camera c.
- **sharpness, m**: denotes sharpness of the weighting factors. In this paper, we set $m = 10$ by heuristics.

2.4.2 Texture Generation

For the viewpoint dependent vertex-based method, we first generate the texture. The format is an array of vertex colors, of which size is a product of a number of vertices and a number of cameras.

Algorithm 1 presents generation of the texture. In the algorithm, `CheckVisibiliy` is a function which returns *true* if vertex v is visible from camera c, and *false* in the otherwise. The visibility is judged by orientation of normal of the vertex and whether the vertex is occluded or not using a depth buffer[1]. `PickUpColor` is a function which returns a color value of vertex v in camera c.

[1]The depth buffer is an image holding depth values of vertices from the optical center of a camera, instead of color values.

Algorithm 1: Texture Generation.

input : mesh, $\boldsymbol{M}$, and group of cameras, $\boldsymbol{C}$
output: texture, $\boldsymbol{T}$
foreach *vertex v in* $\boldsymbol{M}$ **do**
  foreach *camera c in* $\boldsymbol{C}$ **do**
    IsVisibile ← `CheckVisibility(v, c)`;
    if IsVisibile **then**
      $\boldsymbol{T}_{v,c} \leftarrow$ `PickUpColor(v, c)`;
      $\boldsymbol{T}^{A}_{v,c} \leftarrow 1$;
    else
      $\boldsymbol{T}^{R}_{v,c} \leftarrow \boldsymbol{T}^{G}_{v,c} \leftarrow \boldsymbol{T}^{B}_{v,c} \leftarrow \boldsymbol{T}^{A}_{v,c} \leftarrow 0$;

2.4.3 Viewpoint Dependent Vertex-based Rendering

We render an image of 3D video, i.e., the mesh and the texture, in Algorithm 2. In the algorithm, `NormalizeWeightingFactors` is a function for normalizing the weighting factors by

$$\bar{\boldsymbol{w}}_c = \frac{\boldsymbol{w'}_c}{\sum_C \boldsymbol{w'}}, \tag{7}$$

where $\boldsymbol{w'}_c = 0$, if vertex v is invisible from camera c, i.e., $\boldsymbol{T}^{A}_{v,c} = 0$, and $\boldsymbol{w'}_c = \boldsymbol{w}_c$ in the otherwise. The term, +2 of $\boldsymbol{w}_c \leftarrow (\vec{V}_{eye} \cdot \vec{V}_c + 2)^m$ in the first **foreach**, is a bias for letting the value in the parentheses be greater than zero.

Algorithm 2: Rendering of 3D video (viewpoint dependent vertex-based method).

input : viewpoint, $\vec{P}_{eye}$, viewing direction, $\vec{V}_{eye}$, mesh, $\boldsymbol{M}$, texture, $\boldsymbol{T}$, and group of cameras, $\boldsymbol{C}$
foreach *camera c in* $\boldsymbol{C}$ **do**
  $\boldsymbol{w}_c \leftarrow (\vec{V}_{eye} \cdot \vec{V}_c + 2)^m$;
foreach *vertex v in* $\boldsymbol{M}$ **do**
  if $\vec{V}_{eye \to v} \cdot \vec{N}_v < 0$ **then**
    $\bar{\boldsymbol{w}} \leftarrow$ `NormalizeWeightingFactors(`$\boldsymbol{w}, \boldsymbol{T}_v$`)`;
    $\boldsymbol{T}^{R}_{v,c} \leftarrow \boldsymbol{T}^{G}_{v,c} \leftarrow \boldsymbol{T}^{B}_{v,c} \leftarrow \boldsymbol{T}^{A}_{v,c} \leftarrow 0$;
    foreach *color channel k in R, G, B* **do**
      foreach *camera c in* $\boldsymbol{C}$ **do**
        $\boldsymbol{V}^{k}_{v} \leftarrow \boldsymbol{V}^{k}_{v} + \bar{\boldsymbol{w}}_c \boldsymbol{T}^{k}_{v,c}$;
`RenderinpWithGPU(`$\boldsymbol{M}, \boldsymbol{V}$`)`;

3 Visualization of 3D Video for The MR-PreViz System

3.1 Reduction of Size of 3D Video Data

For the MR-PreViz system, we need to reduce the size of the 3D video data, since the size is very huge as described in Section 2, e.g., 300 MB/sec in the resolution of 5 mm^3 (roughly 100,000 vertices, 200,000 faces, and 15 cameras for the texture). Although we are developing a compression method for 3D video [5], we employ a mesh at low resolution and simplify the texture because the MR-PreViz system requires real-time rendering rather than quality. The low-resolution mesh has approximately 60,000 polygons (10 mm^3), and the simplified texture is generated as the *view independent texture*, i.e., one color data for each vertex, by[2]

$$\boldsymbol{V}_v = \sum_C \bar{\omega}_{v,c} \cdot \boldsymbol{T}_{v,c}, \tag{8}$$

[2]Eq. (8) is computed for each color channel; red, green, and blue.

where $\boldsymbol{T}_{v,c}$ denotes the color of vertex v in camera c, and $\boldsymbol{V}_v$ denotes the generated color of vertex v. $\bar{\omega}_c$ is a normalized weighting factor for $\boldsymbol{T}_{v,c}$, which is given by

$$\bar{\omega}_{v,c} = \frac{\omega_{v,c}}{\sum_C \omega_v}, \tag{9}$$

where $\omega_{v,c}$ is a dot product of the viewing direction of camera c and the normal of vertex v, i.e., $\omega_{v,c} = -\vec{V}_c \cdot \vec{N}_v$. Note that $\omega_{v,c} = 0$, if vertex v is invisible from camera c, i.e., $\vec{V}_c \cdot \vec{N}_v \geq 0$ or vertex v is self-occluded.

Thus the data format of one frame of 3D video becomes

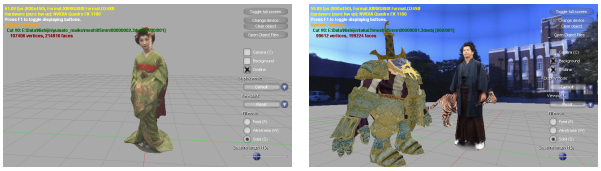
- Array of vertex
  Format: `x, y, z, color`, and
- Array of face index
  Format: `vertex0, vertex1, vertex2`.

By the above process, the size of one frame of 3D video is reduced to be roughly 700 KB and the data rate is 17 MB/sec (6 % against the original data with 5 mm^3 resolution), which is less enough than transaction speed from HDD to a graphics memory.

3.2 Interactive Viewer for 3D Video

We have been developing a viewer for 3D video, which enable us to interactively observe 3D video data from an arbitrary viewpoint. It is developed with DirectX 9.0c, and can run on an ordinary notebook PC (Pentium M 2.26 GHz, 2 GB memory, nVidia GeforceGo 6400) with approximately 200 fps using the simplified 3D video data that is described in the previous section. It has sufficient performance for the MR-PreViz system that requires portable and on-site equipment.

The viewer can render multiple 3D video data with CG models and/or a live-action background (Figure 9), and also have a function of simple editing of object's scale, translation, and rotation. We are going to make the graphics engine of the viewer into a module for the MR-PreViz system.



(a) One object (b) One object and CG characters with an omni-directional image

(c) One object and a live-action background (d) Two objects and a virtual background

Figure 9: Interactive Viewer for 3D Video.

4 Experimental Results

4.1 Data Acquisition

In order to examine our studio for the MR-PreViz project, we captured a swordfight scene in which two samurais wearing kimonos

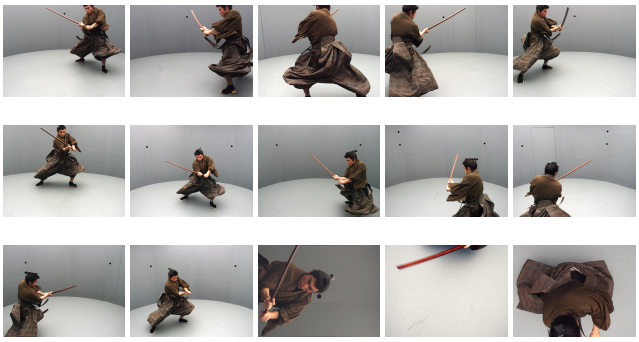


Figure 10: Examples of captured images (samurai 1, #331).

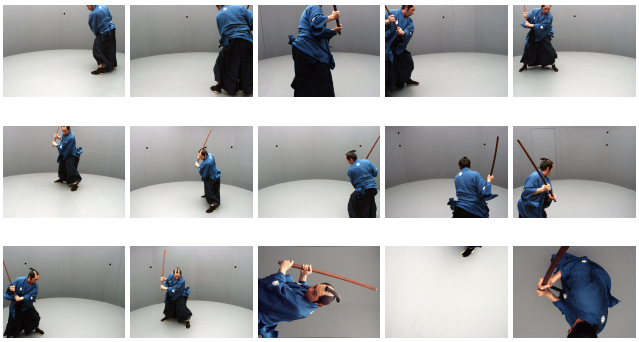


Figure 11: Examples of captured images (samurai 2, #361).

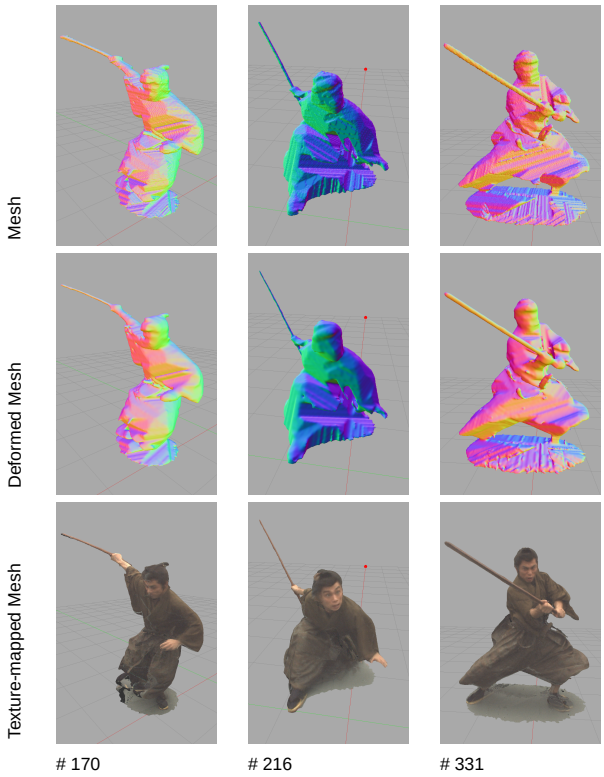


Figure 12: Examples of generated 3D video frames (Samurai 1, 5 mm^3). The color of the Mesh and Deformed Mesh denotes the surface normal by replacing (x, y, z) with (r, g, b).

were fighting a battle with slender wooden swords. We invited two professional actors of samurais and one swordfight arranger, and requested them to act as in the case of the real filming. For the limitation of the space of our studio, we captured one person at a time, and the action was synchronized by their senses with careful rehearsals. The configuration of the studio is described in Sec. 2.1.

The examples of captured images are shown in Figures 10 and 11. The images illustrate that few motion blurs appear around the wooden sword that moves at a high speed of roughly 15 m/sec[3]. This result indicates that our system for the 3D video capturing has a capability to capture the data for the MR-PreViz system.

4.2 3D Video Generation

Figures 12 and 13 show examples of reconstructed 3D video frames. The resolution of the shape is 5 mm^3, and the size is approximately 100,000 polygons and 13 MB with the viewpoint-dependent texture per one frame. The computing time for the 3D video generation is roughly 30 seconds per one frame using the PC cluster system for the reconstruction that is described in Sec. 2.1.

In Figures 12 and 13, *the mesh* represents a mesh which is generated by applying the discreet marching cubes method to a reconstructed volume, *the deformed mesh* represents a mesh which is generated by the method that is described in Sec. 2.3, and *the texture-mapped mesh* represents a mesh that is texture-mapped by the method that is described in Sec. 2.4. The color of the mesh and the deformed mesh denotes the surface normal by replacing (x, y, z) with (r, g, b). The figures illustrate that a slender wooden sword and the shape of kimono can be reconstructed, and the deformed mesh represents a more smooth and accurate shape. In particular, the waving of kimonos can be well represented, which is impossible for an ordinary motion capture system with a tight costume and markers. Some phantom volumes, however, appear around samurais' feet and between the samurais' arms, etc., because of the silhouette extraction error and the insufficiency of the mesh deformation. We are now contending with this issue as future work.

For reference, we show examples of 3D video frames at lower resolution (10 mm^3) for the MR-PreViz system in Figure 14. At this lower resolution, shape of samurais and wooden swords can be represented as well as the results at the higher one, and therefore, the data has sufficient resolution to utilize into the MR-PreViz system.

4.3 3D Video Visualization

In Figure 15, we show the visualized 3D video of the swordfight scene with a virtual CG set of a traditional Japanese downtown. The position and rotation of two samurais are configured manually, and the timing of the action is also edited, but it was almost perfect with their professional skills. 3D video can represent self and mutual occlusions of samurais with no difficulty since it has the 3D shapes of them, which is difficult for an ordinary image-based rendering which has no geometry information. As shown in Figure 15, we can observe a real swordfight from an arbitrary viewpoint and time.

5 Concluding Remarks and Future Work

In this paper, we have presented our 3D video technologies, which enable us to capture high fidelity full 3D shape, motion, and texture of high speed and dynamic actions. The experimental results have shown the utility of our system, which are

[3]The speed is manually estimated from the captured images.

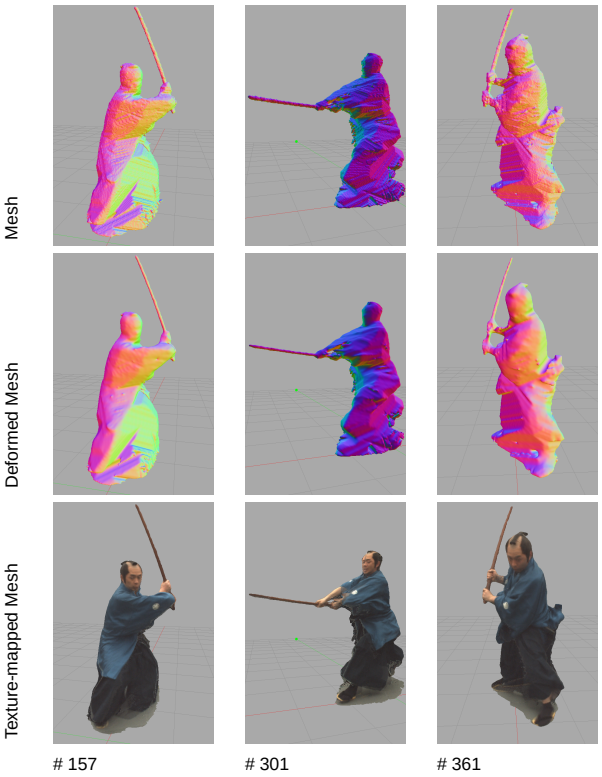


Figure 13: Examples of generated 3D video frames (Samurai 2, 5 mm^3). The color of the Mesh and Deformed Mesh denotes the surface normal by replacing (x, y, z) with (r, g, b).

- acquisition of synchronized multi-viewpoint image sequence with few motion blurs,
- real-time 3D volume reconstruction with a PC cluster,
- 3D shape recovery with a deformable mesh model,
- high fidelity texture mapping for a mesh with limited accuracy, and
- an interactive viewer which enables us to observe objects from an arbitrary viewpoint.

For the MR-PreViz system, we have generated 3D video of a swordfight scene in which professional actors making up as samurais are fighting a battle with slender wooden swords. The results show that our system can reconstruct a slender object that moves at high speeds, and represent the waving of kimonos successfully. We have also improved the interactive viewer in order to render longer sequence of 3D video by employing the simplified format of 3D video. For future work in the MR-PreViz project, we aim to develop a method to edit a shape and motion of 3D video, which enables us to edit actions of 3D video arbitrarily. As a first step of this, we have developed a method to obtain a 3D kinematic structure from 3D video using the augmented multiresolution Reeb graph [17, 14].

For future work of 3D video technology, we are developing a method for silhouette extraction from multi-viewpoint images using visual hulls. The concept of this method is similar to [20], our method, however, has functions of error detection and correction for reliable silhouette extraction. For generating 3D video of multiple objects in a wide area, we are developing a method to generate 3D volumes, tracking them with active cameras. In order to accomplish it, we are also developing precise calibration methods for a

Figure 14: Examples of generated 3D video frames for the MR-PreViz system (10 mm^3).

set of the active cameras. As other examples of future work, we need a compression method including texture data, while we presented a compression method for a mesh of 3D video [5]. For effective visualization, methods for estimating reflectance parameters of an object and lighting environment which has effects of near light sources are also required. Moreover, we are developing a real-time rendering system for 3D video with a full 3D display system.

Acknowledgement

The work of utilizing 3D video to the MR-PreViz system is supported by “Foundation of Technology Supporting the Creation of Digital Media Contents” project (CREST, JST). The technologies of 3D video are developed with a support of Ministry of Education, Culture, Sports, Science and Technology under the Leading Project, “Development of High Fidelity Digitization Software for Large-Scale and Intangible Cultural Assets.”

References

[1] The mr-previz project. http://www.rm.is.ritsumei.ac.jp/MR-PreVizProject/top.html.

[2] Open source computer vision library. http://www.intel.com/technology/computing/opencv/.

[3] E. Borovikov and Larry Davis. A distributed system for real-time volume reconstruction. In *International Workshop on Computer Architectures for Machine Perception*, pages 183–189, 2000.

[4] J. Carranza, C. Theobalt, M. A. Magnor, and H. Seidel. Free-viewpoint video of human actors. *ACM Transactions on Computer Graphics*, 22(3):569–577, July 2003.

[5] Hitoshi Habe, Yosuke Katsura, and Takashi Matsuyama. Skin-off:representation and compression scheme for 3d video. In *Picture Coding Symposium (PCS) 2004*, 2004.

[6] Takeo Kanade, P. Rander, and P. J. Narayanan. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE Multimedia*, pages 34–47, 1997.

[7] Yukiko Kenmochi, Kazunori Kotani, and Atsushi Imiya. Marching cubes method with connectivity. In *IEEE 1999 International Conference on Image Processing ICIP-99*, pages 361–365, Kobe, Japan, oct 1999.

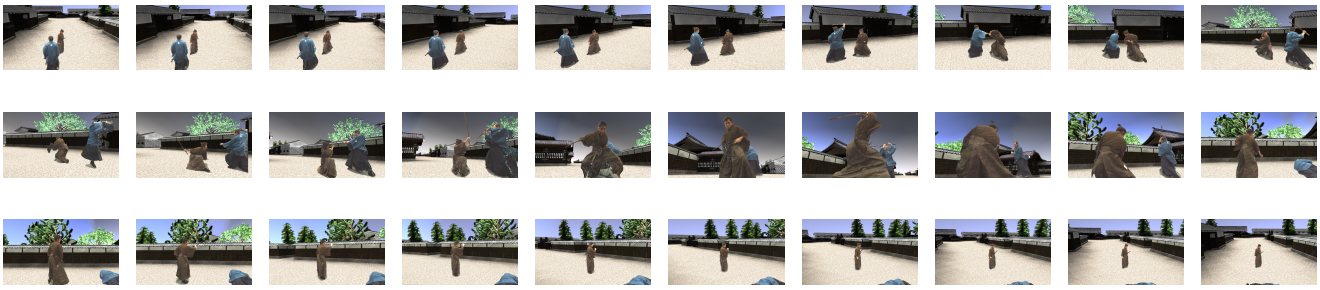


Figure 15: Examples of visualized 3D video.

[8] A. Laurentini. How far 3d shapes can be understood from 2d silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):188–195, 1995.

[9] Takashi Matsuyama and Takeshi Takai. Generation, visualization, and editing of 3d video. In *3DPVT*, pages 234–245, Padova, Italy, jun 2002.

[10] Takashi Matsuyama, Xiaojun Wu, Takeshi Takai, and Shohei Nobuhara. Real-time generation and high fidelity visualization of 3d video. In *Proceedings of Mirage 2003*, pages 1–10, 2003.

[11] Takashi Matsuyama, Xiaojun Wu, Takeshi Takai, and Shohei Nobuhara. Real-time 3d shape reconstruction, dynamic 3d mesh deformation and high fidelity visualization for 3d video. *CVIU*, 96:393–434, December 2004.

[12] Takashi Matsuyama, Xiaojun Wu, Takeshi Takai, and Toshikazu Wada. Real-time dynamic 3d object shape reconstruction and high-fidelity texture mapping for 3d video. *IEEE Transactions on Circuit and Systems for Video Technology*, 14:357–369, 3 2004.

[13] S. Moezzi, L. Tai, and P. Gerard. Virtual view generation for 3d digital video. *IEEE Multimedia*, pages 18–26, 1997.

[14] Tomoyuki Mukasa, Shohei Nobuhara, Atsuto Maki, and Takashi Matsuyama. Finding articulated body in time-series volume data. In F. J. Perales and R. B. Fisher, editors, *The 4th International Conference on Articulated Motion and Deformable Objects (F. J. Perales and R. B. Fisher: AMDO 2006, LNCS 4069)*, pages 395 – 404, 2006.

[15] Shohei Nobuhara and Takashi Matsuyama. Heterogeneous deformation model for 3d shape and motion recovery from multi-viewpoint images. In *3DPVT*, pages 566–573, 2004.

[16] Shohei Nobuhara and Takashi Matsuyama. Deformable mesh model for complex multi-object 3d motion estimation from multi-viewpoint video. In *3DPVT*, 2006.

[17] Tony Tung and Francis Schmitt. The augmented multiresolution reeb graph approach for content-based retrieval of 3d shapes. *International Journal of Shape Modeling (IJSM)*, 11(1):91 – 120, June 2005.

[18] T. Wada, Xiaojun Wu, Shingo Tokai, and Takashi Matsuyama. Homography based parallel volume intersection: toward real-time reconstruction using active camera. In *International Workshop on Computer Architectures for Machine Perception*, pages 331–339, 2000.

[19] Xiaojun Wu, Osamu Takizawa, and Takashi Matsuyama. Parallel pipeline volume intersection for real-time 3d shape reconstruction on a pc cluster. In *The 4th IEEE International Conference on Computer Vision Systems*, 2006.

[20] Gang Zeng and Long Quan. Silhouette extraction from multiple images of an unknown background. In *ACCV*, pages 628 – 633, 2004.