IPSJ Journal Vol. 2 No. 8 169–179 (Aug. 2010)

Regular Paper

Cell-based 3D Video Capture of a Freely-moving Object Using Multi-Viewpoint Active Cameras

Tatsuhisa Yamaguchi ,†1 Hiromasa Yoshimoto ,†1,∗1
Shohei Nobuhara †1 and Takashi Matsuyama †1

We propose a method to capture 3D video of an object that moves in a large area using active cameras. Our main ideas are to partition a desired target area into hexagonal cells, and to control active cameras based on these cells. Accurate camera calibration and continuous capture of the object with at least one third of the cameras are guaranteed regardless of the object's motion. We show advantages of our method over an existing capture method using fixed cameras. We also show that our method can be applied to a real studio.

1. Introduction

3D video[9] is a full 3D dynamic shape and texture data generated from multi-viewpoint video. A number of studies have proposed 3D video generation methods[2],[3],[5],[9],[12]. These methods first capture target objects as a multi-view video by a set of calibrated video cameras that surrounds the objects, and then generate a 3D video using shape reconstruction algorithms such as shape-from-silhouette[8], multi-view stereo or space carving[7], etc. Namely, these methods are based on the analysis of images using camera geometry and photometry cues that are obtained separately. Consequently, the requirements for 3D shape reconstruction from multi-viewpoint images can be summarized as follows:

- Req. 1: Camera calibration
- Req. 2: Visual coverage
- Req. 3: Spatial resolution

Requirement 1 states that the cameras must be calibrated accurately. The second requirement states that every point on the target objects must be simultaneously captured from different viewpoints. Lastly, the objects must be captured with high enough spatial resolution. In other words, 3D video can be generated in the space where these three requirements are satisfied. We call such a space the "capture space." Starck et al.[13] have summarized practical 3D video studio considerations. They argue that spatial sampling resolution by cameras defines both reconstruction accuracy and spatial resolution of texture. They also point out that existing 3D video capture systems typically use a fixed set of cameras focused on a restricted volume. In such static camera systems, reconstruction accuracy and spatial resolution of texture are effectively governed by the number of sampling elements on each camera and the focal lengths of each camera. It means that, where the number of used cameras and their output image resolution are restricted, a trade-off problem arises between reconstruction accuracy and the size of capture space. For example, if we choose a wider angle of view to extend the capture space, the spatial resolution becomes lower. Use of active cameras can overcome this trade-off problem, but it is difficult to calibrate the active cameras accurately and robustly, especially in the presence of large changes in focal length. Additionally, all the active cameras must be controlled in real-time to satisfy requirements 2 and 3.

Previously the authors have proposed a cell-based concept to capture 3D video with active cameras[16]. Cell-based algorithms are characterized by the following steps. The first is to divide the space where the object moves into subspaces named "cells." The second is to precompute a set of dedicated camera control parameters that satisfy the visual coverage and spatial resolution requirements in each cell. Then we apply static camera calibration methods for each cell before capture. The third is the cell-based camera control. The cameras are controlled so as to satisfy Req. 2 by "watching" one of the cells, using the precomputed control parameter; each camera selects which cell to watch depending on the object position, but does not follow the object itself continuously. Consequently, cell-based algorithms discard camera images when the camera view is switched. Based on this concept, we also have proposed an algorithm to capture 3D video of an object in 16). However, the algorithm assumed that the object moves along a given path. Therefore, the algorithm cannot be used to capture freely-moving

†1 Kyoto University
∗1 Presently with Enegate Co., Ltd

objects.

This paper extends our previous work in an important way: here, we tackle the 3D video capture problem of a *freely-moving* object. This work removes the assumption about a priori knowledge of the object's trajectory. We are able to achieve this by developing new strategies for cell placement and camera control.

The rest of the paper is organized as follows: Section 2.1 describes the general idea of our cell-based algorithm. Sections 2.2 through 2.8 describe our algorithm in detail. Section 3 shows the effectiveness of our algorithm through some experiments. Section 4 concludes this work with discussions and possible future research directions.

2. Cell-Based 3D Video Capture Method

2.1 Idea

There are two essential design decisions to be made when building a cell-based algorithm; how to divide the space into cells and how to control active cameras based on the cells. Since the object can move freely, shape reconstruction should be performed equally well regardless of the object position and its direction of movement. Thus the division and the cell shape should be both homogeneous and isotropic. As a natural solution, we adopt a regular hexagon tessellation for cell arrangement.

The second design decision is the camera control rule based on these cells to ensure the continuous 3D video capture of a freely-moving object. Our cell-based method reduces this problem to an assignment problem of the cameras to the cells. The hexagonal cell arrangement has a 6-neighborhood structure. Thus any object movement can be expressed by a movement to one of the six cells. The capture system must be ready to capture the object for all of these directions. One intuitive and possible solution is to assign the cameras to the cell where the object exists and the six cells next to it. However, it is not desirable from the standpoint of Req. 2, since it can only ensure capture by one seventh of the cameras. Thus we have devised a more efficient way by reducing the number of the cells to be covered. For example, another simple and intuitive method is to divide the cameras into two groups and to assign them to watch the cell where the object exists and the next cell that the object is moving to. However, this method fails when the object passes across a cell vertex where three cells are in contact with each other. For example, when the object is located at $\vec{p}$, as shown in **Fig. 1**, it can either go into cell B, C, or stay in cell A. The capture system must anticipate all of these three cases by watching the three cells. This proves that the capture system must be able to observe at least 3 cells simultaneously.

We propose a method to capture the three nearest cells from the object using three homogeneous camera groups. We prove that this is also a sufficient number. For example, assume that the object approaches cell D that is currently the fourth-nearest. D becomes the third-nearest cell instead of C when the object arrives at $\vec{p'}$. At this point, the capture system no longer needs to watch cell C. Consequently, the cameras that have been assigned to cell C can change views to watch cell D. Additionally, the distance to D is not zero but equal to or longer than $R/2$, where R is the radius of the cells as described in Fig. 1. This distance gives time for the cameras to switch their views before the object reaches cell D. Similarly, cell G can be covered by the same cameras. Cells E and F can be covered by the cameras currently watching cell B. This discussion is valid regardless of the cell the object lies in, by associating the three camera groups to three sparse subsets of the cells as shown in **Fig. 2**. In this way, our method ensures continuous capture of the object at least with one third of the cameras. To generalize, the maximum number of cells that share a single vertex gives the minimum number of required camera groups. Hence a hexagonal cell shape is also the best from this standpoint.

The rest of this section describes each step of our method in detail.

2.2 Problem Formulation

Our method can be applied to the following situation:

(1) Only one target object.
(2) The object movement is restricted to a given target area.
(3) The object's maximum velocity is given.
(4) The object's size is given.
(5) Active cameras are set up to surround the target area.
(6) The object is not occluded by other objects from any camera.
(7) The lowest allowable spatial resolution is given.

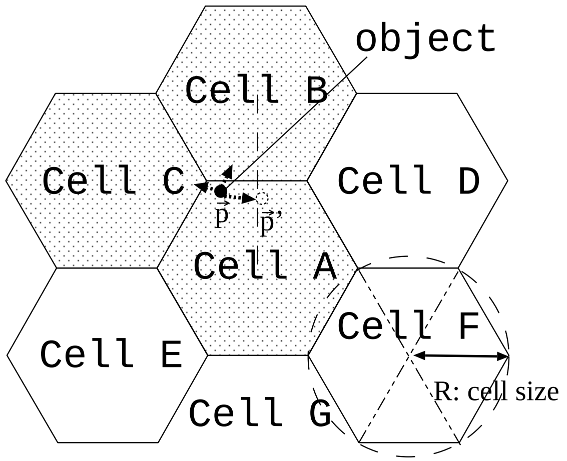


Fig. 1 Cells and the object. A is the cell where the object is present, and thus the nearest cell from the object. B, C and D are the second-, third-, and fourth-nearest cells, respectively. The capture system must be ready to capture A, B and C, especially when the object is nearby the vertex shared by these cells. On the other hand, cell D can be covered by the cameras that watched C before the object approaches D.

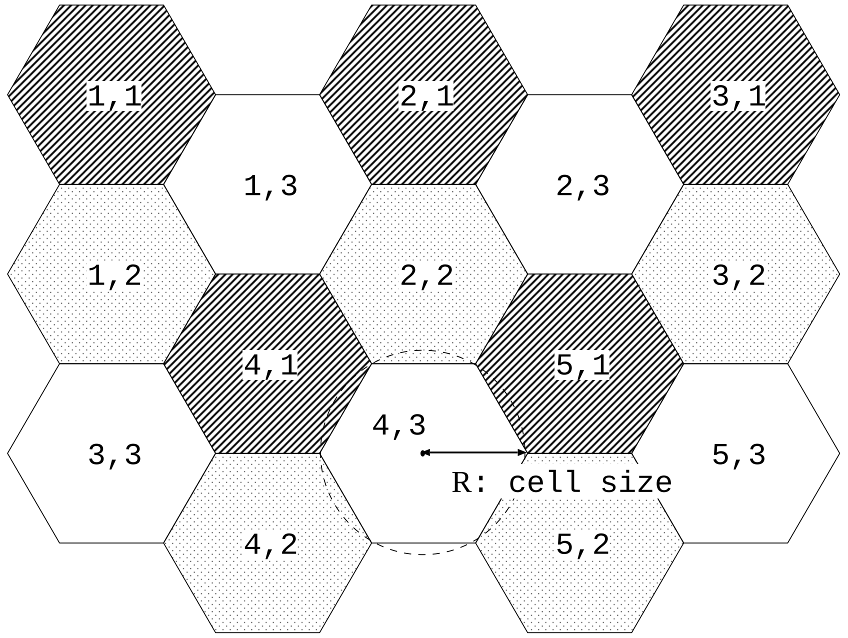


Fig. 2 Cell assignment for three camera groups. The numbers in the figure represents j, l = cell index and group index.

Active Camera Model and Spatial Resolution

We assume that each active camera can be approximated by a partially-fixed viewpoint pan-tilt-zoom (PFV-PTZ) camera model[10)6)]. A PFV-PTZ camera is a camera whose projection center is encased in a limited volume around the rotation center. The changes in their projection centers are relatively small when capturing objects far enough from such cameras. We approximate the projection center of an active camera by a steady point regardless of the pan, tilt, zoom and focus motions of the camera. Therefore the camera positioning problem can be separated from the active camera control problem. We also assume that the cameras are mounted on stations surrounding the target area. The cameras change directions and focal lengths according to each camera control parameter, which consists of pan, tilt, zoom and focus positions. We denote camera control parameters for camera i as $\mathbf{e}_i$. Since these parameter changes accompany physical motions of the cameras, there exist time lags between the transmission of these parameters and the end of the corresponding motions. The lengths of these time lags depend on the current and target state of each camera, but are guaranteed to be shorter than τ_{s} seconds.

Spatial resolution of reconstructed 3D video is governed by several factors; e.g. sampling resolution by the cameras, shape reconstruction algorithm, actual object shape, relative positions of viewpoints and the object, etc. Among these factors, we only consider sampling resolution by the cameras. In this paper, we represent spatial resolution by the minimum distance between two nearest distinguishable points on the object. Spatial resolution realized by a single camera is governed by its focal length, number of sampling elements in the camera and the distance to the object from the camera. It means that, each camera has a depth limit of the view depending on its focal length.

We define the view frustum of each camera — the region of the space that

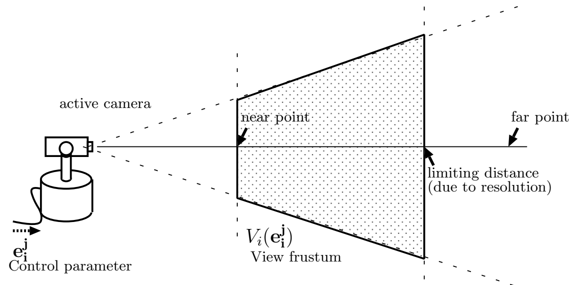


Fig. 3 View frustum.

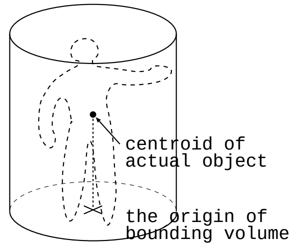


Fig. 4 An example of bounding volume that represents the maximum allowable size of the object, in case of a cylinder.

can be captured by camera i — reflecting the lowest allowable spatial resolution. Each view frustum is modeled as a rectangular frustum. The distance to the near plane is governed by the near point of focus. On the other hand, the distance to the far plane is governed by both the far point of focus and the distance limit due to the spatial resolution, as shown in **Fig. 3**. The shorter one governs the distance to the far plane. The view frustum of each camera is a function of the camera control parameter. We denote the view frustum of camera i, whose control parameter is set to $\mathbf{e}_i$, as $V_i(\mathbf{e}_i) \subset \mathbb{R}^3$.

Target Object

The target object moves within a target area at varying speeds that is slower than $v_{\mathtt{max}}$. The object's size is specified by the user as a bounding volume. The object can change its shape within the bounding volume located at the same position with the object, as shown in **Fig. 4**. Any shape can be used as the bounding volume, though it would be natural to use a solid of revolution, e.g. a cylinder or a hemisphere, since the object is supposed to move in arbitrary directions.

We adopt the world coordinate system that has the origin at the center of the target area and the z axis directed upright. We represent the object state by the projection point of its centroid to the floor and denote it by the 2D coordinates $\vec{p} = (x, y)$. In other words, we define a 2D state space which can be mapped to the floor plane and represent the object state by a point in this state space.

By making the assumptions about the active cameras and the object, one problem instance for our algorithm consists of the variables listed in Tables 1 and 2.

Table 1 Scenario

$\mathbf{D} \subset \mathbb{R}^2$	Target area, within which the object is allowed move during capture.
$v_{\mathtt{max}}$	Maximum speed of the object.
$\mathbf{B} \subset \mathbb{R}^3$	Bounding volume of the object.
s[mm/pixel]	Lowest allowable spatial resolution.

Table 2 Resources

$N_{\mathtt{cam}}$	Number of active cameras.
$\mathrm{O}_1, \mathrm{O}_2, \ldots, \mathrm{O}_{\mathrm{N}_{\mathtt{cam}}}$	Active camera positions.
$\tau_{\mathtt{s}}$	Upper limit of the time required for changing active camera state.
$\tau_{\mathtt{cam}}$	Video capture interval.

A scenario is given by users reflecting the scene to be captured. On the other hand, resources are governed by the studio equipment. The output of our algorithm is a 3D video of a moving object in a widespread area. The algorithm consists of the following processes.

(1) Camera grouping
(2) Cell generation
(3) Camera view adjustment
(4) Camera calibration
(5) Real-time tracking
(6) 3D Video generation

2.3 Camera Grouping

Due to the reasons described in section 2.1, the first step of our algorithm divides the cameras into three groups. Since only one of the groups is available for capture at the worst case, every group should have enough cameras to satisfy Req. 2 by itself. Accordingly, it is desirable that the cameras in each group are equally distributed in all directions. Thus we adopt the following scheme.

(1) Represent the camera positions by spherical coordinates (r_i, θ_i, ϕ_i) where θ_i is the zenith angle that satisfies $0 \leq \theta_i \leq \pi/2$, and ϕ_i is the azimuth angle that satisfies $0 \leq \phi_i < 2\pi$.

(2) Sort the camera indices according to ϕ_i. Let the sorted camera indices be $i_1, i_2, \ldots, i_{N_{\mathtt{cam}}}$.

(3) For all $n = 1, 2, \ldots, N_{\mathtt{cam}}$, assign camera i_n to group $(n - 3\lfloor n/3 \rfloor + 1)$.

In the following description, we express the camera groups by $\mathbf{A}_1, \mathbf{A}_2$ and $\mathbf{A}_3$ where $\mathbf{A}_l = \{i | \text{camera } i \text{ is assigned to group } l\}$. We also use the notation $l(i)$ to mean "the group index that camera i is assigned to."

2.4 Cell Generation

The second step divides the target area into cells using regular hexagon tessellation shown in Fig. 2 and assigns them to the camera groups. Our algorithm represents a cell as a joint subset of the target area. The cells are assigned to the three camera groups exclusively as shown in Fig. 2. Thus we denote cells using two indices; group index l and cell index j.

$$\mathbf{C}_{j,l} \subset \mathbf{D}(l = 1, 2, 3; j = 1, 2, \ldots, N_l^{\mathtt{cell}}) \tag{1}$$

Here, $N_l^{\mathtt{cell}} \in \mathbb{N}$ is the number of the cells assigned to group l. We also define the following symbols. They are precomputed later in the following steps.

$\mathbf{e}_i^j$ Camera control parameters for directing camera i to $\mathbf{C}_{j,l(i)}$. $\mathbf{e}_i^j$ consists of pan, tilt and zoom values.

$E_i(\mathbf{e}_i^j)$ Camera parameters — intrinsic and extrinsic, geometric and photometric parameters — of camera i with the state specified by $\mathbf{e}_i^j$.

We define the distance between the object located at $\vec{p}$ and a cell $\mathbf{C}_{j,l}$ by Eq. (2). It is defined as the distance from the object to the nearest point in the cell.

$$d(\vec{p}, \mathbf{C}_{j,l}) = \min_{\vec{q} \in \mathbf{C}_{j,l}} ||\vec{q} - \vec{p}|| \tag{2}$$

Cell arrangement by the regular hexagon tessellation shown in Fig. 2 has four degrees of freedom: 2D displacement, rotation, and the size of the cells. The displacement and the rotation does not affect the 3D video capture as long as the cameras are not too close to the target area. This is because the cameras in each group are equally distributed in all directions by the camera grouping step described in section 2.3. Thus we give these parameters manually. On the other hand, the size of the cells is critical and must be designed while considering the camera control rule.

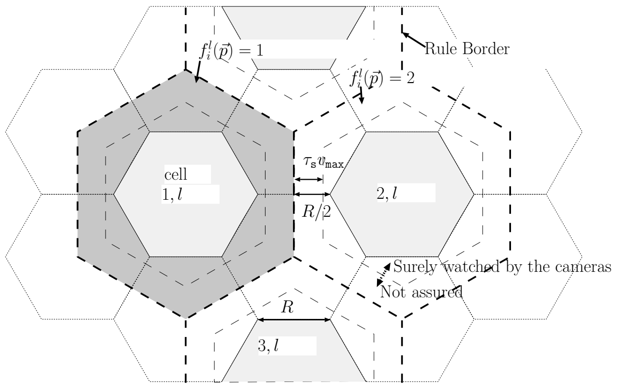


Fig. 5 Visualization of camera control rule for a single group l.

Camera Control Rule and Cell Size

Our camera control method directs all the cameras in $\mathbf{A}_{l_0}$ to the nearest cell out of $\{\mathbf{C}_{j,l} | l = l_0\}$ from the object. If there are two or more nearest cells, the cameras should be directed to one of those cells. This control rule can be represented by functions that map an object position to a cell index:

$$f_l(\vec{p}) = \operatorname*{argmin}_j d(\vec{p}, \mathbf{C}_{j,l}) \tag{3}$$

Our cell arrangement shown in Fig. 2 gives the camera control rule as shown in **Fig. 5**. The cameras in group l begin to switch views when the object goes across a rule border for group l, as shown in Fig. 5. Switching camera views from one cell to another cell requires a certain amount of time shorter than $\tau_{\mathtt{s}}$. During these periods, the images from such "in-motion" cameras cannot be used for shape reconstruction because they are not calibrated. Meanwhile, the object can move by $\tau_{\mathtt{s}} v_{\mathtt{max}}$ in the worst case. It means that where the distance from the nearest rule border for group l is shorter than $\tau_{\mathtt{s}} v_{\mathtt{max}}$, the object is not guaranteed to be captured by the cameras in $\mathbf{A}_l$. Thus the necessary and sufficient condition for cameras in $\mathbf{A}_l$ to ensure observation of the object in cell $\mathbf{C}_{j,l}(j = 1, 2, \ldots, N_l^{\mathtt{cell}})$ with respective state $\mathbf{e}_i^j$, for arbitrary object movement, is represented by Eq. (4). Since the object exists in one of the cells at any time, this is also a sufficient condition to capture the object continuously, with at least one of the three camera groups.

$$R \geq 2\tau_{\mathtt{s}} v_{\mathtt{max}} \tag{4}$$

Eq. (4) means that the minimum size of the cells is limited by $\tau_{\mathtt{s}}$ and $v_{\mathtt{max}}$. The larger the cell is, the larger space the camera views must cover and thus the spatial resolution becomes lower. In order to maximize the spatial resolution, we adopt the minimum allowable cell size as shown by Eq. (5).

$$R = 2\tau_{\mathtt{s}} v_{\mathtt{max}} \tag{5}$$

To summarize, our cell generation algorithm is described as follows.

(1) Compute the cell size by Eq. (5).
(2) Give a cell position and rotation manually.
(3) Generate other cells by dividing $\mathbf{D}$ according to the hexagon tessellation pattern shown in Fig. 2.

2.5 Camera view adjustment

This step adjusts dedicated camera control parameters in order to watch each cell. In order to guarantee the object capture at any point in a cell, every camera view in group l should include the Minkowski sum of $\mathbf{C}_{j,l}$ and $\mathbf{B}$, as shown in **Fig. 6**. We call this volume the "common view" of cell j, l and denote it by $M_{j,l}$. Thereby the condition is described by Eq. (6).

$$M_{j,l} \subset V_i(\mathbf{e}_i^j) \tag{6}$$

If such $\mathbf{e}_i^j$ does not exist, it means that the object cannot be captured with our method for the given scenario. Our algorithm terminates at this step in this case.

The algorithm to find $\mathbf{e}_i^j$ should be designed depending on the structure of the active cameras in use. We describe one example algorithm that gives $\mathbf{e}_i^j$ for a PFV-PTZ active camera: First, adjust the zoom value so that the spatial resolution becomes s at the farthest point in $M_{j,l}$. Next, adjust the focus value so that the nearest and the farthest points in $M_{j,l}$ to the camera is included within the field of view. Finally, adjust the pan/tilt angles so that $M_{j,l}$ is included in the image frame of the camera. If one or more of these parameters do not exist, then there is no solution.

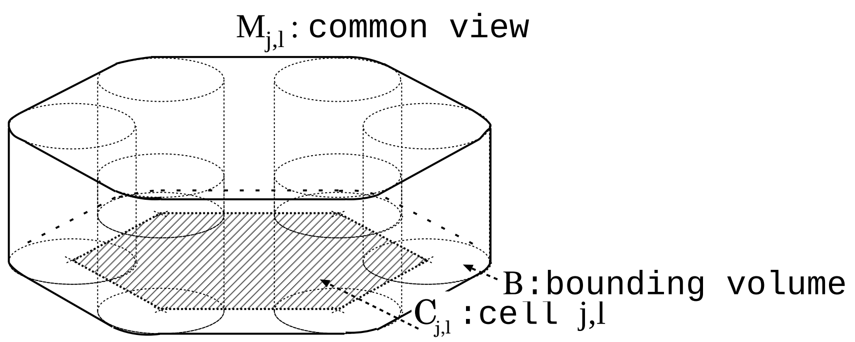


Fig. 6 Minkowski sum of a cell and the bounding volume.

2.6 Camera Calibration

We calibrate the active cameras and obtain $E_i(\mathbf{e}_i^j)$ for all $i = 1, \ldots, N_{\mathtt{cam}}, j = 1, \ldots, N_{l(i)}^{\mathtt{cell}}$. At this step, we do not utilize any explicit active camera model such as the PFV-PTZ camera model. All the active cameras can be regarded as fixed cameras, while they watch one of the cells. Namely, we have $\sum_{l=1}^{3} ||\mathbf{A}_l||N_l^{\mathtt{cell}}$ virtual cameras where $||\mathbf{A}_l||$ is the number of the active cameras forming camera group l. Any existing camera calibration methods for fixed cameras, such as Zhang's[17] and Svoboda's[14], can be applied to these virtual cameras. Since this is an off-line process separated from the following tracking process, we can use any calibration targets such as calibration charts or a point light source to obtain camera parameters accurately and robustly.

2.7 Real-time tracking

The cell generation and the camera view adjustment process define the camera control rule — how all the cameras should be controlled for any object position. Additionally, the camera calibration process gives accurate camera parameters that enable 3D positioning of the object by the images. Thus the tracking can be performed by computing the object position from the images and controlling the active cameras in parallel.

We describe our tracking process using a model of networked computers. The capture process is performed by $N_{\mathtt{cam}}$ camera nodes $\pi_i (i = 1, 2, \ldots, N_{\mathtt{cam}})$, which have one camera connected to each, and one master node $\pi_{\mathtt{M}}$. The nodes are connected to each other to share the object position. In the following description, we denote the time by t and we assume that all the system clocks on the nodes are synchronized.

The measurement of the object 3D position is performed as follows. Each π_i repeats the following procedure in every time interval $\tau_{\mathtt{cam}}$.

2D Tracking Process on each π_i

(1) Grab an image $I_i(t)$.

(2) If the camera is not in motion but watching one of the cells in group $l(i)$, let its cell index be $j_i(t)$

 (a) Store $(t, I_i(t), j_i(t))$.

 (b) Find the object in the image and compute its centroid coordinate $\vec{u}_i(t)$.

 (c) If $\vec{u}_i(t)$ is successfully computed, transmit $(t, \vec{u}_i(t), j_i(t))$ to $\pi_{\mathtt{M}}$.

3D Tracking Process on $\pi_{\mathtt{M}}$

(1) When 2 or more sets out of $\{(t, \vec{u}_i(t), j_i(t)) \,|\, i = 1, \ldots, N_{\mathtt{cam}}\}$ have been received,

 (a) Compute the 3D position of the object, $\vec{\hat{p}}(t) = (x, y, z)$, by triangulation using $\vec{u}_i(t)$ and $E_i(\mathbf{e}_i^{j_i(t)})$.

 (b) Transmit $\vec{p}(t) = (x, y)$ to all π_i.

Finally, the camera control is performed as follows:

Camera Control Process on each π_i

(1) Whenever a new $\vec{p}(t) = (x, y)$ is received,

 (a) Transmit $\mathbf{e}_i^{f_{l(i)}(\vec{p}(t))}$ to the active camera and begin to switch its posture, zoom and focus.

2.8 3D Video generation

In the algorithm described above, each π_i stores $(t, I_i(t), j_i(t))$. From these data, a sequence of multi-view images and camera parameters $\left(I_i(t), E_i(\mathbf{e}_i^{j_i(t)})\right)$ can be obtained. Hence our method can generate a 3D video.

3. Experiment

We show the effectiveness of our method from the standpoint of Reqs. 2 and 3 by two kinds of experiments. First, we compare our method with a method with static cameras using a simulation. Second, we show that our method can be applied to real environments.

Table 3 Scenario and resources for the simulation.

$\tau_{\mathtt{s}}$	$v_{\mathtt{max}}$	$\mathtt{s}$	$\mathbf{B}$
1.0[s]	300[mm/s]	8[mm/pixel]	A cylinder 900[mm] in diameter and 1800[mm] in height

3.1 Quantitative Evaluation by Simulation

We compare our method with one possible capture method using static cameras, which adjusts all the camera directions and focal lengths so as to include the entire target space. We denote this method as "Fixed, wide."

We adopt these four indices for the comparison.

(1) Viewpoint usage

(2) Pixel usage

(3) Accuracy and completeness[11] of the reconstructed shape

(4) Peak signal-to-noise ratio (PSNR) of synthesized images

We used 3D video sequences of a walking person as the virtual scene to be captured. **Fig. 7** shows one of the 3D video frames. Since all of these indexes are also dependent on the object motion and its trajectory, we used two different sequences. **Fig. 8** shows the object trajectories in each sequence. We also used these 3D video sequences as the ground truth shape for evaluation of the 3D shape reconstruction.

We arranged 24 cameras to ensure the object capture by at least 8 cameras at any frame. According to the study by Jonathan et al.[13], 8 cameras is almost sufficient to attain 100mm reconstruction accuracy using the shape-from-silhouette algorithm when capturing a person in a similar 3D video studio as ours. **Fig. 9** shows the arrangement of 24 active cameras and the target area. Other resources and scenario parameters are shown in **Table 3**. We assumed that time lags by camera motions are uniform and equal to $\tau_{\mathtt{s}}$ for all combinations of cameras and cells to be watched.

We applied our cell generation algorithm for these scenario and resources. The cell configuration is shown in Fig. 8. Then, the active cameras were virtually controlled to capture the object and their images were synthesized by rendering the virtual scene. Finally, the 3D shapes of objects were reconstructed from

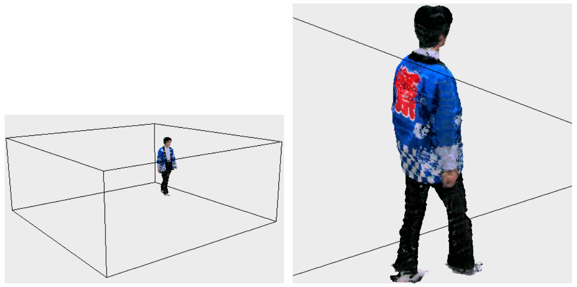


Fig. 7 One of the frames in the 3D video sequence used for the simulation.

the synthesized images. We used a graph-cut based 3D shape reconstruction algorithm of 15) without the super-resolution process.

3.1.1 Viewpoint Usage

The camera control results for the two sequences are summarized in **Fig. 10**. It shows which cell is watched by the cameras in each group at each frame, and the number of cameras that contributed to the shape reconstruction. These figures show that the viewpoint usage exceeded 1/3 in most frames. One of the reasons is that the common views, $M_{j,l}$ overlap between two neighboring cells. Furthermore, the cameras can also observe outside $M_{j,l}$. Thus they could contribute to the reconstruction of the object shape, by partially or fully including the object in their images.

3.1.2 Pixel Usage

Table 4 and **Fig. 11** shows the average pixel usage defined by Eq. (7).

$$\text{(Pixel usage at frame } k) = \frac{1}{|G(k)|} \sum_{i \in G(k)} \frac{N_i^{\mathrm{P}}}{N_i^{\mathrm{I}}} \tag{7}$$

$G(k)$	$\{i \mid$camera i not in motion but captured the object at frame $k\}$
N_i^{P}	Number of pixels occupied by the object in image i
N_i^{I}	Number of pixels in image i. e.g. 307200(=640×480) for VGA.

The average pixel usage was 6.9% with our method and 2.0% with "Fixed, wide." The pixel usage by our method is roughly 3.5 times larger than "Fixed, wide" method. Thus the spatial resolution is about 1.8 times finer in our method in these cases.

Table 4 Pixel usage. Each cell shows the average ± standard deviation for the sequence.

	Proposed[%]	Fixed[%]
Sequence 1	6.7 ± 0.32	1.9 ± 0.10
Sequence 2	7.2 ± 0.45	2.0 ± 0.16

3.1.3 Shape Reconstruction Accuracy and Completeness

Fig. 12, **Fig. 13** and **Fig. 14** summarizes the accuracy and completeness[11] of the reconstructed shapes by the two methods. On average, our method performed equally well or better than the "Fixed, wide" method.

At the frames where the viewpoint usage dropped to 1/3, the completeness by our method lowered significantly but the accuracy was not degraded. In other frames, where more than 2/3 of the viewpoints were available, our method outperformed the "Fixed, wide" method in both measures. **Fig. 15** shows the reconstructed shape at frame 135 in sequence 1. Our method resulted in poorer completeness due to the lower number of viewpoints. We can see a larger error at the chest part of the object with our method. This part was poorly reconstructed due to self-occlusion. However, the accuracy as a whole was better than the "Fixed, wide" method. **Fig. 16** shows the frame where our method resulted in better accuracy and completeness despite the smaller number of viewpoints. One possible reason to describe both results is that, as long as the object surface can be observed by multiple cameras, stereo matching can be performed more accurately with the help of finer texture cues obtained by our method. As shown in **Fig. 17**, the "Fixed, wide" method produced a smaller image of the object, affecting the shape reconstruction accuracy.

3.1.4 PSNR of Synthesized Images

Finally, we evaluate the appearance of generated 3D videos. For each frame, a virtual viewpoint was set up in front of the object. The viewpoint was located three meters away from the object and its angle of view was set to 32 degrees, with which the entire object can be observed by the virtual camera. The original and the reconstructed models were rendered and the images were compared to compute the PSNR for each frame. The results are summarized in **Fig. 18**. Our method resulted in a higher PSNR at every frame. The images for frames 135 and 95 in sequence 1 are shown in Fig. 15 and Fig. 16. At frame 135, although the shape reconstruction completeness is not as good as the "Fixed, wide" method, our method resulted in better image appearance. One of the reasons is that the "Fixed, wide" method generated coarser textures.

3.1.5 Conclusion

In our method, the number of viewpoints decreases to 1/3 for the worst frame. Lack of viewpoints sometimes leads to poorer completeness compared to the "Fixed, wide" method. However, our method can maintain the accuracy of 3D video even with those frames, by capturing object surfaces in higher spatial resolution. Additionally, it also provides finer textures on the object. Having these advantages, the overall accuracy of the 3D video and its appearance were significantly improved.

3.2 Studio Experiment

We have also tested our method in a physical setup. We arranged 23 active cameras in our 3D video studio, which is about 8 meters square. The studio and the camera setup are shown in **Fig. 19**. Each active camera consists of a zoom camera SONY DFW-VL500 and a PTU-46 pan-tilt unit by Directed Perception, Inc. We set up 23 computers as camera nodes and 1 computer as the master node. All the nodes were connected by Ethernet and communication between the nodes were implemented by UDP. The system clocks were synchronized by NTP.

We captured a stuffed toy on a radio control car as shown in **Fig. 20**. We have set scenario parameters as shown in **Table 5** and Fig. 19. Note that the object and the parameters were chosen reflecting the limitations of our studio equipment, not our algorithm itself. We chose a small object because the studio was not large enough and the cameras were too close to capture a whole body of a person. A capture system for a walking person using our algorithm can be realized by scaling up the studio, target area, $v_{\mathtt{max}}$, **B** and s. For example, if we set up a studio 4 times larger than ours, the cameras can be placed 4 times farther and it would enable the capture of a person that walks at 832[mm/s]. The target area size is 12m × 8m, and a spatial resolution of 20[mm/pixel] can be achieved.

Generated cells are shown in **Fig. 21**. 17 cells were generated in total. The number of camera control parameters is shown in **Table 6**. For example, the first group consists of eight active cameras being assigned to seven cells, which

Table 5 Scenario and resources for studio experiment.

$\tau_{\mathtt{s}}$	$v_{\mathtt{max}}$	s	**B**
1.2[s]	208[mm/s]	5[mm/pixel]	A cylinder 800[mm] in diameter and 500[mm] in height

Table 6 Number of the cameras, cells and virtual cameras for each group.

l	$\|\|\mathbf{A}_l\|\|$	$N_l^{\mathtt{cell}}$	$\|\|\mathbf{A}_l\|\| \times N_l^{\mathtt{cell}}$
1	8	7	56
2	8	5	40
3	7	5	35

produces $8 \times 7 = 56$ virtual cameras. In this case, we had 131 virtual cameras in total. We calibrated the cameras in two steps. Firstly, intrinsic parameters were estimated by Zhang's method[17] for the 131 virtual cameras independently. Secondly, the extrinsic parameters were estimated by the eight-point algorithm[4] for each pair of cameras with 2D-to-2D correspondences of unknown 3D points ⋆1, and then refined through a bundle adjustment process which minimizes the sum of symmetric epipolar distances of all the cameras. At this step, all the virtual cameras corresponding to the same cell can be calibrated simultaneously. Moreover, the virtual cameras corresponding to three adjacent cells surrounding a vertex can be calibrated simultaneously to reduce the calibration work load. Thus the number of extrinsic calibration tasks required in total depends on the cell arrangement. At least $\max_l N_l^{\mathtt{cell}}$ calibration tasks are required. Some additional calibration tasks are needed for unifying the world coordinate systems by the result of each calibration process. In our case we performed this step by 10 calibration tasks for the different combinations of the cells.

The object moved on the path shown in Fig. 21 at varying speeds that is slower than $v_{\mathtt{max}}$, and was captured by our algorithm. **Figure 22** shows the viewpoint usage. The object was captured by 7 or more viewpoints at any frame. Since the object moved back and forth, it travelled across rule borders several times around frame 400. The viewpoint usage is low because the cameras in the two groups switched their views several times between cells in order to follow the object.

⋆1 We adopted a similar implementation with Svoboda's[14]. We captured a moving point light source as a multi-view video in order to obtain 2D-to-2D correspondences robustly.

During this period, the other one group could capture the object continuously. Another difference from the simulation in section 3.1 is that some cameras could finish switching its view in less than $\tau_{\mathtt{s}}$ seconds. Even in this case, our algorithm can guarantee that the object is continuously captured with more than $\lfloor N_{\mathtt{cam}}/3 \rfloor$ cameras. **Figure 23** shows some examples of captured images with object positions. We can see that the object size in images is almost uniform regardless of the distance from the camera. Finally, the object shape in each frame was reconstructed using a graph-cut based 3D shape reconstruction algorithm of 15) without the super-resolution process. **Figure 24** shows a rendered example of the reconstructed 3D video. We can see that fine details, such as the harness or letters written on the car, are successfully reconstructed.

4. Conclusion and Future Work

We have proposed a cell-based 3D video capture method that can capture a freely-moving object. Our method precomputes camera control parameters for individual cells in an off-line process. These parameters are later used to control cameras during the capture process. Our method allows us to calibrate the active cameras accurately in the same manner as static cameras. Our method also ensures that the object is captured using at least one third of the cameras. We have shown that our method can capture a moving object with higher resolution compared to an existing method with static cameras.

One of the remaining problems is the optimization of camera control rules — camera grouping, cell arrangement and camera views adjustment — for better visual coverage and spatial resolution. The cell arrangement algorithm proposed in this paper guarantees that the object is captured by at least $\lfloor N_{\mathtt{cam}}/3 \rfloor$ cameras, but is not optimized for measures related to visual coverage. For example, we have proposed a quantitative measure for Req. 2 in 16). We can design a similar measure to estimate the overall quality of 3D videos that will be captured so that camera control rules can be optimized for it. Also, all the cameras in one group are controlled with a single rule. However, geometric properties of the cameras are not uniform. Thus we predict that controlling each camera separately could improve the efficiency of camera usage. For example, view frustums can sometimes have enough length in the direction parallel to its optical axis, so that it may contain more than two cells. In such a case, switching its view could be omitted between those cells, leading to better viewpoint usage.

Another remaining problem is simultaneous 3D video capture of two or more objects. In this case, occlusions between the captured objects should be handled algorithmically.

Acknowledgments This research was supported by the "Foundation of Technology Supporting the Creation of Digital Media Contents" project (CREST, JST). We would like to thank Dr. Tony Tung for his valuable writing corrections.

References

1) Cignoni, P., Rocchini, C. and Scopigno, R.: Metro: measuring error on simplified surfaces, *Computer Graphics Forum*, Vol.17(2), pp.167–174 (1998).
2) Esteban, C. and Schmitt, F.: Silhouette and stereo fusion for 3D object modeling, *Computer Vision and Image Understanding*, Vol.96, No.3, pp.367–392 (2004).
3) Furukawa, Y. and Ponce, J.: Carved visual hulls for image-based modeling, *European Conference on Computer Vision (ECCV)*, pp.564–577 (2006).
4) Hartley, R. and Zisserman, A.: *Multiple View Geometry in Computer Vision Second Edition*, Cambridge University Press (2004).
5) Kanade, T., Rander, P. and Narayanan, P.: Virtualized reality: Constructing virtual worlds from real scenes, *IEEE Multimedia*, Vol.4, No.1, pp.33–47 (1997).
6) Kondou, J., Wu, X. and Matsuyama, T.: Calibration of Partially-Fixed Viewpoint Active Camera, *IPSJ SIG Notes. CVIM (in Japanese)*, Vol.2003(36), No.137-19, pp.149–156 (20030327).
7) Kutulakos, K.N. and Seitz, S.M.: A Theory of Shape by Space Carving, *International Journal of Computer Vision*, Vol.38, No.3, pp.199–218 (2000).
8) Laurentini, A.: The visual hull concept for silhouette based image understanding, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.16, No.2, pp.150–162 (1994).
9) 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, *Computer Vision and Image Understanding*, Vol.96, No.3, pp.393–434 (2004).
10) Matsuyama, T.: Cooperative Distributed Vision – Dynamic Integration of Visual Perception, Action, and Communication –, *Proc. of Image Understanding Workshop*, Monterey CA, pp.365–384 (1998). (invited paper).
11) Seitz, S.M., Curless, B., Diebel, J., Scharstein, D. and Szeliski, R.S.: Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms, *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (2006).

12) Starck, J. and Hilton, A.: Virtual view synthesis of people from multiple view video sequences, *Graphical Models*, pp.600–620 (2005).
13) Starck, J., Maki, A., Nobuhara, S., Hilton, A. and Matsuyama, T.: The Multiple-Camera 3D Production Studio, *IEEE Transactions on Circuits and System for Video Technology*, Vol.19, No.6, pp.856–869 (2009).
14) Svoboda, T., Martinec, D. and Pajdla, T.: A convenient multicamera self-calibration for virtual environments, *Presence: Teleoperators and Virtual Environments*, Vol.14, No.4, pp.407–422 (2005).
15) Tung, T., Nobuhara, S. and Matsuyama, T.: Simultaneous super-resolution and 3D video using graph-cuts, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2008).
16) Yoshimoto, H., Yamaguchi, T. and Matsuyama, T.: A Cell-Based 3D Video Capturing Method with Active Cameras, *The IEICE Transactions on Information and Systems (Japanese Edition)*, Vol.J92-D, No.9, pp.1579–1590 (2009).
17) Zhang, Z.: A flexible new technique for camera calibration, *IEEE transactions on pattern analysis and machine intelligence*, Vol.22, No.11, pp.1330–1334 (2000).

(Received December 25, 2009)
(Accepted August 4, 2010)

Tatsuhisa Yamaguchi received his B.Sc. in Engineering and his M.Sc. in Informatics from Kyoto University, Japan, in 2005 and 2007 respectively. Since 2007, he has been a Ph.D. student in the Department of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University. His research interests include computer vision theory, algorithms and its applications. He is a member of RSJ.

Hiromasa Yoshimoto received his B.S. and M.S. degrees from the Graduate School of Information Science and Electrical Engineering, Kyushu University in 2000 and 2002 respectively. He has been a visiting researcher at Kyoto University since 2005 and with Enegate Co., Ltd. since 2009. His research interests include computer vision theory, algorithms and its applications. He is a member of IPSJ and IEICE.

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

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 the Pattern Recognition Journal.
He was awarded Fellowships from the International Association for Pattern Recognition, the Information Processing Society of Japan, and the Institute of Electronics, Information, and Communication Engineers Japan.

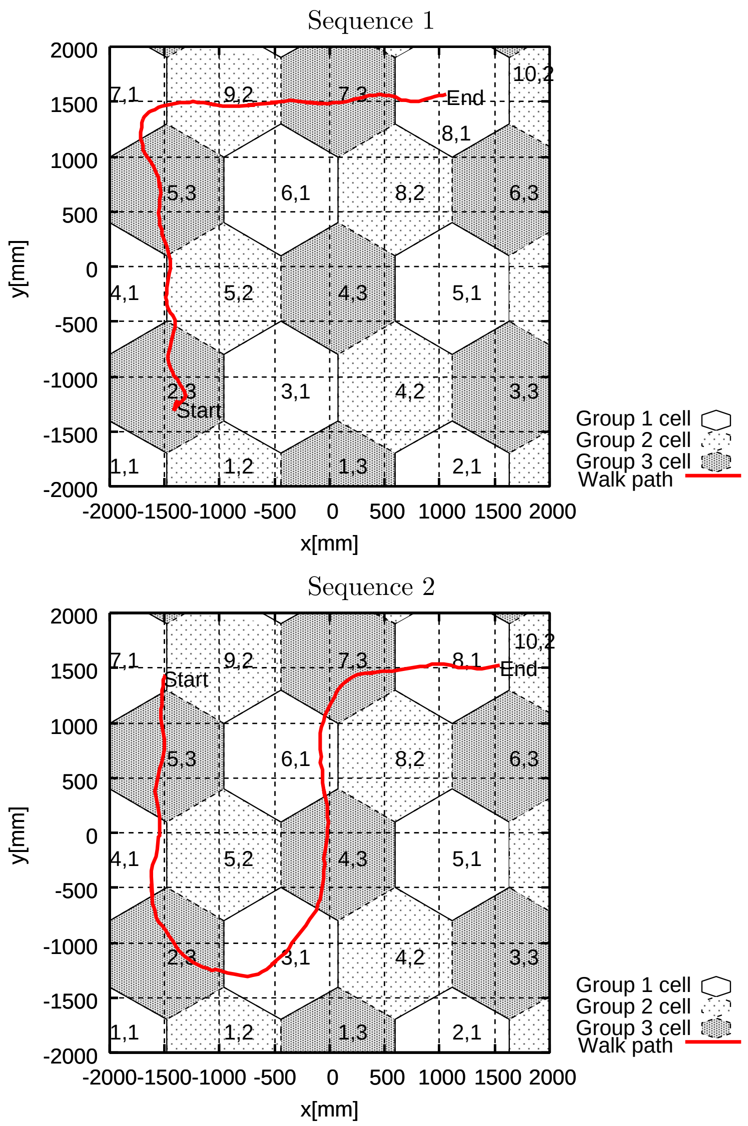


Fig. 8 The object paths used for the simulation and the cell arrangement. Each hexagon represents a cell and the numbers near the centers of them represents cell and group index.

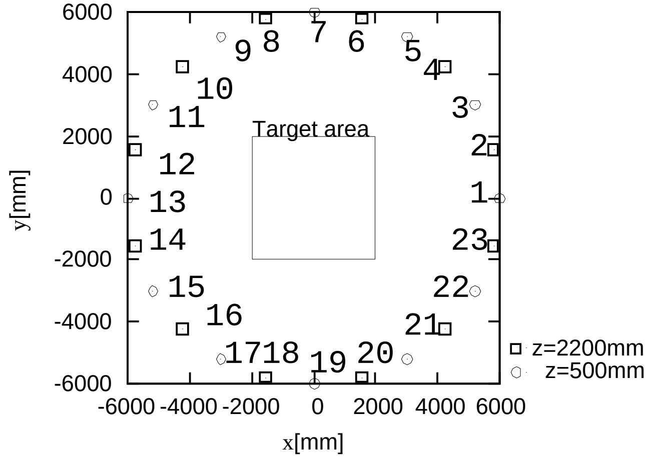


Fig. 9 Camera configuration and target area for the simulation. The squares and circles represent the camera positions. The squares represent higher cameras and the circles represent lower cameras.

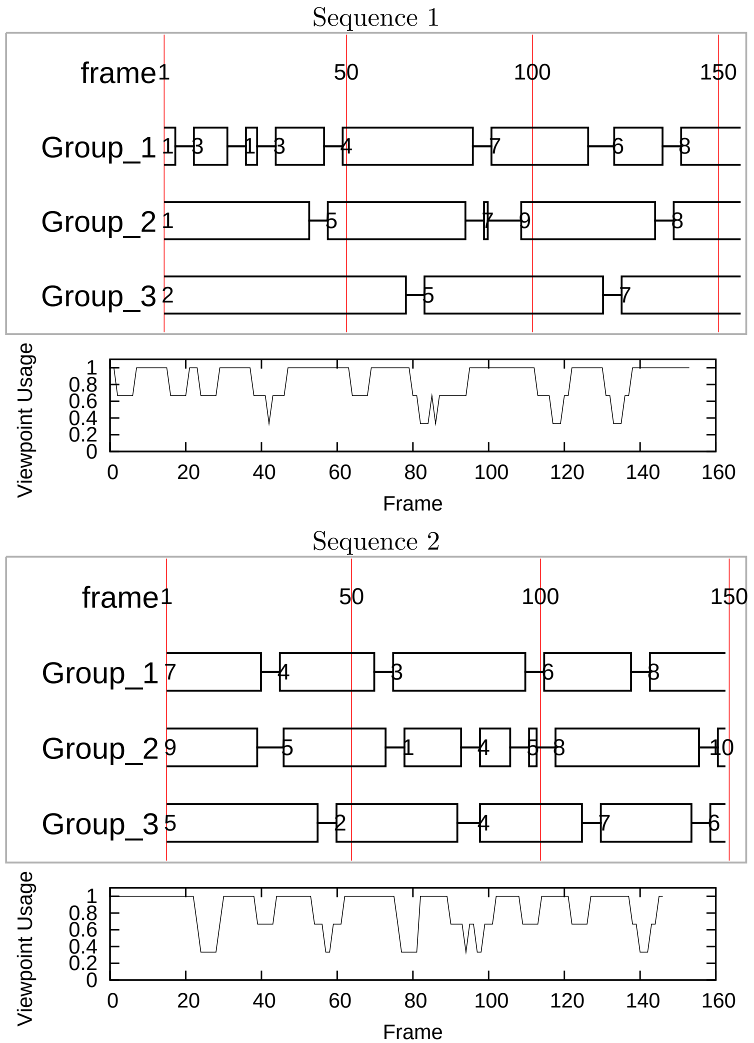


Fig. 10 The timing chart of the active camera control for each sequence. The boxes represent the interval when the cameras in each group watched a cell. Gaps indicate that the cameras were in motion. Numbers in boxes represent the cell index j that the cameras in each group watched.

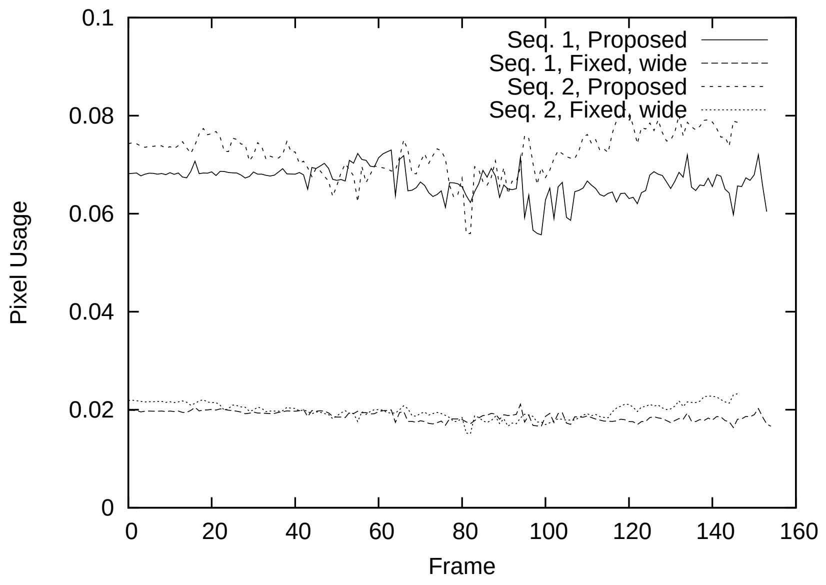


Fig. 11 Pixel usage.

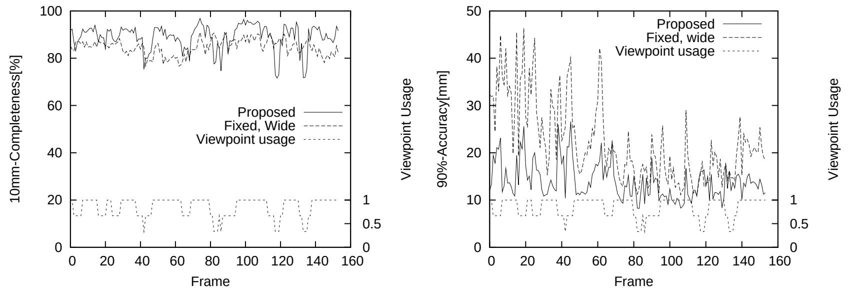


Fig. 12 Completeness and accuracy for sequence 1.

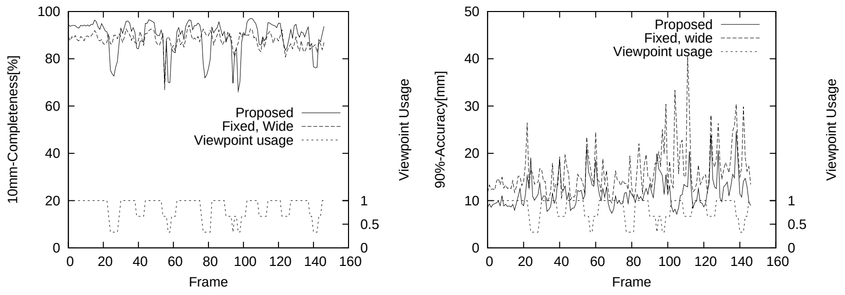


Fig. 13 Completeness and accuracy for sequence 2.

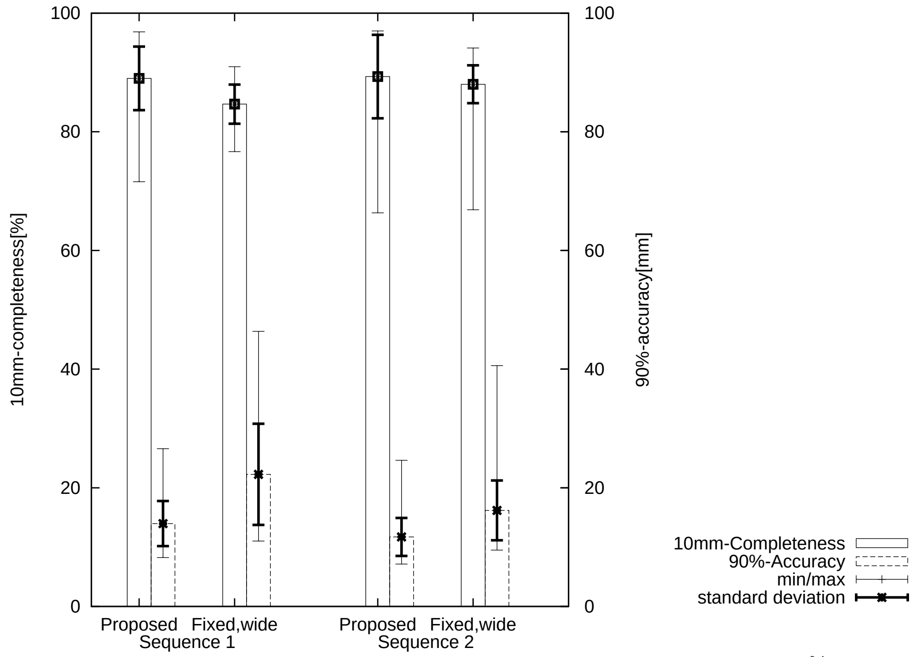
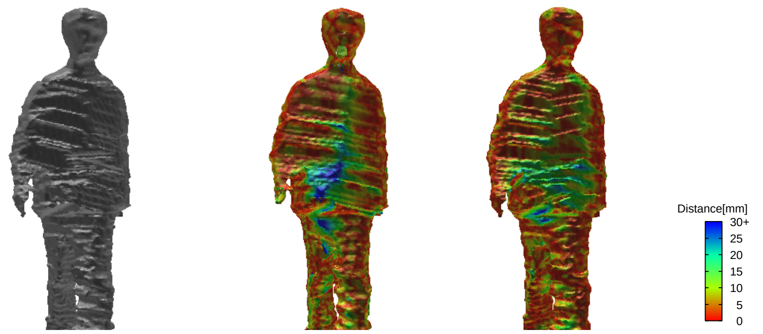
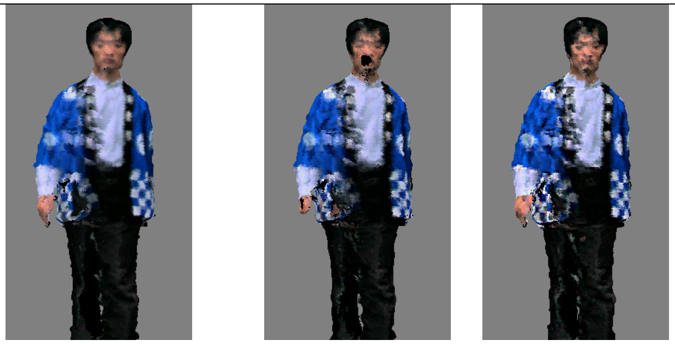


Fig. 14 Comparison of the two methods in terms of 10mm-completeness and 90%-accuracy.



	Ground truth	Proposed 8 cameras	Fixed, wide 24 cameras
90%-accuracy[mm]		12.1	14.7
10mm-completeness[%]		71.9	85.9



PSNR[dB]		22.8	21.3

Fig. 15 Shape reconstruction result in frame 135 of sequence 1. The first row shows the shapes, whose colors indicate the distance to the ground truth surface. Distances are computed using Metro[1]. The bottom row shows the rendered images and their PSNR in comparison to the ground truth.

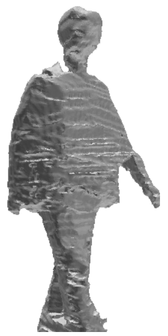
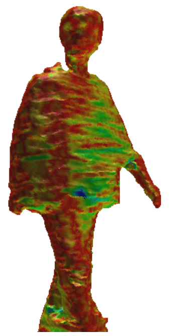
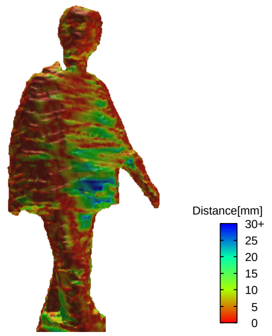
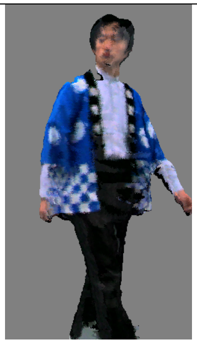
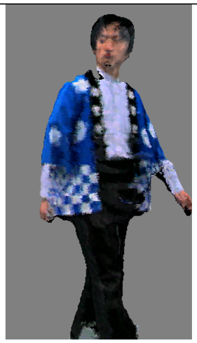
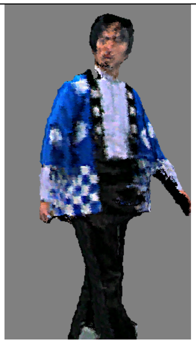
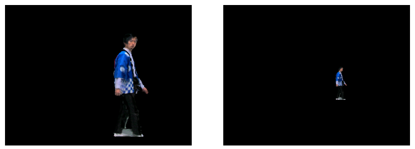
		
Ground truth	Proposed 16 cameras	Fixed, wide 24 cameras
90%-accuracy[mm]	9.9	11.5
10mm-completeness[%]	92.1	87.9
		
PSNR[dB]	23.8	20.4

Fig. 16 Shape reconstruction result in frame 95 of sequence 1.



Proposed Fixed, wide

Fig. 17 Images obtained by camera 1 at frame 95 in sequence 1.

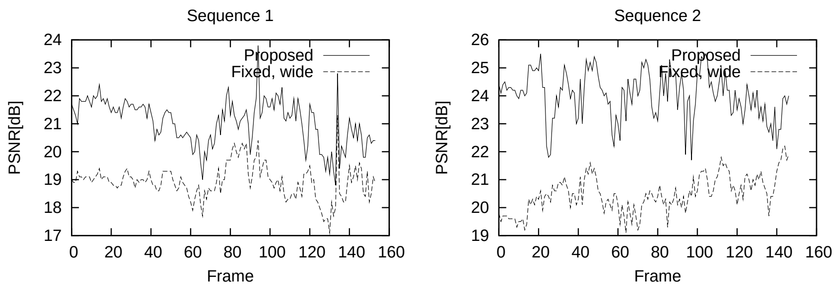


Fig. 18 PSNR of rendered images.

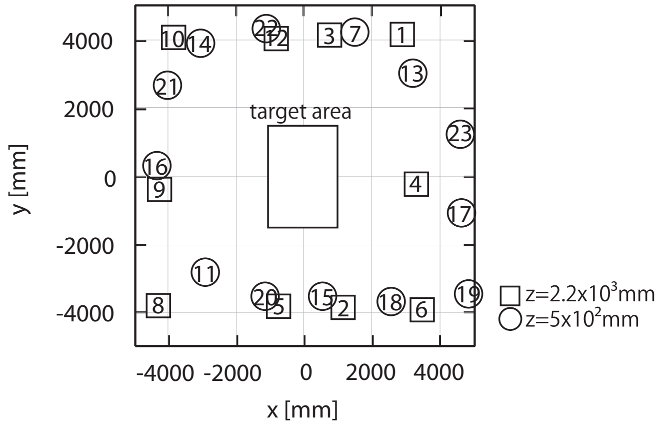


Fig. 19 Our studio, camera setup and target area. The numbers in circles or squares represent the camera positions. The cameras represented by the squares were mounted on the stations hung from the ceiling. The others were mounted on the stations on the floor.

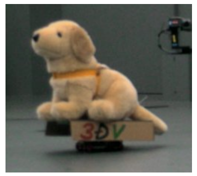


Fig. 20 The object captured in the real studio experiment: a stuffed toy on a radio control car.

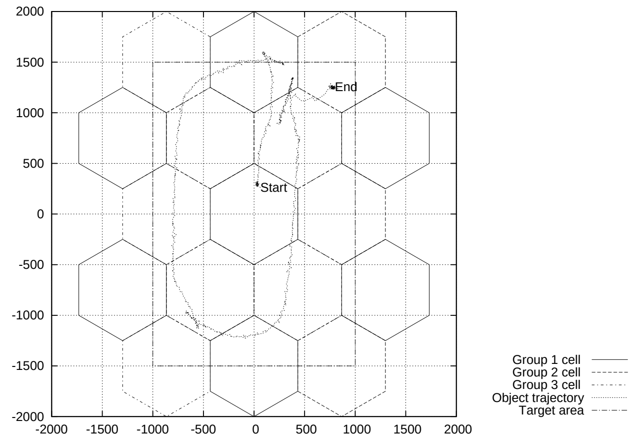


Fig. 21 Generated cells and object trajectory in the studio experiment.

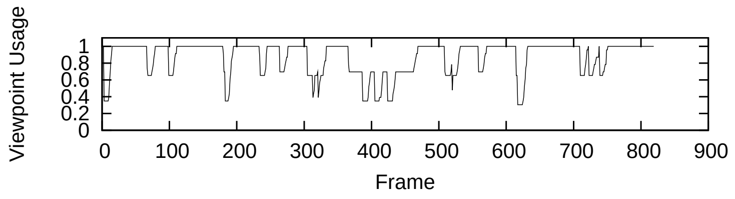


Fig. 22 Viewpoint usage in the studio experiment.

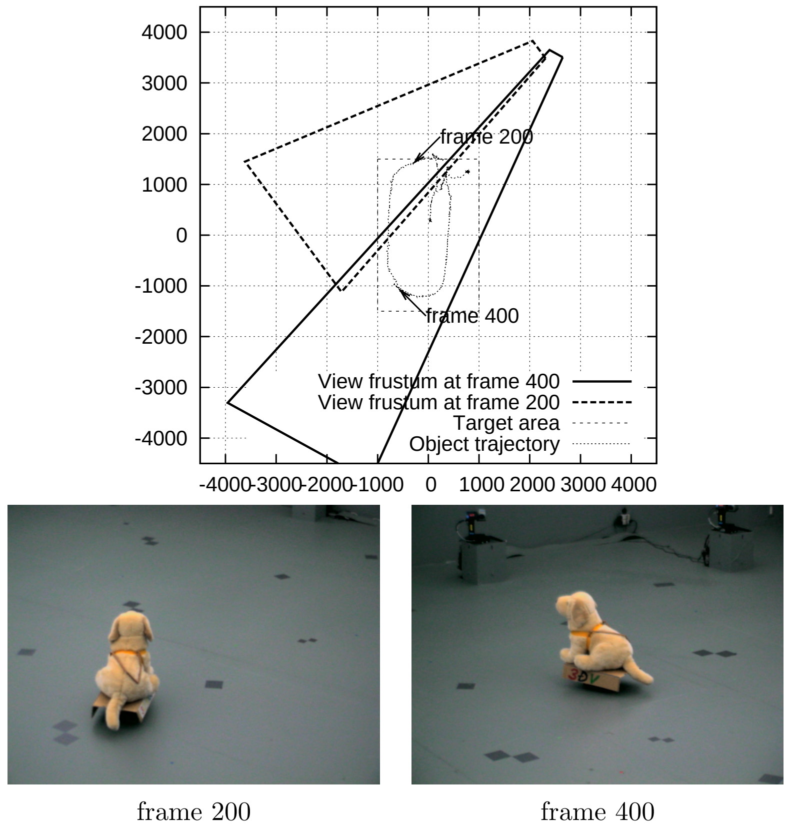


Fig. 23 Captured images by camera 1, object positions and camera view frustums at frames 200 and 400.

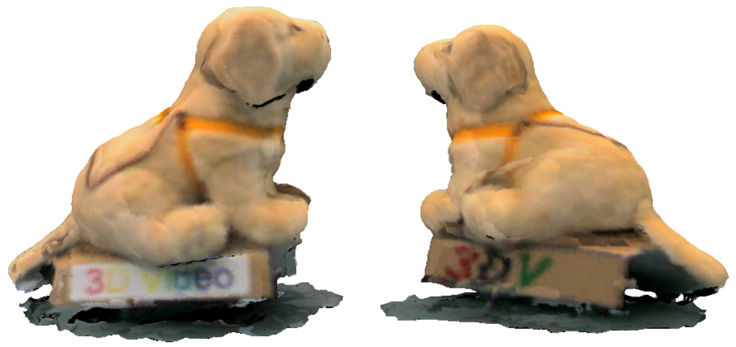


Fig. 24 Reconstruction result at frame 395 with 8 cameras.