九州大学学術情報リポジトリ
Kyushu University Institutional Repository

Frame Rate Stabilization for Real-Time Free-viewpoint Video Generation : 3D Shape Reconstruction in Multiple Resolution

Nabeshima, Rui
Department of Intelligent Systems, Kyushu University

Ueda, Megumu
Department of Intelligent Systems, Kyushu University

Arita, Daisaku
Department of Intelligent Systems, Kyushu University

Taniguchi, Rin-ichiro
Department of Intelligent Systems, Kyushu University

https://hdl.handle.net/2324/5934

出版情報：Proceedings of the 2nd Joint Workshop on Machine Perception and Robotics, 2006-11
バージョン：
権利関係：

Frame Rate Stabilization for Real-Time Free-viewpoint Video Generation -3D Shape Reconstruction in Multiple Resolution-

Rui Nabeshima Megumu Ueda * Daisaku Arita † Rin-ichiro Taniguchi

Department of Intelligent Systems
Kyushu University
744 Motooka Nishi-ku Fukuoka 819-0395 JAPAN

Abstract

This paper represents a method to stabilize the frame rate of real-time free-viewpoint video generation. When we apply real-time free-viewpoint video to live video broadcasting, the frame rate of video processing should be constant. Otherwise, presented video gives viewers unnatural feeling due to time inconsistency. Our method to generate free-viewpoint video is based on 3-D shape reconstruction and visualization of the 3-D shape by CG technique, whose processing time mainly depends on the number of triangles of 3-D shape reconstructed. Therefore, processing time of the video generation, i.e., the frame rate, is not constant. To solve this problem, we propose a new method which flexibly varies the space resolution of the 3D-shape reconstruction stage, and which makes the processing time of free-viewpoint video nearly constant. We have implemented our method on a PC cluster and realized real-time processing of the free-viewpoint video generation, whose experimental results show the effectiveness of our method.

1 Introduction

Currently, radio, television, etc. are used as methods of telepresence. However, media using 3D information are more intuitive than those using 1D or 2D information like them because the world where people live is 3D space. Several researches have been done for generating the free-viewpoint video which shows objects in the real world from an arbitrary viewpoint using multiple cameras since Kanade et al. [1] had proposed the concept of "Virtualized Reality" [2][3]. However, they cannot generate the free-viewpoint video in real-time.

We are researching on-line generation of free-viewpoint video[4], whose process consists of three stages:

*Currently, Ricoh Company, Ltd.
†Currently, Institute of Systems & Information Technologies/KYUSHU

1. reconstructing 3D shapes in a voxel form by the visual cone intersection method[5],
2. converting the voxel form of 3D shapes into the triangular patch form of them, and
3. coloring the triangular patches.

In the first stage, we reconstruct 3D shapes of objects explicitly. Matusik et al.[6] have proposed the image-based visual hull technique, which can generate a free-viewpoint video in real-time. This method generates a virtual viewpoint video by projecting rays from the viewpoint on image planes of cameras. This means that the method does not reconstruct 3D shapes explicitly and the amount of computation is smaller than explicit reconstruction. However, since the method directly generates a virtual viewpoint video, the amount of computation is increased relatively to the number of virtual viewpoints. It is impossible for the method to deliver free-viewpoint videos to multiple viewers, who can control individually their own virtual viewpoint. On the other hand, our method can broadcast the triangular patch form of objects with color information, or a CG data, to all viewers and free-viewpoint videos can be generated on their own terminals. This means that the amount of computation does not depend on the number of viewers.

However, there is a problem in our method. When the surface area of objects becomes larger, the frame rate becomes lower since processing time of the second stage and that of third stage depends on the number of triangular patches. Stability of the frame rate is very important for on-line distribution of free-viewpoint videos since large jitters of on-line distribution requires longer queues for data transmission.

In this paper, we propose a new method which flexibly varies the space resolution of the 3D-shape reconstruction stage, and which makes the processing time of free-viewpoint video nearly constant. This is realized by using an octree-based visual cone intersection method[7] and raising its resolution step by step until the time allowed for one frame is over.

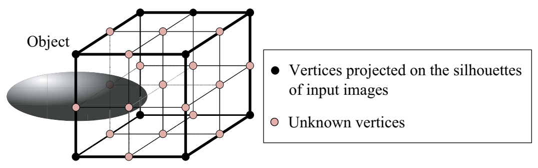


Figure 1: Failed Voxel (Case Example)

2 3D Space Representation Using Octree

We use an octree-based visual cone intersection method proposed by Sato et al[7] to increase the space resolution step by step.

2.1 Pass Algorithm

An *octree* is a tree to index three dimensions. Each node has either eight children or no child. This means that one voxel is divided into eight voxels when the space resolution is increased.

Each voxel is classified into three types as follows.

White The eight vertices are projected on the silhouettes

Black The eight vertices are projected on the background

Gray The other voxels (borders of silhouettes)

Voxels of a pre-defined size called initial voxels are classified into the above three voxel types and only gray voxels are subdivided. Such classification and subdivision procedure is recursively executed until the finest space resolution. This subdivision procedure from the initial space resolution to the finest one is named *pass algorithm*.

2.2 Multi-Pass Subdivision Algorithm

Since the occupancy test described above is very simple, it often ends in failure. Fig. 1 shows a typical example. In this example, a voxel is projected on a narrow tip of a silhouette that avoids its vertices. Though, this voxel in classified a black, but the voxel must be classified as gray. This voxel is named *failed voxel*. Then, a Multi-Pass Subdivision Algorithm[7] is proposed to solve this problem. In Step1, the pass algorithm is applied to initial voxels. Afterward, failed voxels are detected in Step2. Failed voxels are subdivided and the pass algorithm is applied to the voxels obtained in Step3. Failed voxels are detected once again in Step2. As long as failed voxels are detected, Step2 and Step3 are repeated. If no failed voxels are detected, the process ends.

The way to detect failed voxels is as follows. We connect neighboring gray voxels in the finest space resolution, and thus we get the surface of a reconstructed geometry. If there is a hole on the reconstructed geometry, there must be failed voxels near the hole.

3 Visual Cone Intersection in Multi-Resolution

We apply the Multi-Pass Subdivision Algorithm until not the finest space resolution but lower resolution (We call the resolution on which the process is stopped *cutoff resolution.*). When the process in the cutoff resolution is finished, the cutoff resolution is made higher if one frame period is not passed. When the cutoff resolution reaches the finest space resolution or when one frame period passes, 3-D shape reconstructed at this time is supplied to the next stage. This is how a fluctuation of processing time gets smaller and we stabilize a frame rate.

3.1 Visual Cone Construction

The process of visual cone construction in multi-resolution is shown in Fig. 2. The procedure from Step3 to Step8 is recursively executed until the cutoff resolution reaches the finest space resolution. When one frame period passes, the process stops at either the middle of Step4 or that of Step5 and ,then, we perform Step6 and Step7.

Step1 Apply the multi-pass subdivision algorithm until the cutoff resolution

Step2 Output a processing result

Step3 Make the cutoff resolution higher

Step4 Subdivide failed voxels which can be subdivided (Voxel of *A* in Fig. 2)

Step5 Subdivide gray voxels which can be subdivided (Voxel of *B* in Fig. 2)

Step6 Detect failed voxels which newly arise (Voxel of *C* in Fig. 2) and subdivide them

Step7 Output a processing result obtained from Step3 to Step6

Step8 Return Step3

The output in Step7 is only the difference from the previous output. The difference involves child nodes of failed voxels and gray voxels in Step4 and Step5 and subtrees whose root is a failed voxel in Step6. Sent data does not have child nodes of black and white voxels. This makes the amount of data transfer between the processes smaller.

In Step1 or Step6, when there is a failed voxel which cannot be divided in the current cutoff resolution, they are

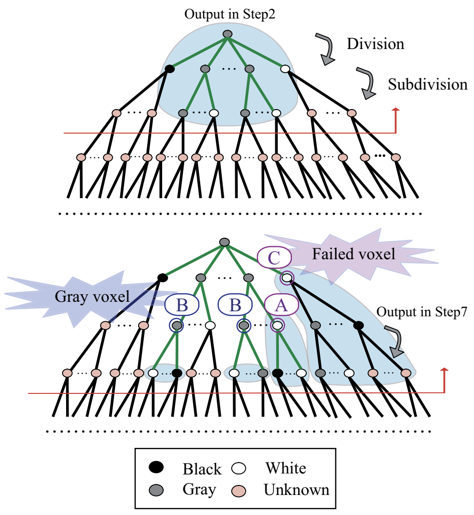


Figure 2: A Flow of Visual Cone Construction

retained and subdivided in the Step4 of the next iteration, in which the cutoff resolution is higher.

3.2 Visual Cone Intersection in Multi-Resolution

We perform visual cone intersection at multistage processing. We divide cameras into groups and a visual cone is reconstructed in each group. Afterwards, we get final 3-D shape by applying intersection operation to the visual cones. This is why one PC cannot get all camera images due to restriction of the data input capability of PC. In addition, this has an advantage that we can increase the number of cameras if we increase the number of PCs.

However, a space resolution of visual cones which are obtained on several PC may be different due to a difference of cutoff resolution. To apply intersection operation to such visual cones, we require a visual cone intersection in multi-resolution. Furthermore, since a visual cone is sent whenever cutoff resolution becomes higher, intersection operation starts as soon as the first visual cone is received. Afterwards, when a new visual cone is received, we apply intersection operation to only required nodes.

First, we explain a process of visual cone intersection using the output of Step2. We scan all octrees which are output in each PC from a root by breadth first search. Since a node of the same position in octrees corresponds to a voxel of the same position in voxel space, we perform visual cone intersection by applying intersection operation to the node. If there is at least one black voxel in nodes of the same position, we judge the node as black. If there is no black voxel and there is at least one failed voxel, we dig into the octrees. If all nodes of the same position in octrees are judged as white, we judge the node as white. Otherwise, we judge

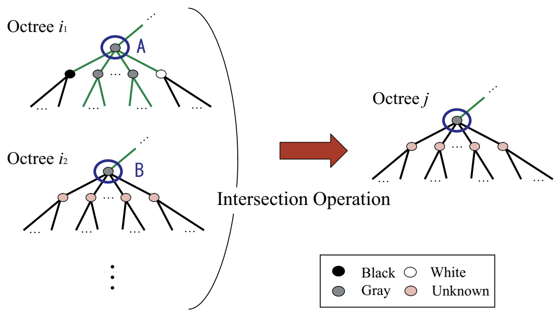


Figure 3: Intersection Operation

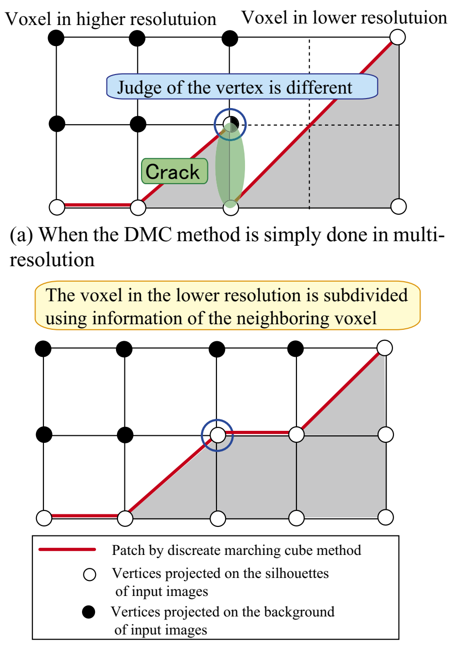
the node as gray and dig into the octrees. Furthermore, we get information of each node's (voxel's) eight vertices by bit logical-and operation simultaneously.

Secondly, we explain a process of visual cone intersection using the output of Step7. We judge types of nodes which are obtained in Step4 and Step5 by the same method as above. However, there is a node judged as gray voxel which is subdivided in a octree (A in Fig.3) and which is not subdivided in another octree (B in Fig.3) by stopping the process. In this case, we should dig into the octree and apply intersection operation to child nodes of the node by subdividing the node (B in Fig.3). However, we do not dig into the octree any more not to increase the processing time. This also means that we clear up information of the node which is subdivided (A in Fig.3) and nodes of the descendant. We operate intersection operation from a node judged as failed voxel in Step6 and dig into the octree. However, if a parent or ancestor of the node was judged as black in the octree, we clear up information about this subtree. Because, if there is a node of parent or ancestor which is judged as black, nodes of children or descendant must be black in the nature of visual cone intersection.

4 Patch Configuration in Multi-Resolution

We use a Discrete Marching Cubes(DMC) method[8] to convert a voxel form of 3D shapes to a triangular patch form. However, if we simply apply the DMC method to voxels in multi-resolution, cracks between these patches happens. Cracks make free viewpoint video unnatural. Therefore, we need a process which eliminates cracks.

We explain cracks in Fig.4 which is simplified to 2-D. In Fig.4(a), a crack is generated owing to different judgment between a left voxel in higher space resolution and a right voxel in lower space resolution in a middle vertex which a mark attached. That is, a crack is generated when a voxel in higher space resolution and a voxel in lower space resolution neighbor and judgment of a shared vertex is different.


Voxel in higher resolutuion
Voxel in lower resolutuion
Judge of the vertex is different
Crack
The voxel in the lower resolution is subdivided using information of the neighboring voxel
Patch by discreate marching cube method
Vertices projected on the silhouettes of input images
Vertices projected on the background of input images

(a) When the DMC method is simply done in multi-resolution

(b) Proposal technique

Figure 4: Patch Configuration in Multi-Resolution

We detect such a vertex and subdivide voxel in lower space resolution (Fig.4(b)). Judgment of this vertex depends on the voxel in higher resolution. Judgment of the other additional vertices is performed by not a camera image but information about vertices of the voxel in lower resolution. Specifically, if sets of vertices in the both sides of a vertex are at least black and black, the vertex is black. Otherwise the vertex is white. In this way, we can judge vertex quickly by easy logical operations.

5 System Configuration

We obtain visual cones from each three cameras and make their intersection. When the process proceeds, the surface area diminishes and the number of gray voxels decreases. It increases the number of voxels which do not have to be divided, and thus the processing time decreases.

The system configuration which we propose is as follows. Processes are distributed to PCs shown in Fig. 5 and executed in pipeline parallel. Node-D and Node-E are the same as conventional system[4].

Node-A: Each node-A extracts object silhouettes from video frames captured by a camera by background subtraction and noise reduction, and sends the silhouette image to a node-B and a node-D. Each node-A sends a binary silhouette image to a node-B to reduce the amount of data since each node-B uses the image to reconstruct a visual cone. On the other hand, each node-A sends an RGB silhouette image to a node-D since each node-D uses the image to color a visual hull.

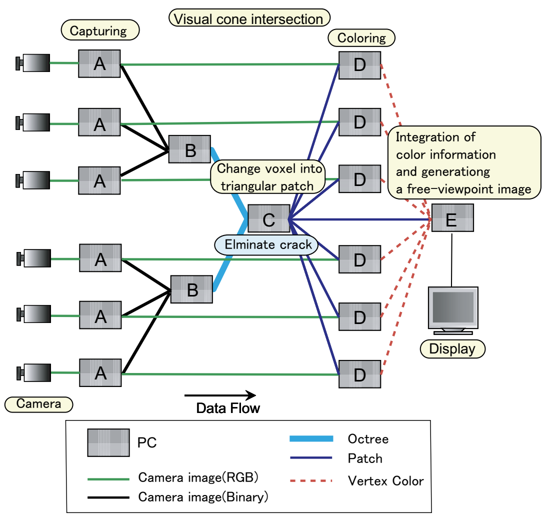


Figure 5: System Configuration

Node-B: Each node-B constructs a visual hull. To classify voxels, node-B projects vertices on to the silhouette images from three viewpoints. Each vertex is regarded as occupied only when all projected point on each image are occupied. Each node-B sends the node-C a visual hull as octree. Each node-B sends visual hull and information about the cutoff resolution. When each node-B stops the process, each node-B tells a node-C that the visual hull and information which are sent in previous time is the final result.

Node-C: A node-C makes intersections of the visual hulls whenever it receives one from node-B. Furthermore, the node-C converts only gray voxels into triangular patches by the DMC method. Since a space resolution of these gray voxels may be different, the node-C perform crack patching before DMC method. Then, the node-C sends the voxel space and its corresponding marching cube patterns to node-D and node-E since the data size of both voxel space and its marching cube patterns is smaller than that of all triangular patches.

Node-D: First each node-D transforms the shape model represented in a voxel space into triangular patches by using the marching cube patterns sent from the node-C. Then, each node-D colors visible vertexes of the shape model based on one camera image. Finally, each node-D sends color information of all the vertexes of the shape model to the node-E.

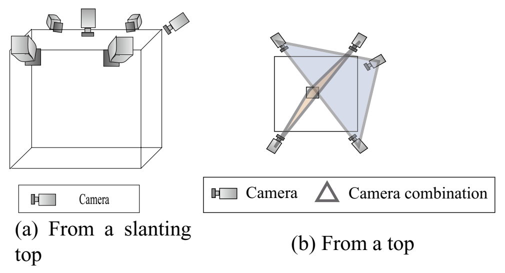


Figure 6: Camera Arrangement and Combination

Node-E: First, the node-E receives the position of the virtual viewpoint directed by the user. Then the shape model is transformed into triangular patches in the same way as a node-D. Finally, the node-E integrates color information of all cameras and generates an image from the directed viewpoint.

6 Experiments

Using the system we proposed, we generate a free-viewpoint video in real-time to evaluate the processing time and the number of patches.

6.1 Experimental Environments

We have used nine IEEE-based cameras with 640×480 pixel resolution, and 16 PCs (six node-As, two node-Bs, one node-C, six node-Ds, one node-E), each of which has an Intel Pentium4 (3GHz), 1GB memory and NVidia GeForce FX. PCs are connected with one other by Myrinet, a giga-bit network. All the cameras are calibrated in advance by Tsai's method [9]. The camera arrangement and combination of visual cone intersection in node-Bs are shown in Fig. 6. The maximal space resolution is $128 \times 128 \times 128$ and the size of a minimum voxel is $2cm$. The depth of the octree is five and the space resolution of initial voxels is $8 \times 8 \times 8$. The cutoff resolution is two level, $64 \times 64 \times 64$ and $128 \times 128 \times 128$. That is to say, we apply multi-pass algorithm until the depth of the octree is four and space resolution is $64 \times 64 \times 64$, and we advance to $128 \times 128 \times 128$.

Using the system, we generate a free-viewpoint video in which two persons go in or go out from the measurement. Change of the number of persons causes change of the amount of processing.

6.2 Generated Free-viewpoint Video

Fig. 7 shows images generated in fixed resolution and multi-resolution. The space resolution of the fixed resolution system is $64 \times 64 \times 64$ and $128 \times 128 \times 128$. When there is one person, there is enough time to reach the space resolution with $128 \times 128 \times 128$. When there are two persons, the process is stopped at the space resolution with $64 \times 64 \times 64$. Therefore, a quality of the image in multi-resolution is near the one in fixed -resolution with $128 \times 128 \times 128$ when there is one person and is near the one in fixed -resolution with $64 \times 64 \times 64$ when there are two persons.

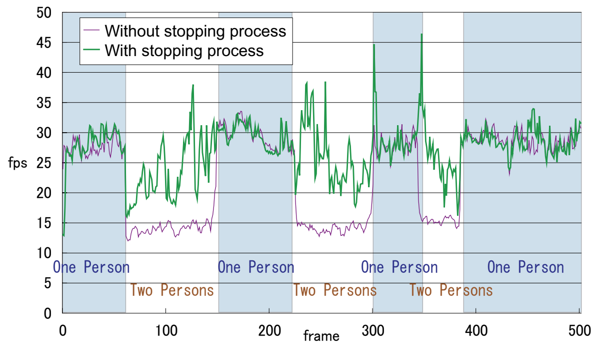


Figure 8: Frame Rate

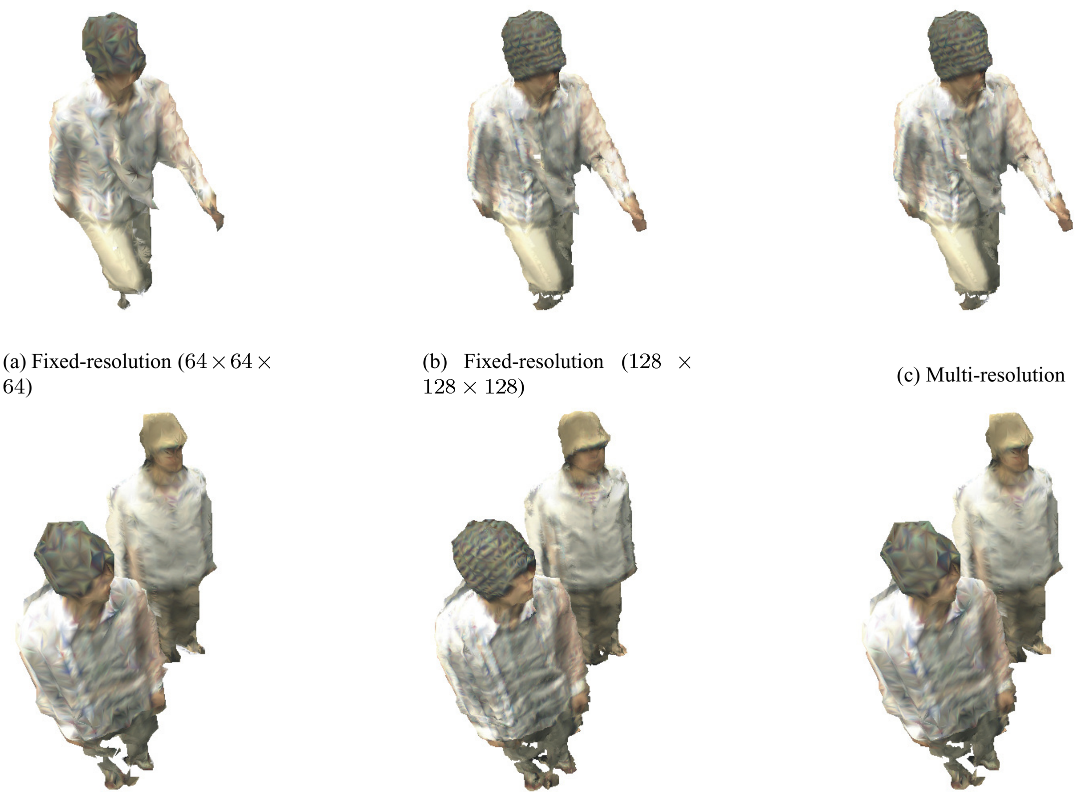
6.3 Processing Time

Not only the average of frame rate but also the variance is important for on-line free-viewpoint video distribution.

Fig. 8 shows the frame rate in case the number of persons changes without stopping the process and with stopping the process. When the number of people changes from one to two, the frame rate without stopping the process decreases significantly. On the other hand, the decrease of the frame rate with stopping the process is much smaller. However, even the frame rate with stopping the process decreases.

6.4 Number of Patches

Fig. 9 shows the number of patches without stopping the process and with stopping the process. In the latter case, we measure also the number of patches with crack patching and without crack patching. When there is one person in any cases, the space resolution becomes mostly $128 \times 128 \times 128$. Therefore, the crack patching is not required in any cases, and the number of patches of three graphs is almost equal. The number of the patches when there are two persons is increasing about double compared with ones when there is one person. Although the number of patches with stopping process is increasing, the increase is much smaller than the previous case. However, even the number of patches with stopping process is increasing. Because, when there is one person, a surface is small and the process reaches the finest space resolution and stopping the process does not happen. Thus, if we make the finest space resolution higher, stop-



(a) Fixed-resolution ($64 \times 64 \times 64$)

(b) Fixed-resolution ($128 \times 128 \times 128$)

(c) Multi-resolution

(d) Fixed-resolution ($64 \times 64 \times 64$)

(e) Fixed-resolution ($128 \times 128 \times 128$)

(f) Multi-resolution

Figure 7: Generated Image

ping process happens when there is one person, and the number of patches does not increase when the number of persons changes. This increase of the number of patches makes the frame rate decreasing when there are two persons.

When there are two persons and the stopping process happens, the number of patches is increasing due to the crack patching since there are voxels in a different space resolution. If the number of patches is increasing considerably, the frame rate is not stable. However, this increase due to the crack patching is small. Thus, the crack patching which we propose is quite useful.

7 Conclusion

In this paper, a new method which flexibly varies the space resolution of the 3D-shape reconstruction stage, and which makes the processing time of free-viewpoint video nearly constant. We make some experiments to show the frame rate stabilization independent of the object. Major future works are as follows:

- stabilizing the number of patches,
- stopping processes in a coloring stage,
- developing an on-line free-viewpoint video distribution system.

References

[1] T. Kanade, P. W. Rander, P. J. Narayanan:“ Concepts and early results”, IEEE Workshop on the Representation of Visual Scenes, pp.69–76, Jane 1995.

[2] Shohei Nobuhara, Takashi Matsuyama:“ Heterogeneous Deformation Model for 3D Shape and Motion Recovery from Multi-Viewpoint Images”, Proceedings of the 2nd International Symposium on 3D Data Processing, Visualization, and Transmission, pp. 566–573, Thessaloniki Greece, September 2004.

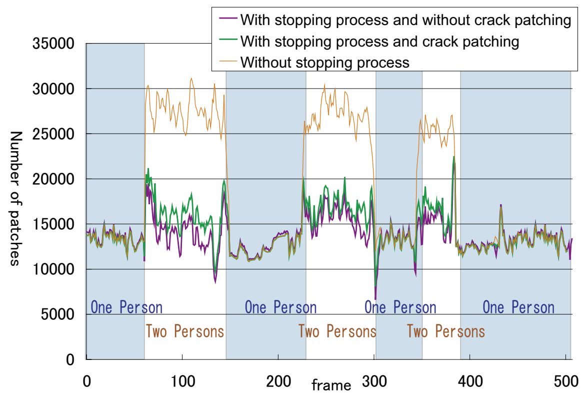


Figure 9: Number of Triangle Patches

[3] J. Carranza, C. Theobalt, M. A. Magnor, H.-P. Seidel:“ Freeviewpoint video of human actors”, in ACM Trans. on Graphics, vol. 22, no. 3, pp.569–577, July 2003.

[4] Megumu Ueda, Rui Nabeshima, Daisaku Arita, Rin-ichiro Taniguchi:“ Real-time Free-viewpoint Video Based on 3D Shape Reconstruction”, Proc. of the 1st Joint Workshop on Machine Perception and Robotics September 2005.

[5] W. N. Martin, J. K. Aggarwal:“ Volumetric description of objects from multiple views” IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 5, No. 2, pp.150–158, 1983.

[6] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan:“ Image-Based Visual Hulls”, Proc. of SIGGRAPH, pp.369–374, 2000.

[7] Hidenori Sato, Hiroto Matsuoka, Akira Onozawa, Hitoshi Kitazawa:“ Image-Based Photorealistic 3D Reconstruction Using Hexagonal Representation”, IPSJ Journal, Vol. 46, No. 2, pp.639–648, 2005.

[8] Yukiko Kenmochi, Kazunori Kotani, and Atsushi Imiya:“ Marching cubes method with connectivity”, in Proc. on International Conference on Image Processing, vol. 4, pp.361–365, Oct 1999.

[9] R. Y. Tsai:“ A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses”, in IEEE Trans. on Robotics and Automation, vol. 3, no. 4, pp.323–344, 1987.