2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Fast 3D Reconstruction of Human Shape and Motion Tracking by Parallel Fast Level Set Method

Yumi Iwashita*[1], Ryo Kurazume*[1], Kenji Hara*[2], Seiichi Uchida*[1], Ken'ichi Morooka*[3], Tsutomu Hasegawa*[1]
*[1]Graduate School of Information Science and Electrical Engineering
*[2]Graduate School of Design *[3]Digital Medicine Initiative
Kyushu University, Fukuoka, JAPAN
yumi@is.kyushu-u.ac.jp

Abstract— **This paper presents a parallel algorithm of the Level Set Method named the Parallel Fast Level Set Method, and its application for real-time 3D reconstruction of human shape and motion. The Fast Level Set Method is an efficient implementation algorithm of the Level Set Method and has been applied to several applications such as object tracking in video images and 3D shape reconstruction using multiple stereo cameras. In this paper, we implement the Fast Level Set Method on a PC cluster and develop a real-time motion capture system for arbitrary viewpoint image synthesis. To obtain high performance on a PC cluster, efficient load-balancing and resource allocation algorithms are crucial problems. We develop a novel optimization technique of load distribution based on the estimation of moving direction of object boundaries. In this technique, the boundary motion is estimated in the framework of the Fast Level Set Method, and the optimum load distribution is predicted and performed according to the estimated boundary motion and the current load balance. Experiments of human shape reconstruction and arbitrary viewpoint image synthesis using the proposed system are successfully carried out.**

I. INTRODUCTION

Arbitrary viewpoint image synthesis is a key technology for developing a 3D interactive television system. In this system, a viewer can select an arbitrary viewpoint of his choice and the system synthesizes the virtual image from the viewpoint. Efficient cooperative work in shared virtual space will also be realized with this system. For generating arbitrary viewpoint images of realistic objects, image-based approach [1] and model-based approach [2] [3] [4] have been proposed so far.

The image-based approach is based on the appearance of a target. In this approach, multiple cameras are placed around the target, and the arbitrary images are synthesized from a number of captured images. In general, though more realistic images can be obtained than the model-based approach, dozens of cameras have to be installed densely around the target to synthesize realistic virtual images.

On the other hand, the model-based approach is based on the reconstruction of the 3D shape of a target. In this approach, the 3D model of the target is reconstructed using silhouette or stereo images captured by multiple cameras firstly, and the arbitrary viewpoint images are synthesized by capturing rendered images of the textured 3D model. Although the 3D shape reconstruction is computationally expensive, a wide range of viewpoints can be chosen as virtual viewpoints around the target. The number of cameras required is also smaller than the image-based approach. Therefore, we adopt the model based approach in this paper.

For recovering the 3D shape of a target with multiple video cameras, the volumetric intersection technique [5] [6] and the multi-view stereo technique [2] [7] have been proposed so far. The volumetric intersection technique is based on a silhouette constraint, that is, a 2D silhouette of an object constrains the object inside a frustum produced by back-projecting the silhouette from the corresponding viewpoint [5] [6]. Franco et al. [8] reconstructed a 3D model by estimating object occupancy probabilities from silhouette information. They extended the system [8] to recover occluders in a 3D scene[9]. Since the volumetric intersection technique cannot reconstruct a complex shape such as a concave object, photometric information is additionally utilized for recovering an accurate model such as voxel carving method [10] [11]. However, these methods need large calculation costs.

Though the multi-view stereo technique can deal with a concave shape, it is also computationally expensive due to the stereo correspondence between multiple images. In addition, most of conventional systems based on these techniques are designed for a single and an isolated target. In case that there are multiple objects in the scene, it is quite difficult to reconstruct a 3D model of each object separately due to the mutual occlusion.

To overcome these problems, we have proposed a motion capture system using multiple stereo cameras and the Fast Level Set Method (FLSM) [12]. The FLSM is an efficient implementation algorithm of the Level Set Method [13] [14] and has been applied to several applications such as object tracking in video images [12]. In the proposed system, multiple stereo cameras are located around targets, and the accurate 3D models of multiple targets are reconstructed separately and robustly against mutual occlusion by integrating stereo range data using the FLSM.

In this paper, we implement the proposed system on a PC cluster and develop a real-time motion capture system for arbitrary viewpoint image synthesis. To obtain high performance on a PC cluster, efficient load-balancing and resource allocation algorithms are crucial problems. We develop a novel optimization technique of load distribution based on the estimation of moving direction of object boundaries. In this technique, the boundary motion is estimated in the framework of the Fast Level Set Method, and the optimum load distribution is predicted and performed according to the estimated boundary motion and the current load balance. Experiments of human shape reconstruction and arbitrary viewpoint image synthesis using the proposed system are successfully carried out.

II. Fast Level Set Method

The Level Set Method (LSM) [13] [14], introduced by Osher and Sethian, has attracted much attention as a method that realizes a topology free active contour model. Various applications based on the LSM have been presented so far including motion tracking, 3D geometrical modeling, and simulation of crystallization. However, the calculation complexity remains an open problem. To overcome this problem, the Fast Level Set Method (FLSM) [12] was proposed as a high speed execution technique, and it was applied to real-time applications, such as 2D real-time tracking of moving objects in video images. In this section, basic ideas of the LSM and the FLSM are described.

A. Level Set Method and its high speed execution algorithm

The LSM utilizes an implicit function Φ which is defined in a space one dimensional higher than that of where a contour (surface) of interest is described. This function Φ, which is defined as a distance function from a current contour in general, is updated according to a next PDE (Partial Differential Equation).

$$\Phi_t = -F(\kappa) \mid \nabla\Phi \mid \tag{1}$$

where, κ is a local curvature of Φ, and F is a speed function. The contour to be tracked is detected as the cells with a value of zero of the implicit function (zero level set), that is, the contour line of $\Phi = 0$. In the implementation of the LSM, the space is uniformly split by cells, and Eq. (1) is solved iteratively using numerical schemes such as the upwind scheme.

To solve Eq. (1), the speed function $F(\kappa)$ has to be determined at each cell for every update process of Φ. The distribution of the $F(\kappa)$, which is known as the extension velocity field [15], is constructed as follows: i) at the current zero level set cell, F is calculated according to the intensity of the current image at first; ii) next, at each cell except the zero level set cell, the speed function F is copied from the nearest zero level set cell. However, finding the nearest zero level set cell needs large calculation cost.

To overcome these problems, several techniques have been proposed in the past, such as the Narrow Band Method [14] and the Fast Marching Method [14].

B. Fast Level Set Method

Though the Narrow Band Method is high speed execution algorithm compared with the conventional Level Set Method, the computational cost is still expensive. To overcome this problem, the Fast Level Set Method (FLSM) was proposed [12].

The key idea of the FLSM is the use of a reference map (Fig.1 (a)). This map indicates the classification of cells according to a distance from a center cell. For example, the class R_r consists of cells which are located $\sqrt{r}$ away from the center cell.

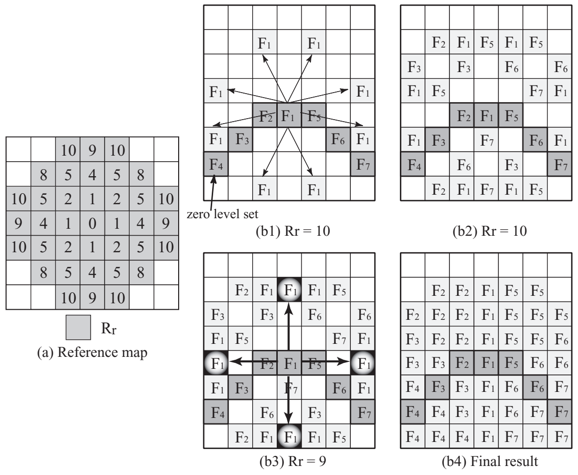

Fig. 1. Reference map and the construction process of the extension velocity field.

The extension velocity field is constructed efficiently using the reference map. Here, it is assumed that the speed function at the zero level set cell has already been determined. At first, one of the zero level set cells is chosen. Then, using the class $R_{\delta(\delta+1)}$ in the reference map, all cells which are located $\sqrt{\delta(\delta+1)}$ away from this zero level set cell are selected, and the speed function at the zero level set cell is registered to these cells tentatively. This procedure is repeated for all the zero level set cells. Next, the same procedure is performed using the next class $R_{\delta(\delta+1)-1}$. In case that some value has already been registered at the cell, new value which is in the closer zero level set cell is overwritten. This process is repeated until all the classes in the reference map are selected. After this registration processes, the speed function of the "nearest" zero level set cell is registered at each cell, and the extension velocity field is constructed consequently (Fig.1 (b)).

III. Fast 3D Shape Reconstruction by the Parallel Fast Level Set Method

In this section, we describe the fast 3D model reconstruction system of real objects using multiple cameras and the FLSM [16]. Then, we introduce a new system using a PC cluster, and propose a new optimization technique for dynamic load balancing and resource allocation on the PC cluster based on the estimation of moving directions of object boundaries.

A. Reconstruction of 3D models against occlusion

Figure 2 shows the developed system using multiple stereo cameras. In this system, each 3D model is recovered according to the following procedure [16].

1) At first, multiple stereo cameras (Point Grey Research, BumbleBee) are installed as shown in Fig.2. For camera calibration, 3D stereo range data of an object with known shape is captured by all stereo cameras, and

mutual camera positions are estimated by applying the iterative closest point (ICP) algorithm to the range data and the object surface. Internal camera parameters are assumed to be calibrated beforehand.
2) Synchronized depth images of the target objects are captured from stereo cameras.
3) Depth images are backprojected onto the 3D voxel space, and voxels containing stereo range data are extracted.
4) The FLSM is applied to the extracted voxels and the isolated smooth surfaces of each person are reconstructed. In an occluding region, the surface remains stationary to keep closed and smooth shape.
5) Repeat steps (2) to (4).

B. Implementation on a PC cluster system

For achieving real-time processing, the above process is implemented on a PC cluster consisting of eight PCs (Pentium Xeon, 3.06 GHz). Each PC of this system is connected with Myrinetxp and three stereo cameras are hooked to three PCs as shown Fig.3. At first, synchronized depth images are captured from stereo cameras. Next, the depth images are sent to four PCs which execute the FLSM calculation (PC1-PC4). Here, the 3D space is divided into four regions, and each region is assigned to one of four PCs, respectively. Since each PC has two CPUs, the FLSM calculation is executed in parallel. Finally, the results of the FLSM calculation are sent to a PC for displaying 3D models.

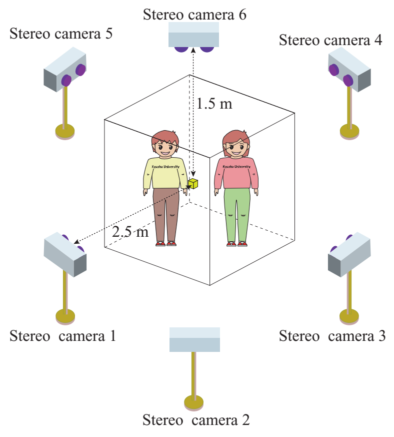

Fig. 2. 3D shape reconstruction system using multiple stereo cameras [16].

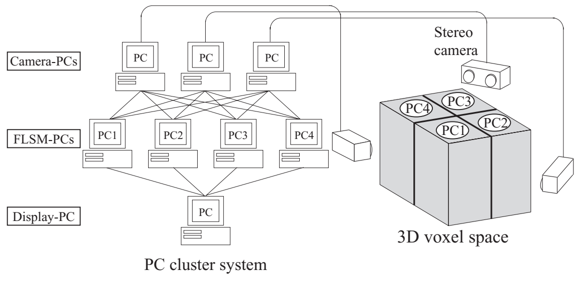

Fig. 3. PC cluster system.

C. Optimization method of load distribution for the Parallel FLSM

Let's consider the case that the 3D space is divided into 4 regions uniformly. Since the computational complexity of the FLSM changes due to the position and the shape of the reconstructed 3D model, it may be occurred that the load is concentrated to some PCs as shown in Fig.4 (a). (Fig.4 shows the cross section of the 3D space.) In addition, in case the target moves quickly, update calculation of the surface has to be executed frequently so as not to cause delay. For allocating resource among PCs dynamically, we develop an effective technique to ensure an enough number of update calculation by decreasing the number of handling voxels of the particular PCs. This optimization technique of load distribution is based on the estimation of moving direction of object boundaries. In this technique, the boundary motion is estimated in the framework of the FLSM, and the optimum load distribution is predicted and performed according to the estimated boundary motion and the current load balance. The flow of the optimization technique is as follows.

1) At first, the 3D model of the target is reconstructed in four regions of the 3D space assigned to FLSM-PCs. The moving direction of object boundary is also estimated in the framework of the FLSM simultaneously.
2) Results of the reconstructed 3D model and the estimated moving direction are sent to the Display-PC from each FLSM-PCs, and then every result is integrated in the whole 3D voxel space.
3) For optimizing the load distribution for the FLSM, division planes of the 3D space are updated in the Display-PC as the following procedure.
   a) The division planes are set according to the position of the target and the number of voxels of the reconstructed 3D model.
   b) The division planes set in step (3.a) are updated according to the estimated moving direction.
4) Results of the division planes updated in step (3) are sent to the FLSM-PCs.
5) Repeat steps (1) $\sim$ (4).

C-1. Estimation of moving direction of object boundary

In this section, an estimation method of moving direction of object boundary is described. Firstly, we extract a surface voxel $Z(t,j)$ at time t ($0 \leq j < num(t)$, $num(t)$ is the number of voxels of the reconstructed 3D model) which is located along the normal direction of a surface voxel $Z(t-\Delta t, i)$ at time $t-\Delta t$ (Fig.5). Corresponding voxel along the normal direction can be detected without additional calculation cost in the framework of the FLSM explained in Section II.B. Firstly, we attach a label $L(t-\Delta t, i)$ uniquely to a surface voxel $Z(t-\Delta t, i)$ as shown in Fig.6 (a). By overwriting with the label at the same time as the process of construction of the velocity field, the label of the nearest surface voxel is stored at each voxel (Fig.6 (a)). Then, the surface is updated at time t, and the corresponding surface voxel along the normal direction can be detected by checking the label stored in the surface voxel $Z(t, j)$ (Fig.6 (b)).

By assuming that local shape of the surface is kept between time $t - \Delta t$ and t, the correctness of the correspondence between the surface voxels $Z(t-\Delta t, i)$ and $Z(t, j)$ can be evaluated by comparing directions of their normal vectors. For example, in case the surface moves upward as shown in Fig.5, the degree of the correctness of the correspondence between the normal direction $N(t - \Delta t, i1)$ and $N(t, j1)$, which are calculated at the surface voxels $Z(t - \Delta t, i1)$ and $Z(t, j1)$ respectively, is high. Therefore the normal direction can be considered as the moving direction of the surface. On the other hand, in case the degree of the correctness of the correspondence between the normal direction $N(t - \Delta t, i2)$ and $N(t, j2)$ at the surface voxels $Z(t-\Delta t, i2)$ and $Z(t, j2)$ is low, it can be considered that the normal direction doesn't coincide with the moving direction of the surface. Thus, the moving direction $\boldsymbol{md(t, j)}$ of the surface voxel $Z(t, j)$ is estimated as the following equation.

$$\begin{aligned}&\boldsymbol{md(t,j)}\\&\quad = F(Z(t,j))|(\boldsymbol{N(t-\Delta t,i)},\boldsymbol{N(t,j)})|\boldsymbol{N(t,j)}\Delta t \qquad (2)\end{aligned}$$

where $F(Z(t, j))$ is the speed function of the FLSM and $\boldsymbol{N(t, j)}$ is the normal direction at surface voxel $Z(t, j)$ as follows:

$$\boldsymbol{N(t,j)} = \frac{\nabla\psi(t,j)}{|\nabla\psi(t,j)|} \qquad (3)$$

Then moving directions of local areas of the surface, that is, the local moving directions are estimated by the following procedure.

1) At first, the surface voxel $Z(t, j)$ is extracted if the magnitude of the moving direction $\boldsymbol{md(t, j)}$ of the surface voxel is larger than a certain threshold c (Fig.7 (a)).
2) Suppose a cube with d on a side, which is centered at the extracted surface voxel $Z(t, j)$. We define this cube as a local moving region L_{l+1} (l is the number of extracted moving regions beforehand), and calculate the moving direction and the number of voxels of the region L_{l+1} as $\boldsymbol{Lmd_{l+1}}$=$\boldsymbol{md(t, j)}$ and $lnum_{l+1}$, respectively.
3) In case the vertex of the cube is involved in other local moving region $L_k (0 \leq k \leq l)$, local moving regions L_k and L_{l+1} are connected each other (Fig.7 (b)). The local moving direction and the number of voxels of the new local moving region are calculated as $Lmd_k \leftarrow \frac{(lnum_k \times Lmd_k + lnum_{l+1} \times Lmd_{l+1})}{(lnum_k + lnum_{l+1})}$ and $lnum_k \leftarrow lnum_k + lnum_{l+1}$, respectively.
4) Repeat steps (1) $\sim$ (3).

Here, c and d are constants.

One of other methods for estimating the moving direction of the target is a method based on time-series data of the centre of gravity of the reconstructed 3D model. However, in case the local areas of the object are moved, it is difficult to estimate the local moving direction of each area by the above method. On the other hand, the proposed method can estimate the moving directions of the local areas of the object. Moreover, the moving directions of the object are estimated without additional calculation cost in the framework of the FLSM.

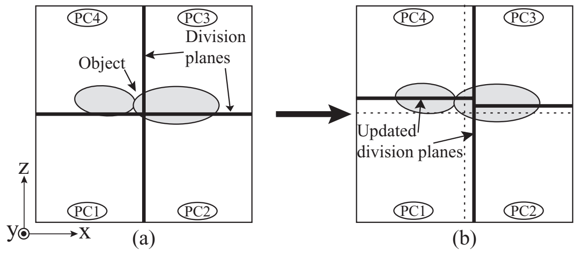

Fig. 4. Division of 3D voxel space according to target's shape.

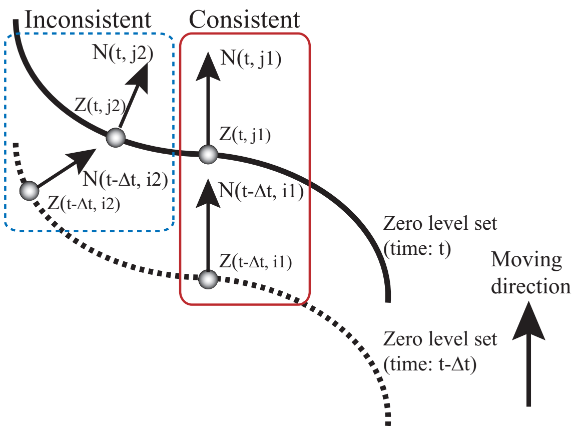

Fig. 5. Estimation of moving direction of object boundary.

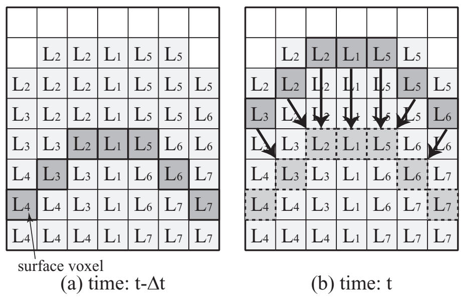

Fig. 6. Detection of surface voxel along the normal direction.

C-2. Optimization method of load distribution for the Parallel Fast Level Set Method based on the estimated moving direction

As shown in Fig.4, the positions of the division planes of the 3D voxel space are set according to the position of the target and the number of voxels of the reconstructed 3D model as follows: the reconstructed 3D models are sent from FLSM-PCs to the Display-PC, and these 3D models are integrated in the 3D voxel space. Then, firstly the data of the reconstructed 3D model is sorted according to the x-coordinate value. A yz-plane dividing the 3D voxel space is set with the central value D_x of the sorted data, and the 3D voxel space is divided into two regions. Next, the data of the 3D model in each region is sorted according to the z-coordinate value. Two xy-planes dividing the two regions are set with the central values D_{z1}

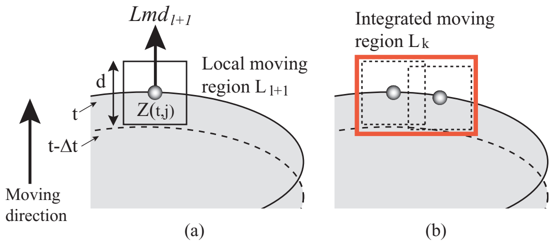

Fig. 7. Estimation of local moving direction.

and D_{z2} of the two sorted data. By this procedure, the 3D voxel space is divided into four regions so that the number of voxels of the 3D model is equalized as shown in Fig.4 (b).

Next, for allocating resource among PCs dynamically, the division planes are updated so as to ensure the enough number of update calculation for the PC which contains the moving surfaces. This optimization procedure is realized by considering the number of handling voxels of the PC and the estimated local moving direction $\boldsymbol{Lmd}$. Let's consider the case shown in Fig.8(a). In this case, according to the local moving direction $\boldsymbol{Lmd}$ and the local moving region (dotted lines), the z coordinate value D_{z1} of the division plane P_1 is updated as the following equation.

$$D_{z1} \leftarrow D_{z1} + e\frac{min(p_1, p_2)}{(p_1 + p_2)} Lmd_z \qquad (4)$$

where Lmd_z is the value of the local moving direction $\boldsymbol{Lmd}$ along z axis, p_1 and p_2 are the lengths of the local moving region from the division plane P_1 along z axis, and e is constant. If the local moving region contains the division plane P_1, e is set to 5.0, otherwise e is set to 0.0. By moving the division plane according to the above procedure, the number of update calculation of the FLSM can be increased, since the load of the PC1 which involves the moving surface is reduced. As a result, the object surface can be tracked without delay even if the target moves quickly. In the same way, the positions of other division planes are updated according to the estimated local moving direction of the object surfaces.

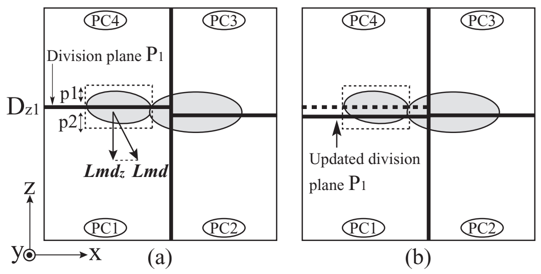

Fig. 8. Division of 3D voxel space based on estimated moving direction.

IV. Experiments

In this section, firstly we show some experimental results of the estimation of the moving direction with simulated models. Next, the fast 3D shape reconstruction and arbitrary viewpoint image synthesis of dance motion are shown.

A. Estimation of the moving direction with simulated models

Firstly, we carried out experiments of the estimation of the moving direction using two kinds of simulated models. The 3D space is a cube 1.5m on a side, and the resolution of the 3D space is $100 \times 100 \times 100$ (1.5cm voxel on a side). In an experiment with a sphere as shown in Fig.9(a), the sphere is translated toward y axis and rotated around y axis, and the moving directions of the sphere are estimated. Table I shows the average and the standard deviation of angles between the estimated local moving direction and the correct moving direction. The correct barycentric velocity of the sphere and the estimated moving velocity are shown in Table II.

Next, in an experiment of an object combining two pillars and two spheres as shown in Fig.9(b), the local moving directions are estimated. In this experiment, the local regions A and B are moved along the x-axis and z-axis, respectively, and the local moving direction of these regions are estimated. Table III shows the average and the standard deviation of angles between the estimated local moving direction and the correct moving direction.

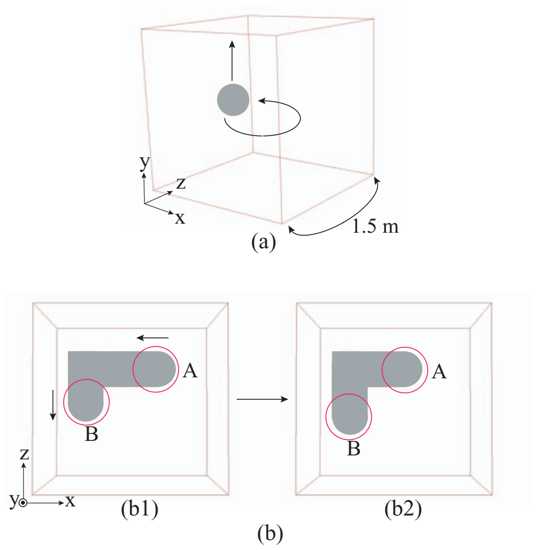

Fig. 9. Estimation of the moving direction.

TABLE I
ESTIMATION ERROR OF MOVING DIRECTION.

	Average [deg.]	Standard deviation [deg.]
Translation toward y axis	2.81	1.68
Rotation around y axis	4.09	2.29

TABLE II
ESTIMATION ERROR OF MOVING SPEED.

Barycentric velocity [m/sec.]	0.045	0.060	0.075	0.090
Estimated moving velocity [m/sec.]	0.045	0.059	0.077	0.091

B. 3D model reconstruction of a moving target

Next, we captured the dancing motion using the developed stereo-based motion capture system, and tracked and

TABLE III

ESTIMATION ERROR OF LOCAL MOVING DIRECTION.

	Average [deg.]	Standard deviation [deg.]
A	5.68	3.26
B	4.83	2.50

reconstructed the 3D model in real-time. Figure 10(a) shows the motion of the right arm at t=1.7∼3.3 [sec.]. In this experiment, the resolution of the 3D space is 80 × 80 × 80 (1.875cm voxel on a side), and the stereo range data is calculated at every 100 [msec.].

Experimental conditions are as follows: (Exp.A) the 3D space is uniformly divided into four regions, (Exp.B) the 3D space is divided according to the number of voxels of the reconstructed 3D model, (Exp.C) the 3D space is divided according to the estimated local moving direction for increasing the number of update calculation of the FLSM for a particular PC which involves the moving surface (proposed method), and (Exp.D) the sufficient number of update calculation of the FLSM is executed by offline calculation. Figure 10 (b)∼(e) show the experimental results of (Exp.A∼D). The number of voxels of the reconstructed 3D model in (Exp.A∼C) and (Exp.D) are shown in Figs.11 and 12, respectively. As seen from these results, the number of voxels reconstructed by (Exp.C) and (Exp.D) are quite similar compared to the number of voxels by (Exp.A) and (Exp.B) at t=1.7∼3.3.

In this experiment, since the right arm is moved rapidly in the region assigned to PC1, the number of handling voxels of PC1 should be decreased for reconstructing the accurate 3D model. Table IV shows the calculation time of the FLSM with PC1 in (Exp.A∼C). The average processing time for 1 updating period at t=1.7∼3.3 in Exp.C is 10.58 [sec.]. On the other hand, the processing time in Exp.A and B are 16.24 and 12.79 [sec.], respectively.

Figure 13 and Table V show maximum, average, and standard deviation of errors of the reconstructed 3D model in (Exp.A∼C), respectively. The error is defined as the minimum distance between voxels in (Exp.D) and (Exp.A∼C). From these results, we can conclude that accurate 3D model is reconstructed by (Exp.C) compared to the models by (Exp.A) and (Exp.B).

TABLE IV

COMPARISON OF CALCULATION TIME WITH PC1.

	Average [msec.]	Standard deviation [msec.]	Max. [msec.]
Exp.A	16.24	6.982	48.10
Exp.B	12.79	6.343	36.94
Exp.C	10.58	5.468	27.83

C. Generating arbitrary viewpoint images

We carried out experiments of arbitrary viewpoint image synthesis using six stereo cameras (Fig.2). The resolution of the 3D space is 100 × 100 × 100 (1.5cm voxel on a side).

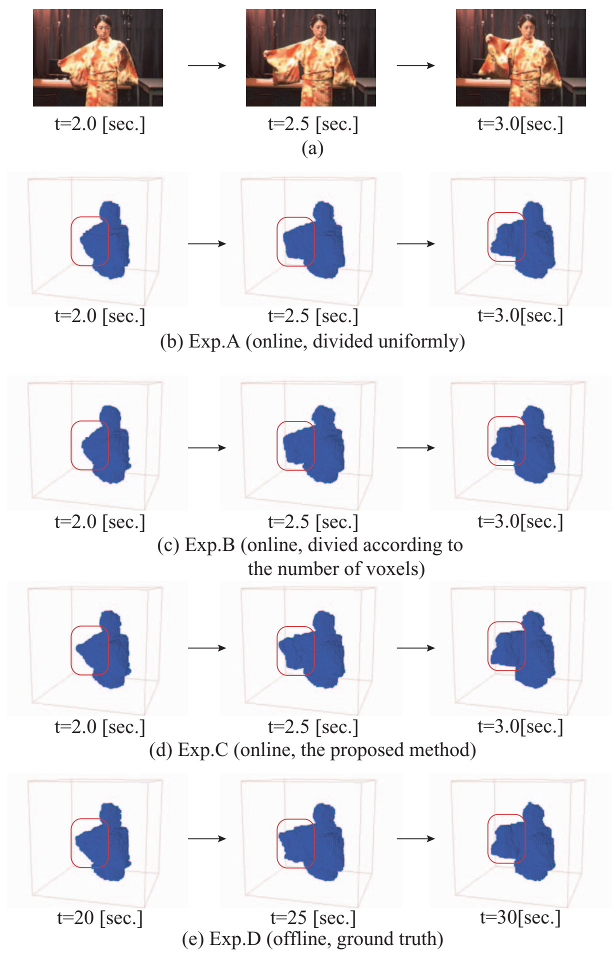

Fig. 10. 3D shape reconstruction of moving object.

TABLE V

COMPARISON OF ERRORS OF RECONSTRUCTED 3D MODEL.

	Average [cm]	Standard deviation [cm]	Max. [cm]
Exp.A	0.5839	1.101	17.18
Exp.B	0.3774	0.8658	13.26
Exp.C	0.3017	0.7217	7.016

Figures 14 (a1) and (a2) show examples of captured color images and Figs. 14 (b1) and (b2) are synthesized images. The 3D voxel model is converted into triangular patches by the Discrete Marching Cubes algorithm, and then the captured texture images are mapped to the triangular patches.

V. CONCLUSION

This paper proposed a parallel and efficient algorithm of the Level Set Method named the Parallel Fast Level Set Method and the optimum load distribution technique on a PC cluster which realizes dynamic load balancing and resource allocation. The proposed technique is based on the estimation of the moving directions of object boundaries, and the boundary motions are estimated in the framework of the Fast Level Set Method. The optimum load distribution is predicted and performed according to the estimated boundary

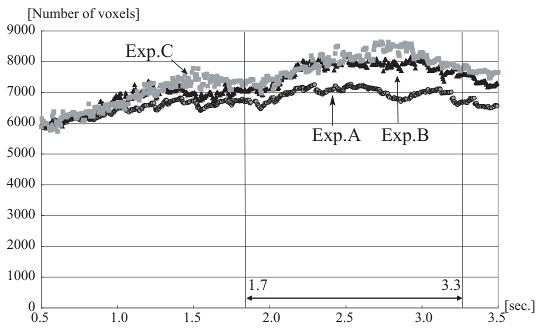

Fig. 11. Number of voxels of reconstructed 3D model in (Exp.A ∼ C).

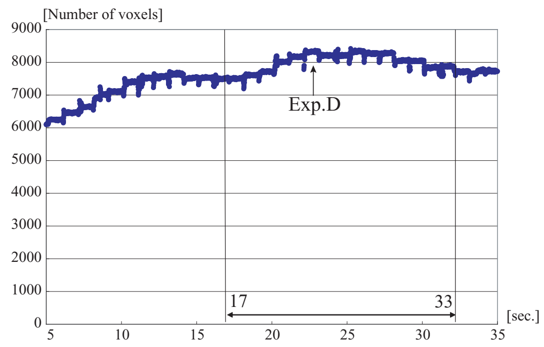

Fig. 12. Number of voxels of reconstructed 3D model in (Exp.D).

motion and the current load balance. The efficiency of the proposed method was verified through the tracking experiments of dance motion. In the future work, we have the following two plans: i) we implement the proposed system on a PC cluster with sixteen PCs and generate more precise arbitrary viewpoint images in real-time; ii) we apply the proposed system to acquisition system of human skill for robot applications.

References

[1] M. Levoy, P. Hanrahan, "Light Field Rendering", *ACM SIGGRAPH*, pp.31-42, 1996

[2] T. Kanade, P. Rander and P. Narayanan, "Virtualized Reality: Constructing Virtual Worlds from Real Scenes", *IEEE Multimedia, Immersive Telepresence*, vol.4, No.1, pp.34-47, 1997

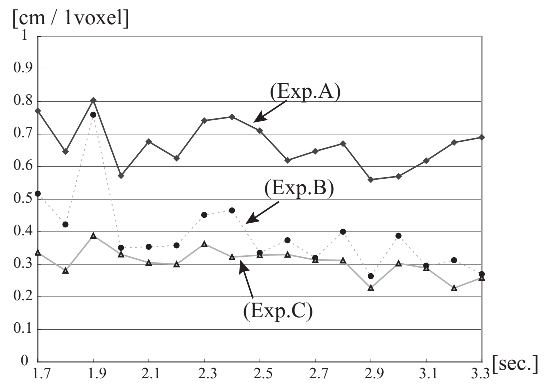

Fig. 13. Errors of reconstructed 3D model.

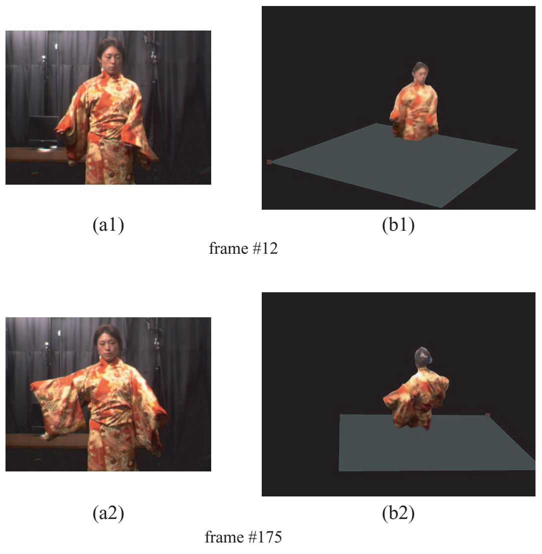

Fig. 14. Generating free viewpoint images.

[3] S. Vedula and S. Baker and T. Kanade, "Image-Based Spatio-Temporal Modeling and View Interpolation of Dynamic Events", *ACM Transactions on Graphics*, vol.24, No.2, April, 2005

[4] J. Carranza and C. Theobalt and M.A. Magnor and H.-P. Seidel, "Free-viewpoint video of human actors", *ACM SIGGRAPH*, pp.569-577, 2003

[5] W. Martin and J. Aggarwal, "Volumetric description of objects from multiple views", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.5, No.2, pp.150-158, 1983

[6] A. Laurentini, "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

[7] O. Faugeras and R. Keriven, "Complete dense stereovision using level set methods", *In Fifth European Conference on Computer Vision*, vol.1, pp.379-393, 1998

[8] J.-S. Franco and E. Boyer, "Fusion of multi-view silhouette cues using a space occupancy grid", *IEEE Intl. Conf. on Computer Vision*, vol.2, pp.1747-1753, 2005

[9] L. Guan and J.-S. Franco and M. Pollefeys, "3D Occlusion Inference from Silhouette Cues", *IEEE Conf. on Computer Vision and Pattern Recognition*, pp.1-8, 2007

[10] K. Kutulakos and S. Seitz, "A theory of shape by space carving", *International Journal of Computer Vision*, vol.38, No.3, pp.199-218, 2000

[11] S. Nobuhara and T. Matsuyama, "Heterogeneous Deformation Model for 3D Shape and Motion Recovery from Multi-Viewpoint Images", *Proc. of the 2nd International Symposium on 3D Data Processing, Visualization, and Transmission*, pp.566-573, 2004

[12] Y. Iwashita and R. Kurazume and T. Tsuji and K. Hara and T. Hasegawa, "Fast Implementation of Level Set Method and Its Realtime Applications", *IEEE International Conference on Systems, Man and Cybernetics*, pp.6302-6307, 2004

[13] J. Sethian, "A fast marching level set method for monotonically advancing fronts", *Proceedings of the National Academy of Science*, vol.93, pp.1591-1595, 1996

[14] J. Sethian, "Level Set Methods and Fast Marching Methods, second edition", *Cambridge University Press*, UK, 1999

[15] D. Adalsteinsson and J. Sethian, "The fast construction of extension velocities in level set methods", *J. Computational Physics*, vol.148, pp.2-22, 1999

[16] Y. Iwashita and R. Kurazume and T. Tsuji and K. Hara and T. Hasegawa, "Robust Motion Capture System against Target Occlusion using Fast Level Set Method", *Proc. of IEEE International Conference on Robotics and Automation*, pp.168-174, 2006