Cubistic Representation for Real-time 3D Shape and Pose Estimation of Unknown Rigid Object

Hiromasa YOSHIMOTO and Yuichi NAKAMURA
Academic Center for Computing and Media Studies, Kyoto University
Sakyo-ku, Kyoto 606-8501, Japan
yoshimoto@ccm.media.kyoto-u.ac.jp

Abstract

This paper introduces Cubistic Representation as a novel 3D surface shape model. Cubistic representation is a set of 3D surface fragments; each fragment contains subject's 3D surface shape and its color and redundantly covers the subject surface. By laminating these fragments using a given pose parameter, the subject's appearance can be synthesized.

Using cubistic representation, we propose a real-time 3D rigid object tracking approach by acquiring the 3D surface shape and its pose simultaneously. We use the particle filter scheme for both shape and pose estimation; each fragment is used as a partial shape hypothesis and is sampled and refined by a particle filter. We also use the RANSAC algorithm to remove wrong fragments as outliers to refine the shape.

We also implemented an online demonstration system with GPU and a Kinect sensor and evaluated the performance of our approach in a real environment.

1. Introduction

Scene understanding is one of the central goals of classic computer vision research. From the early days in computer vision, scene understanding is formulated as a 3D geometric and photometric reasoning process that acquires a 3D geometric and photometric description of the world from input images. For example, Marr *et al*. [18] proposed a fundamental framework that uses a generalized cylinder which represents the geometry of an object's physical surface, and that outputs 3D scene description by finding the correspondence between the input image and the shape model. After leading works, such as [3, 17], scene understanding is now considered as a mixture of several tasks, such as object detection, pose estimation, 3D reconstruction, and so on.

For these sub tasks, we address two problems. The first problem is mutual dependencies among image, shape, and pose. As shown in Fig.1, scene understanding can be considered as an inverse process that recovers a pair of 3D surface shape $\hat{\boldsymbol{S}}$ and its pose $\hat{\boldsymbol{P}}$ from the input image $\hat{\boldsymbol{I}}$. Here $\hat{\boldsymbol{I}}$ is sensing result that contains degenerated information of actual shape $\boldsymbol{S}$, its actual pose parameters $\boldsymbol{P}$ and measurement noise $\varepsilon_{\boldsymbol{I}}$. As known as an ill-posed problem or a chicken-egg problem, these variables depend on each other; even small differences in a shape and a pose may lead to considerable appearance differences and variations in images. As a result, these mutual dependencies create ambiguities and difficulties in image interpretation. Moreover, in a real environment, these variables have another ambiguities delivered from measurement and estimation errors; 3D shape quantization or its recovering process causes $\varepsilon_{\boldsymbol{S}}$, and pose estimation causes $\varepsilon_{\boldsymbol{P}}$. These inseparable dependency and ambiguity cause the difficulties in simultaneous shape and pose acquisition from input images.

The second problem is a computational cost. Generally, it requires vast amount of computations to explore the large space for possible solutions under the condition of the mutual dependencies. For example, a possible algorithm takes the form of expectation-maximization algorithm (EM) and requires a large number of iterations. Moreover, other important factors in vision, such as shadings and occlusions, may easily make the search space much larger and intractable. We need much improvement on both on computation algorithms by suppressing the enormous amount of combination problems and massive computations with fast and parallel processors such as a graphic processing unit (GPU).

For this purpose, we introduce a novel 3D shape representation, named cubistic representation. As shown in Fig.2, a cubistic representation consists of a set of 3D surface fragments; each fragment contains subject's 3D surface shape and its color, and redundantly covers the subject surface. By laminating the subject with these fragments using a given pose parameter, the subject's appearance can be synthesized.

One of the advantages of cubistic representation is its redundancy. Cubistic representation allows fragments over-

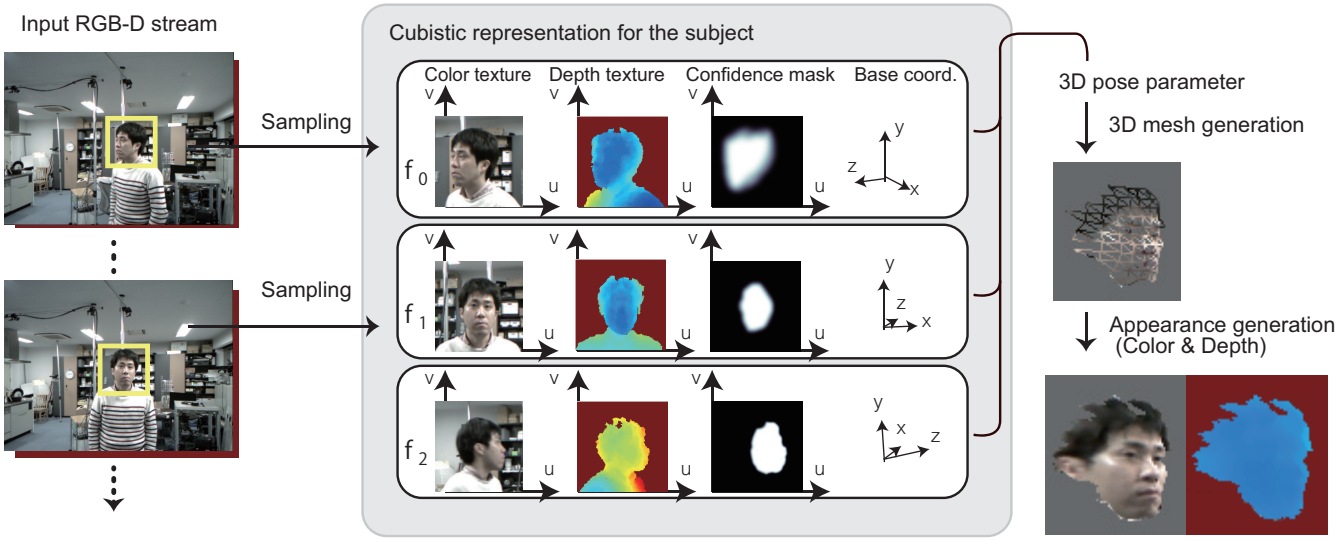


Figure 2: An example of cubistic representation: (center) cubistic representation consists of a set of 3D surface fragments; (left) each fragment is sampled from the input RGB-D images; and (right) the synthesized subject's appearance.

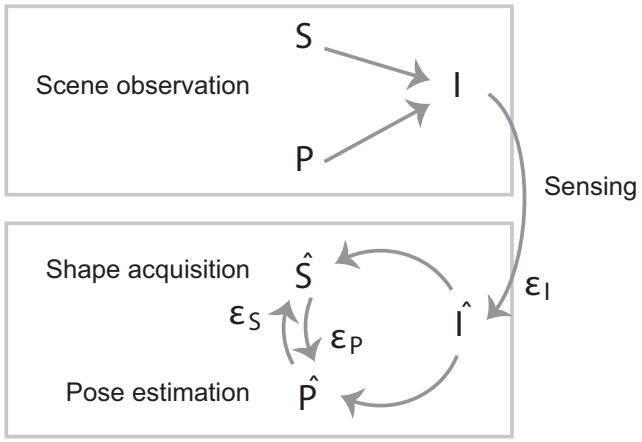


Figure 1: Simultaneous shape and pose estimation is an ill-posed inverse problem: The input image $\hat{\boldsymbol{I}}$ is an observed result of actual shape $\boldsymbol{S}$ and pose $\boldsymbol{P}$. Recovering the shape $\hat{\boldsymbol{S}}$ and the pose $\hat{\boldsymbol{P}}$ from $\hat{\boldsymbol{I}}$ has to deal with the mutual dependencies among them, and consider the measurement error $\varepsilon_{\boldsymbol{I}}$ and the estimation errors $\varepsilon_{\boldsymbol{S}}$ and $\varepsilon_{\boldsymbol{P}}$.

lapping with each others. This characteristic enables the following two approaches. The first is an online shape acquisition. We can consider that each fragment of a 3D shape as a particle in the particle filter scheme: possible partial shapes can be incrementally generated and refined. The second is error detection and its correction. According to the redundancies among the fragments, voting approach can be used to detect the error. Wrong fragments in the subject's shape can be detected by less votes. The wrong fragments with small votes can be removed without serious deformation of the acquired shape. Moreover, other 3D geometrical factor, such as object shadings, surface connectivities and self-occlusions can be introduced in this process.

Another advantage of cubistic representation is the essential parallelism of the particle filter scheme. The above processes can be implemented by using the state-of-the-art technologies of modern GPU to solve the computational cost problem.

To show these advantages of cubistic representation, we implemented an experimental algorithm that extracts a rigid object from the input RGB-D video stream and recovers the object's 3D surface shape with tracking its six-degree of freedom (DoF) pose parameters. Each fragment is considered as a shape hypothesis, which is sampled and refined by a particle filter. Once a shape hypothesis is sampled, pose estimation can be performed by the so-called model fitting process that uses a pair of a given input image and a set of current fragments. The estimated pose is used as next pose hypothesis for obtaining next fragments. For error detection and correction, we use a random sample consensus (RANSAC) algorithm [5] to remove wrong fragments as outliers.

To show the capability and efficacy in a real environments, we implemented an online demonstration system with GPU and a Kinect sensor. This system is real-time head tracking, which detects unknown human head as a single rigid object with separating the head from the other body parts, such as his neck, body and the background. With this system, we evaluate the performance of our approach in a

real environment.

2. Related work and problem statement

In this section, we will first discuss the problem of resolving mutual dependencies in scene understanding. Then, we will explain our idea to solve the problem.

Simultaneous shape acquisition and pose estimation is still a challenging problem in computer vision because of the complexity of the mutual relationship. Many of the practical techniques in computer vision are dealing with special cases, for example, where either of the shape or the pose is known.

When the subject's shape is given, its pose estimation from input images can be formulated as a model fitting or a tracking process. Active appearance model (AAM) [4], for example, provides a good basis for a shape model. AAM is a deformable 3D surface representation, with which accurate appearances can be synthesized with various conditions of 3D geometry and photometry, such as self-occlusions and shadings. A typical problem associated with these pose estimating approaches is that the performance of pose estimation depends on the quality of the shape model. As shown in Fig.1, if the shape model has some error ε_S, it will propagate to the pose parameters as ε_P.

On the other hand, when the subject's pose is fixed, shape acquisition from input images can be formulated as a 3D reconstruction process. Some classical approaches use voxel-based volume reconstruction [13], surface generation [15] and 3D shape registration techniques [1, 2]. A typical problem in these shape acquisitions is also that the quality of the acquired shapes is affected by the accuracy of the given poses, *i.e*., pose errors ε_P will propagate to shape errors ε_S.

Another special case is where the subject's motion is known. For example, Tomasi–Kanade factorization can be used as a direct method which allows resolution of shape and pose by a one-shot algorithm, especially for a rigid object with weak perspective camera model [22]. This is an elegant and simple solution. However, it doesn't deal with the object detection and segmentation problems. Factorization is commonly used with some pre-processors, such as 2D image feature detectors and its trackers [16, 20]. A typical problem in these combinations arises from the limitation of these 2D image processes, *i.e*., they cannot analyze the image appearances in term of 3D geometry.

For simultaneous shape acquisition and pose estimation, we often need a combination of the above processes. For instance, PTAM [11], DTAM [19] and Kinect Fusion [8] use this combination. The purpose of PTAM and DTAM is simultaneous localization and mapping, which estimates camera positions and orientations and a 3D geometric representation of the scene at the same time. PTAM, for example, assumes the world as a single rigid object and uses a point cloud as its representation. Then, PTAM calculates the point cloud and pose by using the above processes. DTAM and Kinect Fusion use voxel-based representation of the scene, which help more correct image analyses considering 3D geometries such as occlusions.

Although these approaches work fine under ideal conditions, there are two serious problems in a real environment. The first problem is concerning segmentation. Without a proper segmentation process to separate moving objects from the static background, dynamic scenes with moving objects would make the shape model inaccurate. The second problem is concerning cumulative errors. As already shown in Fig.1, once the acquired shape has considerable errors, the pose estimation is seriously degraded, and this poor estimation affects the shape estimation in turn. Previous works [8, 11, 19] discussed the problem of these errors.

For the segmentation problem, Kalal *et al*. approach [9] proposed a 2D object tracking approach that acquires the target's appearances in on-line processing. It is basically a 2D tracking; however, it collects the key views of the target at the same time. In other work, Wenze *et al*. proposed an object recognition approach that uses a hybrid model mixing of 3D and 2D primitives as key views [6]. For the cumulative error problem, a possible solution is to use voting or RANSAC algorithms. As related works [14, 21] shows, this voting-based approach can work robustly under real environment.

Our idea is to solve these problems and issues at the same time by using the particle filter scheme for not only pose parameters but also shape representation. As the first, this paper proposes a novel idea for shape representation, which achieves a real-time 3D shape acquisition and pose estimation of an unknown rigid object.

3. Cubistic representation

The key idea of our approach is to use a set of hypotheses for both of shape and pose to overcome the problems mentioned in the previous section. For this purpose, we introduce cubistic representation as a 3D surface shape model. As an example is shown in Figure 2, cubistic representation is a set of 3D surface fragments. Each fragment $\boldsymbol{f}$ has an information of 3D geometry, including the surface shape and its color, and redundantly covers the subject's surface. This idea is similar to a set of key views proposed in [6, 9]; however, cubistic representation is different in containing strict 3D geometric information. By laminating these fragments using a given pose parameter, the subject's appearance can be synthesized with correct shadings and self-occlusions.

Cubistic representation has the following three advantages. First, editing 3D surface shape is realized as simple fragment-wise insertion or deletion operation. This helps online 3D surface shape acquisition process. Second, the estimation errors of 3D shape can be detected and corrected

by using the redundancy among its fragments. Third, the data structure is designed and optimized for GPU architecture. This helps correct and fast image interpretation with strict consideration of geometry and photometry.

Using cubistic representation, the 3D shape acquisition becomes a sampling task that gathers a group of N_f fragments $\{\boldsymbol{f}^k; k = 1, \ldots, N_f\}$ that covers the target object surface $\boldsymbol{S}$. This also solves the 3D surface segmentation issues that require separation of the target from other object in the scene. The shape's error detection can be applied through a task that categorizes all fragments into "inlier" or "outlier" group; the former is the sub-set of fragments that supports the current pose hypothesis and the latter is its complement. Simultaneously, improvement in the acquired 3D surface shape can be achieved as a simple fragment-wise operation: insertion or deletion of a fragment. Moreover, the pose estimation can be solved the simple object tracking by using the group of fragments as subject's shape model.

We designed the exact data structure of cubistic representation so that it can be processed efficiently by GPU and recent computer graphics techniques. Data structure for fragment $\boldsymbol{f}$ is defined by the following data; base coordinates, color texture, heightmap texture, and confidence mask texture. The base coordinates defines a reference uv plane in the world coordinate system, where u and v are 2D texture coordinates. The heightmap texture is a 2D image defined on uv plane. Heightmap texture stores the 3D shape information where each pixel stores the surface evelation from the uv plane. Color texture is a 2D image that stores color of the surface.

When using an RGB-D camera, such as the Kinect sensor or a stereo camera, those heightmap and color textures can be directly extracted from the input RGB-D image by image cropping. This is an advantage of our representation because this process does not add any quantization errors and distortions. It can avoid any undesirable artifacts and approximations that are required in existing 3D surface representation, such as 3D mesh or parametric representation.

The confidence mask is also a 2D texture on the uv plane. This texture is used to store the pixel-level segmentation result. Each pixel value indicates the relevance of $\boldsymbol{\varepsilon_P}$ and is in the range of 0 to 1. The higher value indicates that the corresponding color and heightmap pixel belong to the target rigid object and has less error. This confidence mask can be used as make image for color and depth texture.

In modern computer graphics techniques, the synthesis of constructing a 3D appearance from a combination of fragment textures and its pose in 3D is the so-called *heightmap rendering* process. Fig.2 shows the overview of the process. A 3D mesh is dynamically generated from the fragments by using an arbitrary given pose parameter. The resolution of the 3D mesh is dynamically controlled using a level-of-detail technique where each triangles on the mesh is subdivided by GPU's hardware tessellator with a specified resolution. As results, a pair of color and depth images is synthesized. We carefully designed the fragment data structure by using a set of texture so that recent GPU can execute these operations in an accurate and faster way.

Using the model fitting scheme, the similarities of a shape $\{\boldsymbol{f}\}$ and a pose $\boldsymbol{p}$ to an image $\hat{\boldsymbol{I}}$ is calculated by the appearance difference $D(\hat{I}, G(p, \{f\}))$, where $G(p, \{f\})$ is an image generated by CG rendering process, and $D(I, I')$ is a function that evaluates the differences in the two images $\boldsymbol{I}$ and $\boldsymbol{I'}$ by comparing the each color and depth.

Model fitting scheme also requires partial derivations for its non-liner optimization. The partial derivation of D at a given $\boldsymbol{p}$ with respect to the i-th variable in pose vector $\boldsymbol{p}$ is approximated as follows:

$$\frac{\delta D(\hat{I}, G(p,f))}{\delta T_x} \approx \frac{D(\hat{I}, G(p + \Delta p_i, f)) - D(\hat{I}, G(p - \Delta p_i, f))}{|\Delta p_i|}.$$

where Δp_i is a vector such that only the i-th entry is nonzero. This calculation is also dispatched by GPU.

Fig.3 shows an example of the pose estimating process by minimizing $D(\hat{I}(\tau), G(p_0, f_0))$. The result is shown in Fig.3(c), where the color indicates the re-projection errors that are pixel-wise difference between $\hat{\boldsymbol{I}}(\tau)$ and fitted $\boldsymbol{f}_0$; a red pixel indicates no difference and a green pixel indicates a significant difference. Because some pixels laying on the neck region are visualized as green, our cubistic representation can detect each pixel is located target rigid region or not. Our algorithm estimates the error of shape $\boldsymbol{\varepsilon_S}$ according to these pixel-wise differences.

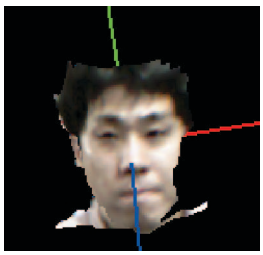
4. Simultaneously 3D shape acquisition and pose estimation algorithm

Using cubistic representation, we solve the simultaneous 3D shape acquisition and its pose estimation as follows. First, we formulate the pose estimation task as pose tracking process that uses a particle filter scheme. We also use a particle filter approach for shape acquisition process, which generates an optimal set of N_c fragments so that maximum possible target surface is covered.

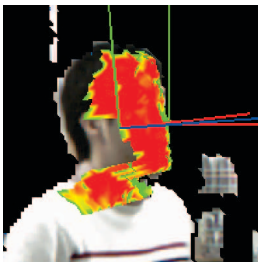
An overview of our algorithm is shown in Algorithm 1. This algorithm requires only two inputs; initial seed position $\boldsymbol{p}(0)$ and RGB-D image stream $\{\hat{\boldsymbol{I}}(t)\}$. Once $\boldsymbol{p}(0)$ is given, our algorithm begins the sampling of fragments around $\boldsymbol{p}(0)$.

This algorithm uses two particles: $\{\boldsymbol{p}^k(t); k = 1, \ldots, N_{\boldsymbol{p}}\}$ for $\hat{\boldsymbol{P}}(t)$ and $\{\boldsymbol{f}^k; k = 1, \ldots, N_{\boldsymbol{f}}\}$ for $\hat{\boldsymbol{S}}$, where $N_{\boldsymbol{p}}$ and $N_{\boldsymbol{f}}$ are the number of particles. These particles are updated using the Condensation [7]. For $\hat{\boldsymbol{S}}(t)$ estimation, RANSAC [5] is also used to categorize $\{\boldsymbol{f}^k\}$ into

(a) The input image $\hat{\boldsymbol{I}}(\tau)$



(b) The input fragment $\boldsymbol{f}_0$



(c) Fitting result

Figure 3: An example of pose estimation: (a) the input RGB-D image at time τ; (b) the input fragment $\boldsymbol{f}_0$ sampled at another time; and (c) the fitting result. (c) is a synthesized image that shows both of the fitting result and its re-projection errors; a red pixel indicates a significant error is almost zero, and a green pixel indicates the error is not small. Green pixels laying on the neck region mean that this region is deformed since $\boldsymbol{f}_0$ is sampled.

"inlier" or "outlier" group. The fragment in outlier group will be randomly selected and replaced with the new fragment.

Please note that this algorithm is not an optimum one; it just uses the greedy optimization approach. The present algorithm is designed to show the advantages and effectiveness of our approach. We have already noticed that the latest particle filter algorithm, such as the MCMC-based particle filter [10], would yield better results. This is a future work of this paper.

5. Experiments

To show the practicality and reliability of our proposed approach, we implemented a real-time head tracking system. Since our experimental algorithm requires a 3D seed position to search a rigid object, we use the Viola–Jones face detector [23]. Once the face detector detects the frontal face region, our algorithm starts to extract the head object region and its motion tracking.

Algorithm 1: Simultaneous 3D shape acquisition and pose estimation algorithm using cubistic representation

Randomly initializes $\{\boldsymbol{f}\}$ using the given $\boldsymbol{p}(0)$
while do
  $\hat{\boldsymbol{I}}(t) \leftarrow$ RGB and depth images at time t
  Updates $\{\boldsymbol{p}^k(t)\}$ from $\{\boldsymbol{p}^k(t-1)\}$ by using Condensation
  foreach j *in uniformly chosen indices from* $[1, N_{\boldsymbol{p}}]$ **do**
    $\mathcal{F} \leftarrow$ Uniformly choose N_c fragments from $\{\boldsymbol{f}^k\}$
    $\boldsymbol{p}^j(t) \leftarrow \underset{p}{\operatorname{argmin}}\, D(\hat{I}(t), G(p, \mathcal{F}))$ using $\boldsymbol{p}^j(t-1)$ as initial value
    $consensus_j \leftarrow$ the number of inlier fragments for $\boldsymbol{p}^j(t)$
  $\hat{\boldsymbol{P}}(t) \leftarrow \boldsymbol{p}^k(t)$ where $k = \underset{j}{\operatorname{argmax}}(consensus_j)$
  foreach $\boldsymbol{f}$ *in* $\{\boldsymbol{f}^k\}$ **do**
    Updates $\boldsymbol{f}$'s confidence mask using $\hat{\boldsymbol{P}}(t)$ and $\hat{\boldsymbol{I}}(t)$
  foreach j *in uniformly chosen indices from* $[1, N_{\boldsymbol{f}}]$ **do**
    if *avg. of confidence values of* $\boldsymbol{f}^j$ *is low* **then**
      $\boldsymbol{f}^j \leftarrow$ New fragment sampled from $\hat{\boldsymbol{I}}(t)$ with $\hat{\boldsymbol{P}}(t)$
    else
      Updates base coord. of $\boldsymbol{f}^j$ with $\underset{basis}{\operatorname{argmin}}\, D(\hat{I}(t), G(\hat{P}(t), f^j))$

5.1. System configuration

For experiment, we use the following hardware; nVidia GTX 580 for GPU, an Intel i7 3.2GHz CPU with linux OS, and Kinect sensor as the RGB-D camera. Prior to conducting the experiment, we calibrated the Kinect sensor by using an established method [12].

Our implementation uses no OpenNI or Kinect SDK features. Most of the computations are implemented using OpenGL 4.4 features and executed by GPU. Once the raw bayer pattern image and disparity image are captured from the Kinect sensor, these two images are directly forwarded into GPU's memory. The rest of processes, such as bayer-decoding, distortion correction and other processes, are executed by GPU. Each processes are implemented using modern OpenGL shader language (GLSL). For example, we use geometry shader for real-time tessellation in the rendering stage and compute shaders for calculation of the function $D()$ and its partial derivation.

Table 1: Computational costs of typical operations: each operations is executed on the GPU, and fast enough for real-time processing.

Operation	Average of computation time	Concurrency
Frag. rendering	0.02 ms	512 frag./GPU
Frag. sampling	0.01 ms	512 frag./GPU
Frag. evaluation	0.03 ms	512 frag./GPU

5.2. Results

First, we measured the processing times of typical fundamental operation with cubistic representation. Table 1 shows the benchmark results using cubistic representation. Each fragment operation, sampling, rendering and evaluation, requires a maximum of only 0.03 ms per fragment. Moreover, up to 512 fragments can be executed in parallel with a single GPU.

We also measured the throughput of the 3D shape and pose estimation system. The system requires approximately 29–80 fps. These benchmark results show that our implementation is presently fast enough for real-time applications.

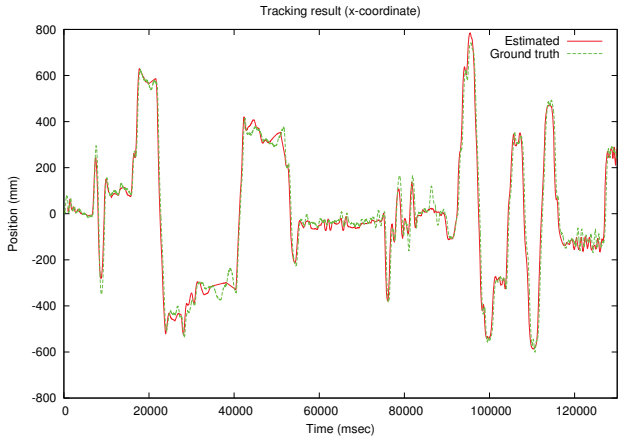
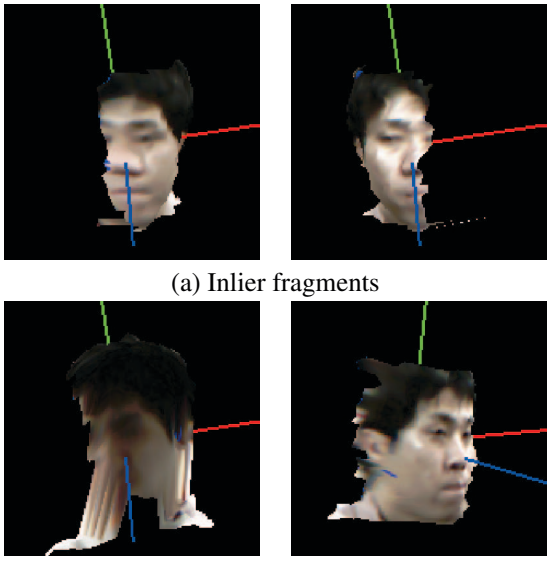


Figure 4: Accuracy of the pose estimation (red line) with respect to target x position (green line).

We also measured the accuracy of the pose estimation by comparing our tracking real-time with the ground truth data. The ground truth data are captured by an electromagnetic 3D motion tracking systems, Polhemus LIBERTY. As this graph shows our algorithm can estimate the 3D pose accurately in the real environment. Fig.4 illustrates some of the results where the red line indicates x position of the head and green line indicates the derived from the ground truth data.

As 3D shape acquisition results, Fig.6 shows some fragments gathered in the experiment. The fragments in (a) are the samples accepted as inliers, and the fragments in (b) are rejected as outliers because of motion blurs and segmentation failure. As these images show, our algorithm can detect errors in shape acquisition. Fig.6 also shows pixel-level segmentation result. The inlier fragments are correctly separated the head region from the other portion, such as neck and body.



(a) Inlier fragments

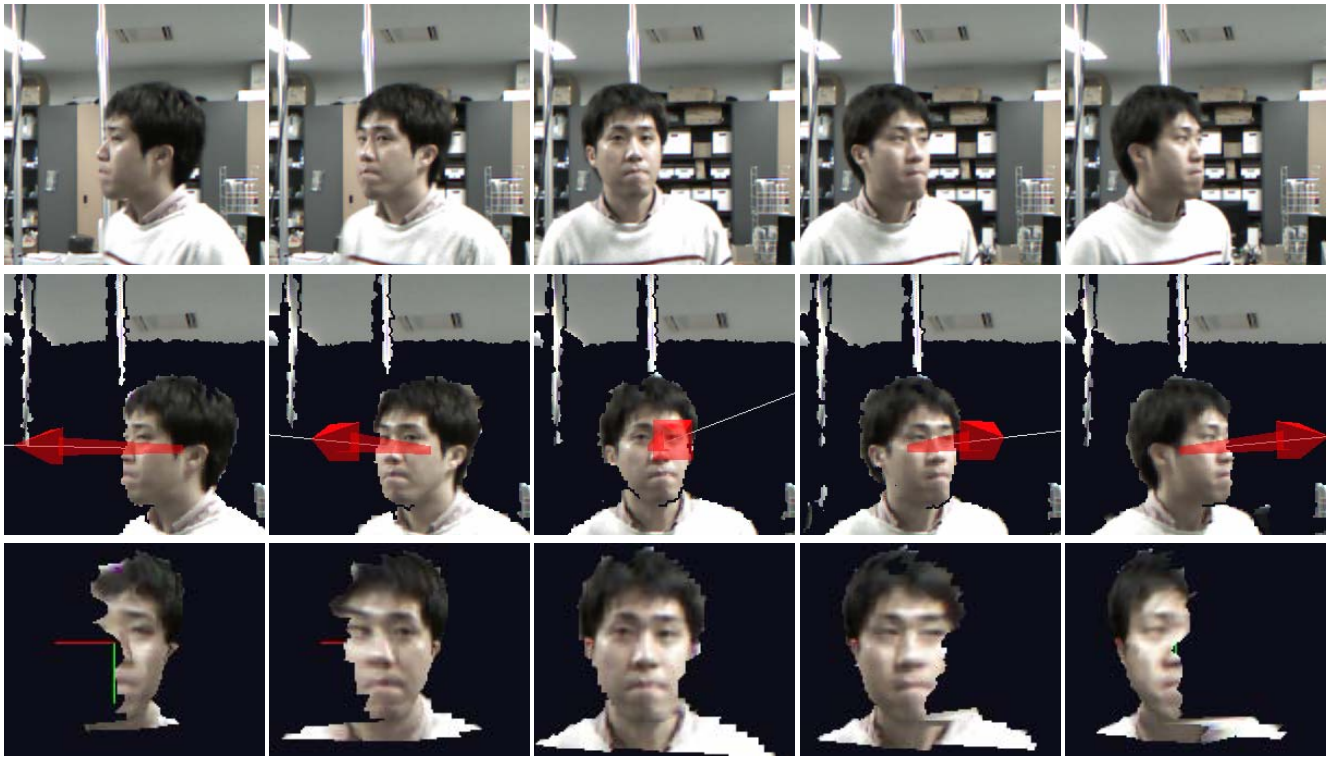
(b) Outlier fragments

Figure 6: Examples of sampled fragments: some of which are included in the inlier group (a), while others are included in the outlier group (b) and rejected. Each rejected fragments contains some artifacts, such as motion blurs or segmentation errors.

Using the algorithm, we also built an application that tracks the unknown human's head shape and pose and estimates where he/she is looking at the digital contents on the display. The looking area are summarized and visualized with a heat map in real-time. With the synthesized frontal portrait shown in Fig.5. The top rows are input images, the middle are annotated image by using arrow CG model and the bottom are synthesized frontal portrait image. This system can visualize the pair of attention area and corresponding the user's portrait in real-time. Its performance shows that our method will be grate helps for the online systems, such as digital signage one.

6. Conclusion and future Work

In this paper, we proposed a cubistic representation for simultaneous shape acquisition and pose acquisition for unknown rigid object. The most important contribution of this paper is using the set of fragments to describe the 3D sur-



Frame # 366 Frame # 369 Frame # 377 Frame # 381 Frame # 384

Figure 5: Head pose estimation results: (top) input images; (middle) the estimated head poses; (bottom) frontal portrait images synthesized from the pairs of the input image and the estimated pose parameter.

face shape of the target. It makes the online shape acquisition problem can be formulated as online shape sampling problem and solved with a particle filter based computation. We also showed the practicality and reliability of our cubistic representation with online system that detects and tracks the head of unknown human. For future work, we are planning to improve the algorithm by using recent MCMC-based particle filtering.

Acknowledgment

This work was supported in part by the JST-CREST project "Creation of Human-Harmonized Information Technology for Convivial Society."

References

[1] P. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb.

[2] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, Feb. 1992.

[3] R. A. Brooks. Symbolic reasoning among 3D models and 2D images. *AI Journal*, 17:285–348, 1981.

[4] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681 –685, jun 2001.

[5] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[6] W. Hu and S.-C. Zhu. Learning a probabilistic model mixing 3d and 2d primitives for view invariant object recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2273–2280, 2010.

[7] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.

[8] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM*

symposium on User interface software and technology, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM.

[9] Z. Kalal, J. Matas, and K. Mikolajczyk. Online learning of robust object detectors during unstable tracking. In *In Proceedings of the IEEE On-line Learning for Computer Vision Workshop*, pages 1417–1424, 2009.

[10] Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1805–1819, 2005.

[11] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *Proc. of Eigth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'09)*, Orlando, October 2009.

[12] K. Konolige and P. Mihelich. *Technical description of Kinect calibration*.

[13] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2):150–162, Feb. 1994.

[14] Y. Li, L. Gu, and T. Kanade. A robust shape model for multi-view car alignment. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2466–2473, 2009.

[15] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, Aug. 1987.

[16] D. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150–1157 vol.2, 1999.

[17] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artif. Intell.*, 31(3):355–395, Mar. 1987.

[18] D. Marr and H. K. Nishihara. Representation and Recognition of the Spatial Organization of Three-Dimensional Shapes. *Proceedings of the Royal Society of London Biological Sciences*, 200(1140):269–294, 1978.

[19] R. Newcombe, S. Lovegrove, and A. Davison. Dtam: Dense tracking and mapping in real-time. In *Proc. of IEEE International Conference on Computer Vision(ICCV)*, pages 2320 –2327, nov. 2011.

[20] J. Shi and C. Tomasi. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, pages 593–600, 1994.

[21] M. Sun, B.-X. Xu, G. Bradski, and S. Savarese. Depth-encoded hough voting for joint object detection and shape recovery. In *ECCV*, Crete, Greece, 09/2010 2010.

[22] C. Tomasi and T. Kanade. shape and motion from image streams: a factorization method. Technical report, International Journal of Computer Vision, 1991.

[23] P. Viola and M. Jones. Robust real-time object detection. In *International Journal of Computer Vision*, 2001.