2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

Real-time 3D Object Pose Estimation and Tracking for Natural Landmark Based Visual Servo

Changhyun Choi, Seung-Min Baek and Sukhan Lee, *Fellow Member, IEEE*

Abstract— **A real-time solution for estimating and tracking the 3D pose of a rigid object is presented for image-based visual servo with natural landmarks. The many state-of-the-art technologies that are available for recognizing the 3D pose of an object in a natural setting are not suitable for real-time servo due to their time lags. This paper demonstrates that a real-time solution of 3D pose estimation become feasible by combining a fast tracker such as KLT [7] [8] with a method of determining the 3D coordinates of tracking points on an object at the time of SIFT based tracking point initiation, assuming that a 3D geometric model with SIFT description of an object is known a-priori. Keeping track of tracking points with KLT, removing the tracking point outliers automatically, and reinitiating the tracking points using SIFT once deteriorated, the 3D pose of an object can be estimated and tracked in real-time. This method can be applied to both mono and stereo camera based 3D pose estimation and tracking. The former guarantees higher frame rates with about 1 ms of *local pose estimation*, while the latter assures of more precise pose results but with about 16 ms of *local pose estimation*. The experimental investigations have shown the effectiveness of the proposed approach with real-time performance.**

I. Introduction

In visual servo control, real-time performance of object recognition with pose has been regarded as one of the most important issues for several decades. Especially the literature has required three dimensional (3D) pose estimation in order to handle a target object in 3D space. While modern recognition methods, such as SIFT and structured light, which are applicable to accurate pose estimation have been recently proposed, these are still inappropriate to be applied to vision-based manipulation due to the high computational burden. For natural control of robot manipulators, we propose a real-time solution for tracking 3D rigid objects using a mono or stereo camera that combines scale invariant feature based matching [1] [2] with Kande-Lucas-Tomasi (KLT) tracker [6] [7] [8] [9]. Since there is no need for a stereo matching process, *mono mode* using a mono camera promises better computational performance. In contrast, *stereo mode* using a stereo camera shows more accurate pose results. We expect that *mono mode* can be applied when coarse pose results are accepted, for example, a robot approaches a target object. Conversely, when truthful pose are necessary, such as grasping a target object, *stereo mode* will be available.

This research was performed for the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs funded by the Ministry of Commerce, Industry and Energy of Korea. Also this work is supported in part by the Science and Technology Program of Gyeong-Gi Province as well as in part by the Sungkyunkwan University.

Changhyun Choi, Seung-Min Baek and Sukhan Lee are with the School of Information and Communication Engineering of Sungkyunkwan University, Suwon, Korea {cchoi,smbaek,lsh}@ece.skku.ac.kr

II. Related Work

Papanikolopoulos and Kanade proposed algorithms for robotic real-time visual tracking using sum-of-squared differences (SSD) optical flow [5]. Combining vision and control, the system feeds discrete displacements calculated from SSD optical flow to controllers or a discrete Kalman filter. Considering that the vision process was taken about 100 ms in 1993, this work showed the possibility of optical flow as a real-time tracker. However, the constraints, such as an assumption of given depth and manual selection of tracking points, used in the work are not acceptable to modern robots. For dependable pose tracking, the depth calculation and the tracking points generation should be fully automatic processes. Moreover, the system only considered roll angle of a target object. Visual servoing system should know not only 3D position (x, y, z) but also 3D orientation (roll, pitch, yaw) of it.

As CPU becomes faster for a decade, the tracking methods using local feature points are recently proposed. Vacchetti and Lepetit suggested a real-time solution, which does not have jitter or drift, for tracking objects in 3D by using standard corner features with a single camera [4]. Their approach is robust to large camera displacements, drastic aspect changes, and partial occlusions. Drawbacks of the system are that the camera should be close enough to one of keyframes which is a kind of prior knowledge. This limited starting condition makes practical limitations on pose tracking applications in visual servo control. For more robust tracking, tracking systems need a recognition scheme which is invariant to scale, rotation, affine distortion, and change in 3D viewpoint.

The work of Panin and Knoll aptly complemented the disadvantages of Vacchetti and Lepetit's work [3]. They proposed a fully automatic real-time solution for 3D object pose tracking with SIFT feature matching and contour-based object tracking. The solution is primarily organized into two parts: global initialization module and local curve fitting algorithm. The former recognizes the initial pose of the object on the whole incoming image through SIFT feature matching. The latter estimates pose of a target object through contour matching on an online color image stream with the calculated initial pose and a given contour model. The approach could track the pose of objects having insufficient texture on the surfaces since it uses the contour information of objects. But, when the colors of a target object and background are similar, it has potential risk of failing of pose tracking because *Contracting Curve Density* (CCD) al-

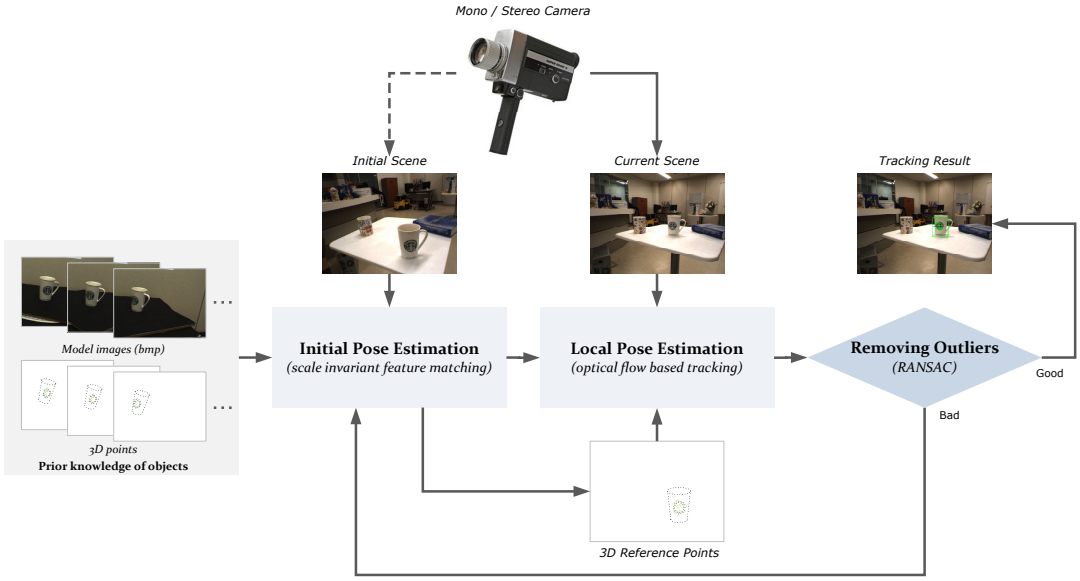


Fig. 1. System Architecture

gorithm used for object contour tracking principally relies on color statistics near the boundary of the object. In addition, there is rotational ambiguity on the pose results according to the shape of contour models such as cylindrical objects or spherical objects.

III. Proposed Approach

In order to overcome the disadvantages of the related works discussed above, we propose a method that combines scale invariant features matching with optical flow based tracking, KLT tracker, for real-time 3D pose tracking. Since SIFT features are invariant to scale, rotation, illumination, and partial change in 3D viewpoint, SIFT feature matching makes our system robustly estimate an initial pose of a target object. However, the speed of SIFT feature matching on modern CPU is not so fast enough to be utilized in robotic arm control[1]. To overcome this difficulty, our approach adopts KLT tracker to calculate 3D pose consecutively from initially estimated pose with cheap computational cost.

KLT tracker is already well established and its implementation is available in public computer vision library [10] [9]. The advantages of the KLT tracker are in that it is free from prior knowledge and computationally cheap. Like other area correlation-based methods, however, this tracker shows poor performance in significant change of scene, illumination, and occlusions. In addition, KLT tracking points are likely to drift due to the aperture problem of optical flow. Our system automatically remove outliers to get accurate pose results.

[1]It takes about 300 ms in our test with 640 × 480 RGB image to determine a 3D pose of an object.

A. Problem Definition: Mono and Stereo

Our method is applicable to both mono camera and stereo camera. Fig. 2 illustrates two camera settings and coordinate frames. $\{C\}$ and $\{C'\}$ represent camera coordinate frames, and $\{M\}$ describes model coordinate frame. Large crosses depict 3D model points with respect to the model coordinate, small crosses indicate 2D model points which are projected 3D model points on image plane S or S'. The problem of *mono mode* can be defined as,

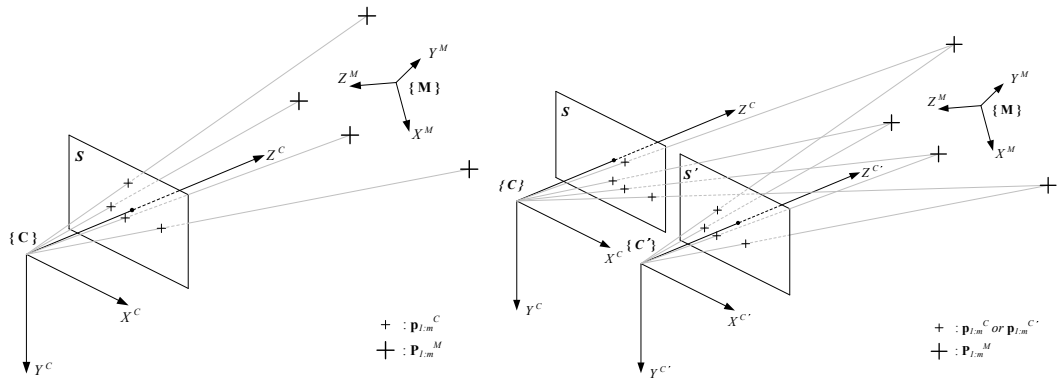
> *Given 2D-3D correspondences and a calibrated mono camera, find the pose of the object with respect to the camera.*

Similarly, the problem of *stereo mode* can be defined as,

> *Given 3D-3D correspondences and a calibrated stereo camera, find the pose of the object with respect to the camera.*

B. System Overview

As depicted in Fig. 1 the proposed method is mainly classified into two parts: *initial pose estimation* and *local pose estimation*. For SIFT feature matching, the system retains `BMP` images of a target object and 3D points calculated from structured light system as prior knowledge. The off-line information is utilized in the *initial pose estimation*. Once an initial pose is determined, the system automatically generates KLT tracking points around the matched SIFT points, and it saves 3D reference points for further use after calculating the 3D points of the tracking points. In the *local pose estimation*, the system quickly tracks 3D pose of the target



(a) (b)

Fig. 2. Camera coordinate frame $\{C\}$ and model coordinate frame $\{M\}$ with 3D model points. (a) Mono camera (b) Stereo camera

Algorithm 1 MONO 3D POSE TRACKING

Input: model images $I_{1:k}(x,y)$, 3D model points $J^M_{1:k}(x,y)$, a current scene image $S(x,y)$

Initialize:

1) Extract m SIFT feature points $\mathbf{p}^C_{1:m}$ globally on the $S(x,y)$ and get m 3D model points $\mathbf{P}^M_{1:m}$ corresponding to the extracted SIFT feature points by looking up $J^M_{1:k}(x,y)$
2) Determine the initial pose $\mathbf{\Phi}^C_0$ of the object by using POSIT algorithm in which $\mathbf{p}^C_{1:m}$ and $\mathbf{P}^M_{1:m}$ are served as parameters
3) Generate n 2D KLT tracking points $\mathbf{t}^C_{1:n}$ within the convex hull of a set of $\mathbf{p}^C_{1:m}$
4) Calculate n 3D reference points $\mathbf{T}^M_{1:n}$ corresponding to $\mathbf{t}^C_{1:n}$ by using the plane approximation based on $\mathbf{t}^C_{1:n}$, $\mathbf{P}^M_{1:m}$, and $\mathbf{\Phi}^C_0$

Loop:

1) Track $\mathbf{t}^C_{1:n}$ by using KLT tracker
2) Remove outliers by applying RANSAC to produce $\mathbf{t}^C_{1:n'}$ and $\mathbf{T}^M_{1:n'}$
3) If the number of remained tracking points n' is fewer than τ_{mono}, go to **Initialize**
4) Determine the current pose $\mathbf{\Phi}^C$ of the object by using POSIT algorithm in which $\mathbf{t}^C_{1:n'}$ and $\mathbf{T}^M_{1:n'}$ are served as parameters.

Output: the pose $\mathbf{\Phi}^C$ of the object

Algorithm 2 STEREO 3D POSE TRACKING

Input: model images $I_{1:k}(x,y)$, 3D model points $J^M_{1:k}(x,y)$, a current scene image $S(x,y)$ and $S'(x,y)$

Initialize:

1) Extract m SIFT feature points $\mathbf{p}^C_{1:m}$ globally on the $S(x,y)$
2) Generate n 2D KLT tracking points $\mathbf{t}^C_{1:n}$ within the convex hull of a set of $\mathbf{p}^C_{1:m}$
3) Determine n 3D reference points $\mathbf{T}^M_{1:n}$ corresponding to $\mathbf{t}^C_{1:n}$ by looking up $J^M_{1:k}(x,y)$
4) Calculate n 3D points $\mathbf{T}^C_{1:n}$ corresponding to $\mathbf{t}^C_{1:n}$ by stereo matching on $S(x,y)$ and $S'(x,y)$

Loop:

1) Track $\mathbf{t}^C_{1:n}$ by using KLT tracker
2) Remove outliers by applying RANSAC to produce $\mathbf{t}^C_{1:n'}$ and $\mathbf{T}^M_{1:n'}$
3) If the number of remained tracking points n' is fewer than τ_{stereo}, go to **Initialize**
4) Calculate $\mathbf{T}^C_{1:n'}$ by stereo matching
5) Determine the current pose $\mathbf{\Phi}^C$ of the object by closed-form solution using unit quaternions in which $\mathbf{T}^C_{1:n'}$ and $\mathbf{T}^M_{1:n'}$ are served as parameters.

Output: the pose $\mathbf{\Phi}^C$ of the object

object by using the 3D reference points and KLT tracking points. Outliers are eliminated by RANSAC algorithm until the number of remained tracking points is fewer than a certain threshold value. When the number of tracking points is fewer than the threshold, the system goes to the *initial pose estimation* and restarts SIFT matching globally, otherwise the system repeats the *local pose estimation*.

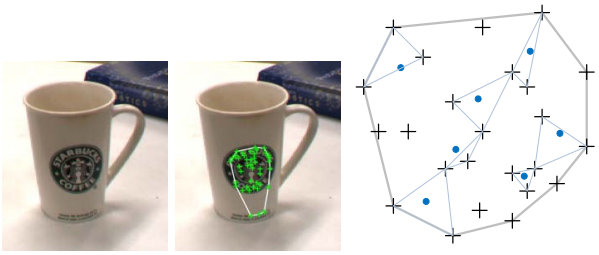
C. Algorithm

For clarity, Algorithm 1 and 2 are given to illustrate the core of our approach. In Algorithm 1 and 2, $I_{1:k}(x,y) = \{I_1(x,y), I_2(x,y), ..., I_k(x,y)\}$ are k RGB model images in which a target object is taken, and $J^M_{1:k}(x,y) = \{J^M_1(x,y), J^M_2(x,y), ..., J^M_k(x,y)\}$ represents the 3D points corresponding to each pixel (x,y) of the RGB model images with respect to the model coordinate frame $\{M\}$. $S(x,y)$ and $S'(x,y)$ denote the current input images. All images used are 640 $\times$ 480 resolution. $\mathbf{p}^C_{1:m} = \{\mathbf{p}^C_1, \mathbf{p}^C_2, ..., \mathbf{p}^C_m\}$, $\mathbf{t}^C_{1:n} = \{\mathbf{t}^C_1, \mathbf{t}^C_2, ..., \mathbf{t}^C_n\}$, $\mathbf{t}^C_{1:n'} = \{\mathbf{t}^C_1, \mathbf{t}^C_2, ..., \mathbf{t}^C_{n'}\}$, and $\mathbf{\Phi}^C$ are point vectors or pose vector of the object with respect to the camera coordinate frame $\{C\}$. Similarly $\mathbf{P}^M_{1:m}$, $\mathbf{T}^M_{1:n}$, and $\mathbf{T}^M_{1:n'}$ are point vectors in the model coordinate frame $\{M\}$.

Although the minimum number of corresponding points are 4 points in POSIT algorithm [13] and 3 points in the

closed-from solution using unit quaternions [14], the threshold for the minimum number of correspondences, τ_{mono} (≥ 4) and τ_{stereo} (≥ 3), should be adjusted according to the accuracy the application needs.

D. Generating Tracking Points



(a) (b) (c)

Fig. 3. The target object and the region for generating 2D KLT tracking points. The tracking points are only generated within the convex hull of a set of the matched SIFT points. In *mono mode*, for calculating the 3D point of each tracking point the algorithm uses three nearest neighboring matched SIFT points. (a) Target object (b) Matched SIFT points and the convex hull of a set of them on the surface of the target object (c) 2D KLT tracking points (bold points) within the convex hull

To track a moving 3D object by using KLT tracker, we have to generate tracking points on the area of the target object. We employ the matched SIFT points used in the *initial pose estimation* for allocating tracking points on the region of the target object. As shown in Fig. 3 (b), our algorithm generates the tracking points only inside the convex hull of a set of the matched SIFT points with the criteria proposed by Jianbo Shi and Carlo Tomasi [8]. In addition to these criteria, in *stereo mode* we regard the tracking points that have 3D points by the stereo matching as good features to 3D track.

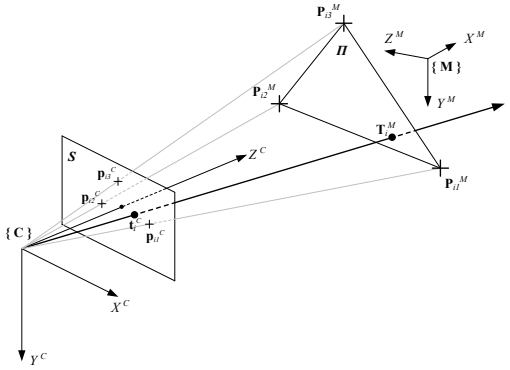


Fig. 4. Plane Approximation. A 3D point $\mathbf{T}_i^M$ corresponding to $\mathbf{t}_i^C$ on the image plane S is determined from the 3D plane Π that three 3D points $\mathbf{P}_{i_1}^M, \mathbf{P}_{i_2}^M, \mathbf{P}_{i_3}^M$ of the nearest neighboring matched SIFT points form.

When the algorithm generates tracking points, it also determines 3D reference points and saves them for the subsequent pose estimation. In *stereo mode* we directly calculate the 3D points by the stereo matching in which Sum of Absolute Differences (SAD) stereo correlation is used. Alternatively, in *mono mode*, we trickily determine 3D points from the approximation with the 3D points of prior knowledge. The key idea of the approximation is to treat the surface as locally flat. Fig. 3 (c) and Fig. 4 illustrate the approximation. In Fig. 3 (c), bold points indicate 2D KLT tracking points, and crosses represent the matched SIFT points. Since we already know the 3D points of the matched SIFT points from prior knowledge, each 3D point of the tracking point can be estimated under the assumption that the 3D tracking point lies on the 3D plane the 3D points of the three nearest neighboring matched SIFT points make.

Fig. 4 represents a 2D KLT tracking point $\mathbf{t}_i^C$, its 3D point $\mathbf{T}_i^M$ and 3D points $\mathbf{P}_{i_1}^M, \mathbf{P}_{i_2}^M, \mathbf{P}_{i_3}^M$ of the three nearest neighboring matched SIFT points with coordinate frames, where the subscripts $_{i_1}$, $_{i_2}$, and $_{i_3}$ are the selected indices from $1, 2, ..., m$. First, the equation of plane Π can be written as

$$ax + by + cz + d = 0 \tag{1}$$

Since our goal is determining the 3D point $\mathbf{T}_i^M$ corresponding to $\mathbf{t}_i^C$, we start from estimating the plane equation Eq. 1. The normal vector $\mathbf{n}$ of the plane can be determined as

$$\mathbf{n} = (\mathbf{P}_{i_2}^M - \mathbf{P}_{i_1}^M) \times (\mathbf{P}_{i_3}^M - \mathbf{P}_{i_1}^M) \tag{2}$$

where $\mathbf{n} = (a, b, c)^T$. The remained coefficient d of the plane equation is then expressed as

$$d = -\mathbf{n} \cdot \mathbf{P}_{i_1}^M \tag{3}$$

To scrutinize the relationship between the 2D point $\mathbf{t}_i^C$ and the 3D point $\mathbf{T}_i^M$ of the KLT tracking point, let's consider in perspective projection. Note that $\mathbf{T}_i^M$ is in the model coordinate frame, so we have to transform $\mathbf{T}_i^M$ into $\mathbf{T}_i^C$ according to the initial pose $\mathbf{\Phi}_0^C$. Using the homogeneous representation of those points, a linear projection equation can be written as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \cong \begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{4}$$

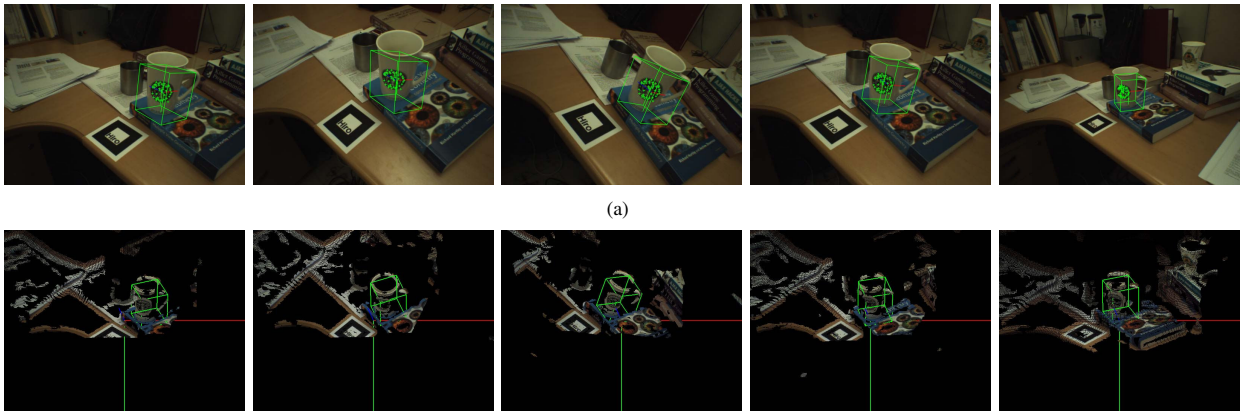
where f_x and f_y represent the focal length (in pixels) in the x and y direction respectively, $\mathbf{T}_i^C = (x, y, z)^T$, $\mathbf{t}_i^C = (u, v)^T$, and c_x and c_y are the principal point. Since our current goal is determining the 3D point $\mathbf{T}_i^C$ corresponding to $\mathbf{t}_i^C$, Eq. 4 can be expressed as (x, y, z) with respect to (u, v)

$$\begin{aligned} u &= \frac{wu}{w} = \frac{f_x x + c_x z}{z} = f_x \frac{x}{z} + c_x \\ v &= \frac{wv}{w} = \frac{f_y y + c_y z}{z} = f_y \frac{y}{z} + c_y \end{aligned} \tag{5}$$

$$\begin{aligned} \frac{x}{z} &= \frac{1}{f_x}(u - c_x) \\ \frac{y}{z} &= \frac{1}{f_y}(v - c_y) \end{aligned} \tag{6}$$

If Eq. 1 is divided by z then the equation becomes

$$a\frac{x}{z} + b\frac{y}{z} + c + \frac{d}{z} = 0 \tag{7}$$



(a)

(b)

Fig. 5. Pose tracking results of the experiment. The results of pose and KLT tracking points are represented on the sequence of input images. Pose is depicted as a green rectangular parallelepiped. (a) Tracking results of *mono mode* (b) Tracking results of *stereo mode* (From left to right) 2nd frame, 69th frame, 136th frame, 203rd frame, and 270th frame.

Therefore, by applying Eq. 6 to Eq. 7 and developing, we can get

$$z = \frac{d}{\frac{a}{f_x}(c_x - u) + \frac{b}{f_y}(c_y - v) - c}$$
$$x = \frac{z}{f_x}(u - c_x) \tag{8}$$
$$y = \frac{z}{f_y}(v - c_y)$$

With Eq. 8, we can determine $\mathbf{T}^C_{1:n}$ and reversely transform $\mathbf{T}^C_{1:n}$ into $\mathbf{T}^M_{1:n}$ according to $\mathbf{\Phi}^C_0$. Therefore, we can get the approximate 3D points $\mathbf{T}^M_{1:n}$ corresponding to the 2D KLT tracking points $\mathbf{t}^C_{1:n}$.

E. Outlier Handling

The KLT tracking points have a tendency to drift due to abrupt illumination change, occlusion, and aperture problem. We regard drifting points as outliers, and try to eliminate the outliers because they influence the accuracy of resulting poses. We adopt RANSAC algorithm [11] to remove the outliers during tracking. If the number of remained tracking points is fewer than a certain number, τ_{mono} and τ_{stereo} in *mono mode* and *stereo mode* respectively, the algorithm goes back to the *initial pose estimation* and restarts the SIFT feature matching.

IV. Experiment

In this section, we present an experimental result that shows abilities of our method to track a 3D object. As shown in Fig. 3 (a), the target object was a white mug that has partial texture on the side surface, and the prior knowledge of it was prepared in advance. The system has been implemented in C/C++ using OpenCV and OpenGL libraries on a modern PC (Pentium IV 2.4GHz) and a Bumblebee® stereo camera of the Point Grey Research Inc with the Triclops™Stereo SDK[2]. We used the stereo camera as *mono mode* or *stereo mode*. Since we needed ground truth, real trajectories of 3D pose, to evaluate tracking results, we used an ARToolkit marker[3]. After estimating the pose of the marker we transformed the pose data from the marker coordinate system to the object coordinate system.

A. Tracking Results

Fig. 5 shows pose tracking results of the object, and the trajectories are plotted with ground truth in Fig. 6. Note that the positions with respect to x and y are well tracked in both modes, while the position of z has some errors with respect to the ground truth. In rotation, *mono mode* shows bigger errors than *stereo mode* especially in pitch and yaw angles. Table. I describes RMS errors for all frames of input images. In *mode mode* the maximum positional errors are within about 3 cm, and the maximum rotational errors are inside approximately 16 degrees. In *stereo mode* the maximum positional errors are within about 2 cm, and the maximum rotational errors are inside around 6 degrees.

TABLE I

RMS ERRORS OVER THE WHOLE SEQUENCE OF IMAGE.

	Mono mode	**Stereo mode**
X (cm)	1.04	0.88
Y (cm)	1.74	1.39
Y (cm)	3.10	2.02
Roll (deg)	5.50	4.78
Pitch (deg)	15.83	5.23
Yaw (deg)	16.22	5.87

B. Real-time Performance

In order to show our method is capable of being a real-time solution, we present the computation times of the experiment. Fig. 7 demonstrates the computation times of the proposed

[2]http://www.ptgrey.com/products/stereo.asp

[3]http://www.hitl.washington.edu/artoolkit/

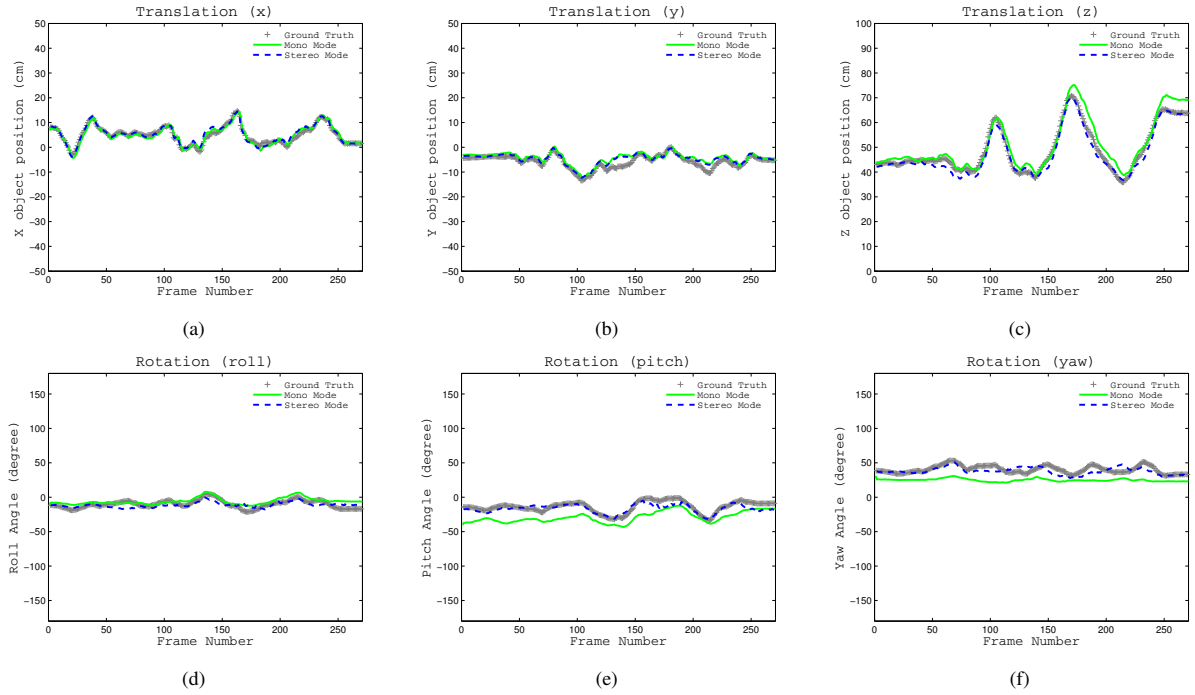


Fig. 6. Estimated trajectories of the object against ground truth. (a) X position (b) Y position (c) Z position (d) Roll angle (e) Pitch angle (f) Yaw angle

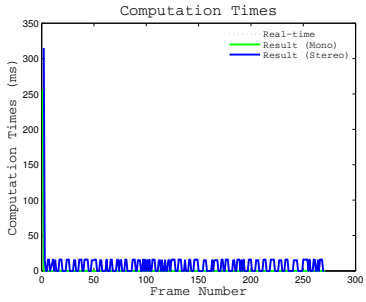


Fig. 7. Computation times of the proposed algorithm on the experiment.

algorithm. The experiment takes below average 1 ms per frame in *mono mode* and average 16 ms per frame in *stereo mode* on a modern PC. The peak at the initial frame is caused by SIFT feature extraction and matching.

TABLE II

COMPUTATION TIMES OF OUR ALGORITHM ON A MODERN PC, PENTIUM IV 2.4GHZ, ON 640 × 480 IMAGES.

	Mono mode	**Stereo mode**
Image(s) acquisition	35 ms	70 ms
Stereo matching	-	210 ms
Initial pose estimation (SIFT)	300 ms	300 ms
Local pose estimation (KLT)	1 ms	16 ms

Table. II describes the computational cost of each part of the algorithm. According to the table, we know that the *initial pose estimation* in *mono mode* and the *initial pose estimation* with the stereo matching in *stereo mode* are bottlenecks in our system. Since the current stereo matching calculates all 3D points on each input frame, it takes amount of computation. We expect a significant improvement of computation time in the stereo matching if it calculates 3D points only on the 2D KLT tracking points.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a method for tracking 3D roto-translation of rigid objects using scale invariant feature based matching and Kande-Lucas-Tomasi (KLT) tracker. The method has two mode, *mono mode* using a mono camera and *stereo mode* using a stereo camera. *Mono mode* guarantees higher frame rate performance, and *stereo mode* shows better pose results. In the *initial pose estimation*, our system used prior knowledge of target objects for SIFT feature matching. In addition, to be used in the *local pose estimation*, 3D reference points are calculated by the plane approximation in *mono mode* and the stereo matching in *stereo mode*.

Future work will be mainly focused on decreasing the computational burden of the *initial pose estimation*. Our approach using KLT tracker has to do SIFT feature extraction and matching, which needs amount of computation, again if the KLT tracking points are disappeared or drifted. This can be further improved by unifying the contour based tracking in order to generate KLT tracking points again inside the tracking contour without SIFT feature matching. Moreover, recent GPU-based implementation of KLT tracker and SIFT, GPU_KLT and SiftGPU, respectively, will enhance the frame rate of our method significantly.

VI. ACKNOWLEDGMENTS

The authors would like to thank KyeongKeun Baek for his kind help to make 3D mug models as well as Daesik Kim for his useful comments to devise the algorithm presented in this paper.

REFERENCES

[1] D. Lowe. (1999) Object recognition from local scale invariant features. *In Proc. 7th International Conf. Computer Vision (ICCV99)*, pages 1150–1157, Kerkyra, Greece

[2] D. G. Lowe, Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[3] G. Panin, A. Knoll, Fully Automatic Real-Time 3D Object Tracking using Active Contour and Appearance Models, *Journal of Multimedia 2006*, Academic Publisher, vol. 1 n. 7, pp. 62-70

[4] Vacchetti, L.; Lepetit, V.; Fua, P., Stable real-time 3D tracking using online and offline information, *Transactions on Pattern Analysis and Machine Intelligence* , vol.26, no.10, pp. 1385-1391, Oct. 2004

[5] Papanikolopoulos, N.P.; Khosla, P.K.; Kanade, T., Visual tracking of a moving target by a camera mounted on a robot: a combination of control and vision, *Robotics and Automation, IEEE Transactions on* , vol.9, no.1, pp.14-35, Feb 1993

[6] B. Lucas and T. Kanade, An iterative image registration technique with an application to stereo vision, in *International Joint Conference on Artificial Intelligence*, pp. 674679, 1981.

[7] C. Tomasi and T. Kanade, Detection and Tracking of Point Features, Technical Report CMU-CS-91-132, Carnegie Mellon University, April 1991.

[8] Jianbo Shi; Tomasi, C., Good features to track, *Computer Vision and Pattern Recognition, 1994*. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on , vol., no., pp.593-600, 21-23 Jun 1994.

[9] J.-Y. Bouguet, Pyramidal Implementation of the Lucas Kanade Feature Tracker: Description of the algorithm, Technical Report, Intel Corporation, Microprocessor Research Labs 2000, OpenCV documentation.

[10] Intel, Open Source Computer Vision Library, http://www.intel.com/research/mrl/research/opencv/.

[11] M. Fischler and R. Bolles, Random sampling consensus: a paradigm for model fitting with application to image analysis and automated cartography, *Commun. Assoc. Comp. Mach.*, vol. 24, pp. 381395, 1981.

[12] Sudipta N Sinha, Jan-Michael Frahm, Marc Pollefeys and Yakup Genc, GPU-Based Video Feature Tracking and Matching, EDGE 2006, *workshop on Edge Computing Using New Commodity Architectures*, Chapel Hill, May 2006.

[13] D. DeMenthon and L.S. Davis, Model-Based Object Pose in 25 Lines of Code, *International Journal of Computer Vision*, 15, pp. 123-141, June 1995.

[14] Horn, Berthold K. P., Closed-Form Solution of Absolute Orientation Using Unit Quaternions, *Journal of the Optical Society of America A 4*, No. 4, April 1987, pp. 629-642.