The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Interoperable RT Component for Object Detection and 3D Pose Estimation for Service Robots

Jaeil Choi, Hideyasu Takahashi, Yasushi Mae, Kenichi Ohara, Tomohito Takubo, Tatsuo Arai
Division of Systems Science and Applied Informatics, Osaka University, Japan

Abstract— **Finding objects and tracking their poses are essential functions for service robots, in order to manipulate objects and interact with humans. We present an approach for object detection and 3D pose estimation for autonomous mobile robots, that is suitable for general uses in a modularized robot control system. Our apprach extracts local features from the input images, searches for the reference pattern, and then produces the 3D pose in camera coordinate system, using only a single reference image and the 6-DOF pose in it. We have created an RT(Robot Technology) component that can be used in any RT-based system, and developed an algorithm that can extend the range of detection and produce robust pose estimation. For evaluation, we have integrated our vision component in an autonomous robot system with a search-and-grasp task, and tested it with several objects that are found in ordinary domestic environment. We present the details of our approach, the design of our modular component design, and the results of the experiments in this paper.**

I. INTRODUCTION

In order to manipulate objects and interact with humans more intelligently, service robots need to find objects and estimate their three-dimensional poses. Although we may tag each object with RFID and other types of equipments and sensors [1], it is highly desirable to use vision as the primary sensing mode, because of availability of small, inexpensive, and low-powered cameras and non-intrusive nature of the vision, without any alteration of the environment.

Object tracking has been an active research area, and most of techniques can be divided into two classes depending on the types of cues: edge-based and texture-based. The former techniques rely on spatial gradients outlining the contour or some geometric features, and frequently used for pose estimation problem with 2D or 3D model of the object [2] [3]. The latter techniques depends on local features (corners, gradient characteristics, or a template) [4] [5] [6], and they are less subject to jitter and better suited for dealing with initial registration and large displacements. These two classes of techniques have complementary advantages and drawbacks. Edge-based techniques have been proved to be effective and robust with 2D and 3D model of the object, but it may fail in the presence of complex background. Initial registration has also been an issue. Texture-based techniques on the other hand are restricted to a class of objects with strong texture, and mainly exploited in 2D tracking, such as KLT tracker [4] and region of interest [7], due to the lack of precision when the scale changes. Some researches have been focused on combining the power of both texture-based and edge-based techniques [8] [9].

Fig. 1. A shot from an on-line experiment with object detection and pose estimation on our robot platform, Enon. The robot was about to grab the object after estimating its pose successfully.

Recent progresses in robust local features, such as SIFT (Scale Invariant Feature Transform) [10], helped a lot to resolve many issues with feature-based approaches, including robustness, precision, invariance in scale, and initial registration. With robust and scale-invariant features, collective matches can successfully be used for global detection and pose estimation. Two main limitations have been the speed and the range of applicable object types, which is common for texture-based techniques. Thus we have seen SIFT features used in many researches for global registration in initialization or intermediate steps [11]. We'd like to note that, with fast-developing GPU(Graphics Processing Unit)-based implementation of local feature extraction, full SIFT extraction and matching can be run in real-time, without having to trade off in terms of feature quality. In addition, for the vision system on service robots, the lack of required object model and global registration are big advantages for more general and practical use.

Another important aspect of object detection and pose estimation for service robots is the performance in the practical settings, which we found quite challenging. When the scene is observed by the robot, the target object is relatively small especially from a wide angle camera, the images are often blurred by motion, the texture of the target object may not be strong enough, and background is usually complex. Even though SIFT features were designed so that they could provide robustness, repeatability, and uniqueness in the feature space, careful matching can improve the matching results further in challenging situations. In this paper, we also present a new approach for feature matching scheme that enforces local geometric consistency, and simple pose estimation algorithm that do not require 3D model of the target object.

System integration of basic robot functions has become more important as individual technologies in Robotics are getting mature and academic researches and industry move toward robotic systems in non-manufacturing fields. Ando *et al.* [15] introduced RT, an open and event-based robot integration platform based on CORBA. This platform, like other platforms with similar goals including Orocos [12], ORiN [13], ORCA [14] and Microsoft Robotics Studio, provides software-level modularization and concurrent execution of robot functions in a distributed system, regardless of operating system and programming language. On this platform, the task of developing a complex robotic system is reduced to identifying compatible modules with desired functions, and binding them together to achieve high-level control. The product of our project is an on-line vision component for object detection and 3D pose estimation with general input/output specification, which can be used in any robot system that is based on RT-middleware, without modification.

In this paper, we explain our system design based on RT-middleware in Section II, and then give more details of the object detection and 3D pose estimation in our vision component in Section III. And then we show the results from our experiments in Section IV. Finally, in Section V, we discuss the strength and shortcomings of our approach – local feature based object detection and 3D pose estimation.

II. DESIGN OF VISION MODULE AS A RT COMPONENT

The goal of our project is to create a module-based control system for a autonomous service robot that can search and find small objects, manipulate the object, and interact with humans (by receiving or handing over the object, for example). Figure 1 shows our robot platform, Enon from Fujitsu Frontech Ltd., with a Bumblebee2 stereo camera from PointGrey Research mounted on its head, while it's trying to grab the target object after pose estimation.

Our system design is based on RT(Robot Technology)-middleware, which is a distributed platform that provides re-usability and interoperability of different components that encapsulate different functions across operating systems or programming languages through standardized interface [15]. The vision component encapsulates object detection and pose estimation, provides general interface for integration, and can be used without modification on any RT-based system. Figure 2 illustrates our component design.

Our experimental system consists of four components, designed around the vision component for object detection and pose estimation. 'Bumblebee' camera component is one of the image capture components, which provides a sequence of images in a simple minimalistic image format (width, height, pixel format, and pixel data). 'Enon Robot Controller' component is the controller component of the robot Enon. This component provides all the basic control of Enon such as arm control, vehicle control, speech, pan and tilt of head, and so on. By combining these components, user can realize the target object manipulation based on user's scenario.

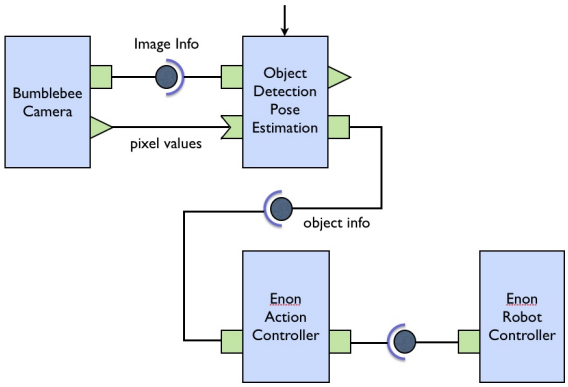

Fig. 2. Our system design with RT-components

Finally, 'Enon Action Controller' component is a component for pose conversion and higher-level logic. This component maintains an action scenario, and decides what to do next based on detection results from the vision component and the robot's current state.

Input of the vision component is an image in the same format with the camera components. Output of the vision component is the pose of the object, represented by a 3×3 rotation matrix and a 3×1 translation vector in camera coordinate system, and a success/failure flag. Alternatively, the vision component also provide a *service port*, which is an equivalent of a synchronized function call in RT-middleware framework, that can be used to invoke the detection and get the pose of the target in return. Table I explains the port types of the vision component.

TABLE I
INTERFACE OF THE VISION RT COMPONENT

interface	port type	description
image info	ConsumerPort	width, height, pixel format
pixel values	InPort	image stream
object info	ProviderPort	search result and object pose

The interface of the vision component was designed with simple specifications of minimal number of ports, in order to improve re-usability and interoperability of the component. The vision component is provided with a reference image of the target object, and its corresponding pose (6 degrees of freedom) in camera coordinate system when the image was taken. This simple prior knowledge is loaded at the start-up of the component. This simplified requirements also make it easier to use the vision component for many other objects. We explain in the next section the algorithms of the vision component in detail.

III. OBJECT DETECTION & POSE ESTIMATION

SIFT features are invariant against scale and rotation, and robust against modest viewpoint changes, illumination changes, and partial occlusions. [11] and other researches have already used these robust local features to detect and

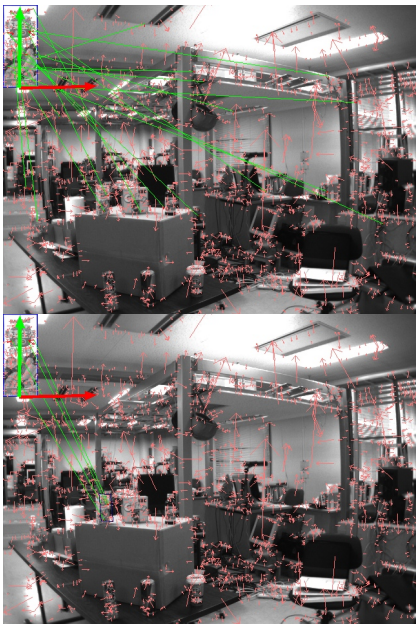

Fig. 3. A challenging scene from a robot in indoor environment. Extracted SIFT features are illustrated as red arrows, and the matching results are shown with green lines. Upper image is the result of global matching, with distance-ratio-criteria set to 0.8, and below is the result of local matching with geometric consistency described in Section III-B

track object and its pose. We also use SIFT features for object detection and pose estimation for our service robot.

The challenge is that, in typical domestic environments, matching and tracking those local features from the viewpoint of a mobile robot is very difficult in practice, because the size of target objects in the scene is relatively small, their texture may not be strong enough, the image is often blurred by motion, and there are a lot of noise due to complex background patterns. Extraction of the local features is also computationally expensive. For example, when we consider more general and practical situations like the one shown in Fig. 3, it is clear that naive approaches which depend only on the uniqueness of individual SIFT descriptors will not produce the result that we want. We also need a stronger, more robust, and more efficient approach for matching and tracking of the local features. In following sections, we explain the details of our approach to local feature matching that exploits local geometric consistency, and 3D pose estimation that requires minimal prior knowledge (six values for reference pose) using planar surface assumption.

A. Local Feature Extraction

For SIFT feature extraction, we used SIFT++ [17], an open-source implementation that was shown to produce as good results as David Lowe's original implementation in terms of repeatability, robustness, and uniqueness of the features [16]. We also have implemented SIFT extraction on GPU using NVIDIA's CUDA framework, using the same procedures and parameters of SIFT++. With initial upsampling (doubling image width and height) of the input image, SIFT++ processes a single 640×480 RGB image in about 3 seconds on 2.4 GHz Intel Core2Duo laptop, and GPU-based SIFT processes the same image in about 250 milliseconds with NVIDIA's GeForce 8600M GT on the same machine, making on-line applications possible.

B. Matching Local Features

Typical approach for SIFT feature matching is to find 'good' matches independently, and use RANSAC algorithm or Hough transform [10] to find the most probable transformation from the given set of matches. Goodness of a match between two features can be evaluated by the ratio of the distances to the matching feature (the closest) and the second closest in the 128 dimensional vector space of SIFT features.

We note that there are three properties of each local feature that we can use for geometric consistency: scale, orientation, and relative position. These values are also calculated for each local feature (keypoint) during the extraction of invariant feature vector, but are not a part of feature descriptor. Hough transform described in [10] utilizes these values by creating bins of different scale and orientations, but relative position is still not used at all. In general, typical approach try to find good matches in entire image first, and then find a consistent pose transformation using clustering or binning. We found matching can be improved significantly, by creating multiple pose hypotheses and find the best hypothes by evaluating their overall fitness and the geometric consistency of the pattern in the local region. This approach helps particularly when we deal with multiple instances of the same pattern or symmetric patterns, because the best match is not excluded by the distance ratio criteria when there are similar features with different orientations at different relative location on the image.

Our object detection component extracts local features, and creates a 2D map of pointers to the features, with the same width and height of the input image, in order to speed up the local search. And then, every feature in the reference pattern is compared with every feature in the scene to find global matches, with the same distance-ratio threshold described above (typically 0.7 or 0.8). The scale and orientation of each match in this global comparison is used for the initial hypothesis of pose transformation. Let the affine transformation of k-th match be T^k, and $f_r^k = (\mathbf{x}, o, s)_r^k$ and $f_s^k = (\mathbf{x}, o, s)_s^k$ be features (with position $\mathbf{x}$, orientation o, and scale s) in the reference and the scene of the match, respectively. Then,

$$T(\mathbf{x}) = \begin{pmatrix} +s'\cos(o') & -s'\sin(o') \\ +s'\sin(o') & +s'\cos(o') \end{pmatrix} (\mathbf{x} - \mathbf{c}_r) + (\mathbf{c}_s - \mathbf{c}_r), \quad (1)$$

$$f_s^k = (\mathbf{x}, o, s)_s^k \simeq T^k((\mathbf{x}, o, s)_r^k) = T^k(f_r^k), \quad (2)$$

where s' and o' are scale and orientation differences, $\mathbf{c}$ is the center of the pattern. Now with a initial hypothesis of a particular match, matches for all the reference features are searched over the local area around the position $\mathbf{x}$ in $(\mathbf{x}, o)_s^k$. For this local search, we set maximum image distance to fairly large value (two times the expected value). We also use fairly generous tolerance range of orientation,

$[-\pi/3, +\pi/3]$ and scale, $[-s/2, +s/2]$, and rather strict distance-ratio criteria 0.5. All these parameter choices means that we take a quite generous approach in finding local matches. And then we choose the closest neighbor in the multi-dimensional descriptor space as the best match, among all the valid ones that passed through all the criteria. Based on the local matches of T^k, we now evaluate the overall similarity of the entire pattern with an evaluation function given as follows:

$$\mathcal{E}(T^k) = \frac{1}{N} \sum^{N} D_i(\mathbf{x}_s, T^k(\mathbf{x}_r)) * D_d(f_s, f_r), \quad (3)$$

where D_i represents L_2 distance in 2D image space, D_d represent L_2 distance in descriptor space of local features (128 multi-dimensional space for SIFT), and N is the number of found local matches for T^k. In general, there might be multiple instances of the same pattern in the scene. Since we evaluate multiple pose hypotheses based on their local matches, it is easy to extend our approach for the matching of multiple instances. At the moment, we just take the best match.

C. 3D Pose Estimation

With the matches found between local features from the reference image and the scene, we can now estimate the 3D pose of the target object, assuming the 3D pose of the object in the reference image is already known.

First, we calculate the homography [18], H, of the object pattern from the reference image to the scene. Then we calculate the pose of the target object R^s and T^s in the current scene using pin hole camera model, assuming we already know the intrinsic parameters K of the camera and the pose of the object in the reference frame R^r and T^r. Here the pose, R and T, are represented as 3×3 rotation matrix and the 3×1 translation vector, even though the pose has only six degrees of freedom.

The pinhole model of the camera is given by:

$$\mathbf{p}^i \quad = \quad K\ [R\ T]\ \mathbf{p}^w, \quad (4)$$

where $\mathbf{p}^i = (u, v, 1)^T$ refers to the homogeneous coordinate of the point on the image frame, and $\mathbf{p}^w = (x, y, z, 1)^T$ is the homogeneous coordinate of the point in the world frame, and K is 3×3 matrix with the intrinsic parameters of the camera.

Since the points in the reference image are matched with the points in the scene images using homography, we get

$$\mathbf{p}^s \quad = \quad k\, H\, \mathbf{p}^r, \quad (5)$$

$$K\,[R^s\ T^s] \quad = \quad k\, H\, K\,[R^r\ T^r], \quad (6)$$

where $\mathbf{p}^s$, R^s, and T^s refer to the point and pose of the object in the scene image, and $\mathbf{p}^r$, R^r, and T^r are the point and pose of the object in the reference image. From the equation above, we can calculate the object pose in the scene image, given the intrinsic parameters and the pose in the reference image.

$$[{}^cR_o^s\ {}^cT_o^s] = k\, K^{-1}\, H\, K\, [{}^cR_o^r\ {}^cT_o^r]. \quad (7)$$

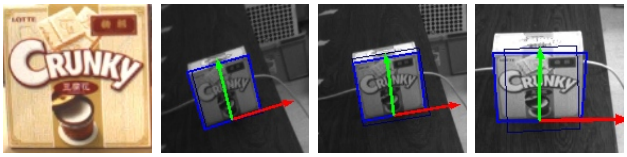

Fig. 4. The results of the 3D pose estimation: The leftmost image is the reference image. The 3D pose was visualized with two axes (x: red, z: green) of the object coordinate system centered at the object bottom center.

However, equation (7) is ill-conditioned, and it's not very robust in practice. It's because we are solving for 12 unknowns (${}^cR_o^s[9]$ and ${}^cT_o^s[3]$) from 8 DOF information. As a result, the rotation matrix ${}^cR_o^s$ in return will not typically be orthogonal, and ${}^cT_o^s$ tends to have some errors.

To resolve this problem, we have added a non-linear optimization step. First, we enforce the orthogonality of ${}^cR_o^s$, and convert the rotation matrix to a rotation vector of 3 values. In order to enforce the orthogonality, we use SVD, and replace ${}^cR_o^s = U\,S\,V^T$ with $U\,V^T$. Through experiments, we found the singular values in S vary in the range of $[0.3, 2.0]$, and that the new ${}^cR_o^s$ and ${}^cT_o$ is good enough as an initial value. And then, we use Levenberg-Marquardt method so that it minimize the sum of the squares of the deviations of the four corners of the pattern:

$$S(R, T) \quad = \quad \sum_{i=1}^{4} [\mathbf{y}_i - f(\mathbf{x}_i, R, T)]^2,$$

where $\mathbf{x}_i$ and $\mathbf{y}_i$ are the four corners of the object pattern boundary in the reference (3D) and the input (2D) images, respectively, and f is the perspective transformation from camera space to the image space given by the calibration of the camera. Through many experiments, we found a small number of iterations is enough for this final optimization, and we set it to 50 in all the experiments in this paper. An examples of the final results of pose estimation are shown in Figure 4. We also perform a sanity check of the estimated pose as a final step, by checking if the object position is indeed in the observable area in front of the camera, within the distance range of $[0.2, 10.0]$ in meters.

The biggest benefit of this approach is that it does not require the explicit model of the target object, thus simplifying the process of constructing the reference pattern for object detection. The system only needs the reference image of the object, and the pose of the object (six DOF) in the image in camera coordinate system. This simple requirements can make it much easier for the robot to create a reference pattern of the target object on the fly, while interacting with humans. On the other hand, our approach also has a weakness in that it assumes planar surface for the pattern. As a result, the result of matching can be worsen if the target is viewed from a different viewpoint. But we note that round surfaces with big viewpoint change can cause problems with local feature matching at the first place, thus we don't consider it to be a big disadvantage.

Our vision component uses both images from the stereo independently, and then combines the two estimations of the

3D pose in order to improve the quality and robustness of the pose estimation. We simply take the average of positions and orientations represented in rotation vector from the left and right image.

IV. EXPERIMENTS

In order to evaluate the performance and robustness of our approach on a mobile robot in realistic environment, we have conducted series of experiments with various objects using an on-line autonomous robot. We used Enon from Fujitsu together with our autonomous control system that is based on RT-middleware as described in Section II. Figure 5 shows the estimated poses rendered on the input images with a rather cluttered background.

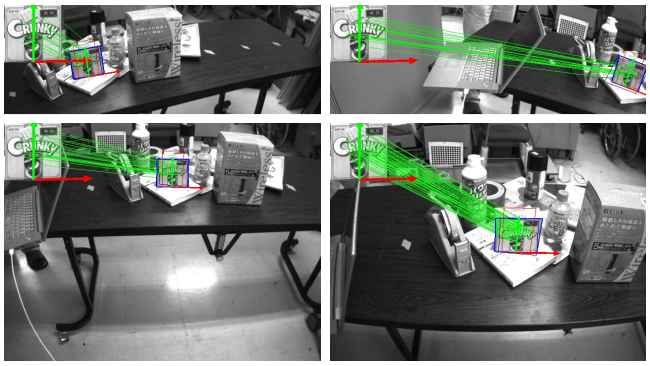

Fig. 5. Pose estimation results while robot is searching and approaching toward the target

Figure 6 shows some captured images during an on-line experiment with our autonomous robot system. The robot was assigned a sear-and-grasp task with given reference pattern and corresponding pose. The robot was programmed to search over the scene for a particular object, approach to the object, align its pose against the object for manipulation, and then grab the object with its hand. Through out the run, the vision component successfully recovered the 3D pose of the target while the robot moved around, and allowed the manipulation module to grab the object without any sophisticated feedback control.

In further analysis of the estimated poses in several experiments, the results are very promising in terms of precision and stability, that we can even use the estimated 3D pose from vision component as direct input parameters of the arm control, leading to successful grasp in many cases. Of course, manipulating objects relying only on frame by frame pose estimation is not a good idea, and the precision in manipulation can be improved significantly by using a real-time feedback control such as visual servo. But it is clear that the results from our pose estimation can be used as a reliable starting point for manipulation control.

The next experiment we performed was the detection of static objects on a table at various distance from the front of the robot. We used six small objects, whose size vary from 5cm to 10cm in width. The reference images were taken at a different time of the day with different background. Figure

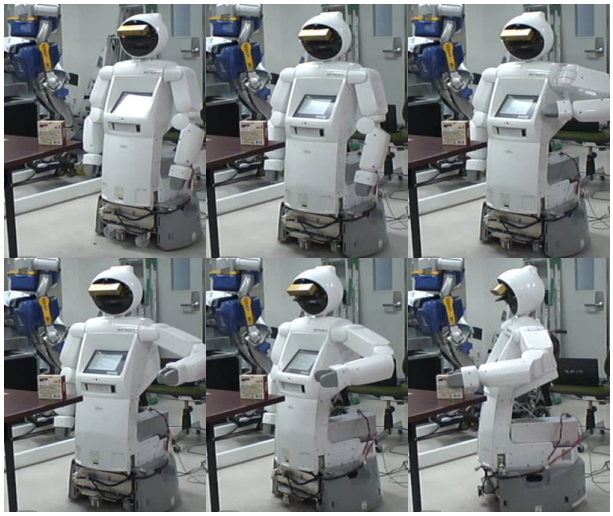

Fig. 6. On-line search-and-grasp: The photos show the robot searching, approaching, confirming the object pose, and grabbing the object.

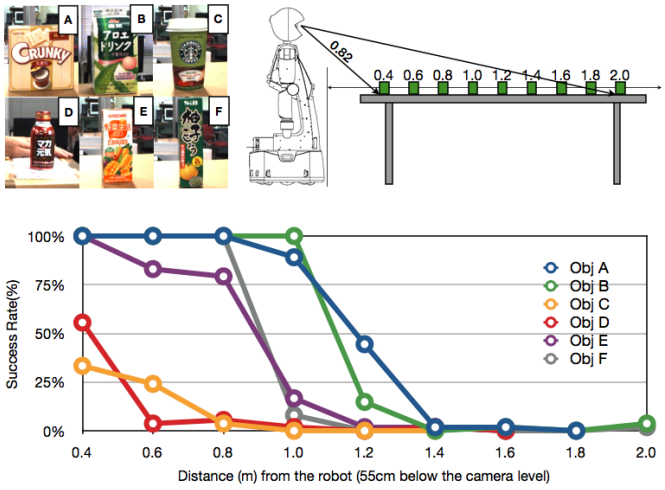

Fig. 7. The result of detection and 3D pose estimation for static objects. The objects were placed on a table at different distances from the robot. Small objects with round surface results in poor detection rate due to perspective transformation. When detected, the variance of the estimated pose in Z axis (the largest) is kept below 0.001(m) for planar objects (0.022 for object C and 0.042 for object D), which means very robust estimation against noise.

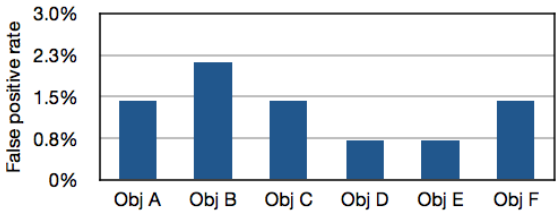

Fig. 8. False positive rate from a video sequence over 73 seconds, in which the robot was navigating through a very cluttered lab environment with many desks, chairs, bookshelves, humans, and other equipments. No target object was present at the scene.

7 shows the results from the experiments. As we can see from the result, our approach find the objects and estimates their pose reliably when they are close, and then success rate drops dramatically when they are too far away. The variance of the estimated pose over many observations of a static object was kept very small in our experiment, which means the pose estimation is very robust against noise. We consider this type of characteristics – sudden drop in the success rate – is more desirable than, for example, gradual deterioration of the estimated pose over distance. The object C and D are small and have round surfaces, and their input images have quite large perspective deformation from the reference images that are taken from a perpendicular viewpoint. As a result, their success rates are significantly lower than other objects with flat surface. Of course, the performance on the non-planar objects can be as good as the planar objects, if the viewing angle in the input images is the same as in the reference image.

To evaluate the false positive rate of the detection, we have also captured a video in a very cluttered lab environment by manually controlling the robot to move through the environment with varying head orientation. The environment had many desks, chairs, humans, bookshelves, and other equipments, without any of the target objects. The result of the object detection for each target pattern is shown in Figure 8. As we can see in the result, our approach using local geometric constraints produces very few false positives. Since the false positives are few and the estimated pose of them tend to be almost random, it is very easy to filter them out by simply checking consistency of the results over consecutive frames, for example.

V. CONCLUSIONS AND FUTURE WORKS

We have presented an approach for object detection and 3D pose estimation, based on local feature matching. The advantage of our approach is that it can be implemented easily as an independent component in a module-based robot control system, because our approach requires minimal information – single image and its 3D pose – for the reference pattern. We have also proposed a new matching scheme between the local features of the reference and scene images that exploits local geometric consistency.

We implemented our approach as a RT(Robot Technology) component that can be used in any RT-based robot control systems. Our experiments show that our matching approach can extend the range of object detection and the estimation results of 3D poses are robust enough, while it maintains very low false positive rate. Our RT-component implementation also shows that our approach can be used in an on-line robot control system for a search-and-grasp task in realistic environment.

Of course, our approach is limited by the same limitations of the local features, which requires reasonable amount of texture on the target object. And for the long range discovery where the object get considerably smaller and its signals get weaker, our approach do not work well for obvious reasons. For such cases, we just need other techniques such as active search and maybe some probabilistic approaches to deal with weak signals and big uncertainty.

Our future research direction includes improving the performance of our implementation, enhance the interoperability of our RT-component, and developing other higher-level components that will operate upon the estimation results of the 3D pose of objects that are returned by the component of our approach. Automatic capture of the reference pattern and its pose through human interactions is another interesting research direction.

ACKNOWLEDGMENT

The authors acknowledge the New Energy and Industrial Technology Development Organization (NEDO) of Japan for funding this research.

REFERENCES

[1] R. Katsuki, J. Ota, Y. Tamura, T. Mizuta, T. Kito, T. Arai, T. Ueyama, And T. Nishiyama, "Handling of Objects with Marks by a Robot", *Int. Conference on Intelligent Robots and Systems (IROS)*, 2003

[2] T. Drummond, R. Cipolla. "Real-time visual tracking of complex structures", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):932–946, 2002

[3] A.I. Comport, E. Marchand, and F. Chaumette, "Robust model-based tracking for robot vision", *International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pp 692–697, 2004

[4] J. Shi, C. Tomasi, "Good features to track", *Computer Vision and Pattern Recognition*, pp.593-600, 1994

[5] F. Jurie, M. Dhome, "Realtime 3D template matching", *Int. Conf. on Computer Vision and Pattern Recognition*, volume 1, pp 791–796, Hawai, December 2001

[6] L. Vacchetti, V. Lepetit, P. Fua, "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

[7] G. Hager, P. Belhumeur. "Efcient region tracking with parametric models of geometry and illumination", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, volume 20(10), pp 1025–1039, 1998

[8] M. Pressigout, E. Marchand, "Real-time planar structure tracking for visual servoing: a contour and texture approach". *International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pp 1701–1706, 2005

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

[10] D.G. Lowe, "Distinctive Image Features from Scale-invariant Keypoints", *Int. Journal of Computer Vision*, vol. 20, pp 91-110, 2003

[11] C. Choi, S. Baek, S. Lee, "Real-time 3D Object Pose Estimation and Tracking for Natural Landmark Based Visual Servo" *International Conference on Intelligent Robots and Systems (IROS)*, 2008

[12] H. Bruyninckx, "Open robot control software: the orocos project", *International Conference on Robotics and Automation(ICRA)*, pages 2523–2528. IEEE Press, 2001

[13] M. Mizukawa, H. Matsuka, T. Koyama, T. Inukai, A. Nodad, H. Tezuka, Y. Noguch, and N. Otera. "Orin: Open robot interface for the network". *In SICE*, pages 925–928. IEEE Press, 2002

[14] A. Makarenko, A. Brooks, and T. Kaupp. "Orca: Components for robotics", *International Conference on Intelligent Robots and Systems (IROS)*, pages 163–168, 2006

[15] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, W.K. Yoon, "RT-Middleware: Distributed Component Middleware for RT (Robot Technology)", *International Conference on Intelligent Robots and Systems (IROS)*, pp.3555-3560, Edmonton, Canada, 2005

[16] J. Bauer, N. Sünderhauf, P. Protzel, "Comparing Several Implementations of two Recently Published Feature Detectors", *Int. Conf. on Intelligent and Autonomous Systems*, 2007

[17] A. Vedaldi and B. Fulkerson, "VLFeat: An Open and Portable Library of Computer Vision Algorithms", *http://www.vlfeat.org/*, 2008

[18] R. Hartley and A. Zisserman, "Multiple View Geometry in computer vision", *Cambridge University Press*, pp. 32–33. ISBN 0-521-54051-8, 2003