Position and Pose Recognition of Randomly Stacked Objects using Highly Observable 3D Vector Pairs

Shuichi Akizuki and Manabu Hashimoto
Graduate School of Information Science and Technology
Chukyo University
Aichi, Japan
Email: {akizuki, mana}@isl.sist.chukyo-u.ac.jp

***Abstract*—We propose a fast and reliable 3D object detection method that can be applied for complicated scenes consisting of randomly stacked objects. The proposed method uses "3D vector pair" that has a common start point and different end points and it has surface normal distribution as the feature descriptor. By considering the observability of vector pairs, the proposed method has been achieved high recognition performance. Observability factor of the vector pair is calculated by simulating the visible state of the vector pair from various viewpoints. By integrating the observability factor and the distinctiveness factor proposed in our previous work, vector pairs that have effectiveness for matching are extracted and these are used for object pose estimation. Experiments have confirmed that the proposed method increases the recognition success rate from 45.8% to 93.1%, in comparison with the state-of-the-arts method. The processing time of the proposed method is fast enough for the robotic bin-picking.**

I. Introduction

To enhance factory automation, it is desirable to implement bin-picking systems that can pick up randomly stacked industrial components with a robot arm. Picking tasks require fast and accurate recognition of the position and pose of objects. An effective approach to achieving this is to apply a model based method that matches a 3D model of an object and an input scene. In this approach, processing costs can be reduced by matching only feature points. Various methods have been developed for extracting 3D feature points; typical example are the Local Surface Patch (LSP) method [1] and The Depth Aspect Image (DAI) method [2], which use mathematical conditions of geometric surface shapes. The LSP method extracts feature points that have extremal values of the Shape Index [3]. The DAI method extracts 3D points that have a large curvature.

Other methods are the Point Feature Histogram (PFH) method [4] and the Fast PFH (FPFH) method [5], which have achieved stable point correspondence by using 3D points that are outliers of the feature vectors of an object model. Another is the Signature Histogram of OrienTation (SHOT) method [6], which extracts 3D points for which there is high repeatability of the reference frames that are local coordinates of feature points.

Recently a pose estimation method was proposed that extracts the Point Pair Features [7], [8] consisting of a pair of oriented 3D points from the boundaries of the object [9]. Although these 3D feature point extraction methods have helped to reduce processing costs and to improve matching reliability, they share a common problem: all extracted feature points are not necessarily observable in the input scene. Recognition reliability decreases if correspondence points are occluded in the input scene because feature points are extracted whether or not they are observable in the input scene.

To address this problem, we extract feature points that can be stably observed in the input scene by computing the observability of data points in an object model. Reliable recognition is achieved by using such feature points.

There are two cases in which feature points are not observable in the input scene:
1.The feature point is occluded by other objects.
2.The feature point is occluded by its own shape.

For practical purposes little consideration needs to be given to the first case, since it is rather difficult to pick up objects occluded by other objects without interfering with the occluding objects. On the other hand, in the second case objects may not be observable even if they are placed on top of a stack. Accordingly, considering the second case in this research, we calculate the observability of 3D points of the object model and extract feature points from stably observed points.

We adopt the Vector Pair Matching (VPM) method [10] for position and pose estimation in the object model. This method can quickly estimate the position and pose of objects by using a pair of 3D vectors that have a common start point and different end points. It regards vector pairs having features with low occurrence probability in the object model as distinctive vector pairs and uses them to reduce the risk of mismatching.

By extracting vector pairs on the basis of feature distinctiveness and feature point observability, the proposed method simultaneously achieves both fast and accurate pose estimation.

In Section 2, the Vector Pair matching we proposed reference [10] is introduced. Section 3 gives the method of extracting vector pairs by integrating observability and distinctivensee. Section 4 shows the method of pose estimation of the object by using voting based method. Section 5 shows experimental results we obtained using a lot of point cloud data, results that demonstrate our method has better performance than previous methods such as the SHOT or the FPFH method.

978-1-4799-4032-5/14/$31.00 ©2014 IEEE

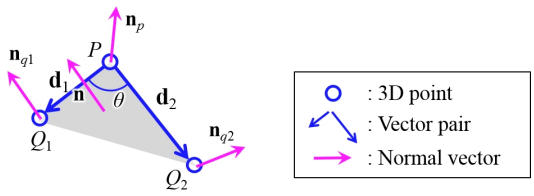


Fig. 1. Structure of 3D vector pair. Blue circle represents 3D point, Pair of blue arrow represent the vector pair, and pink arrow represents normal vector of a point or a triangle.

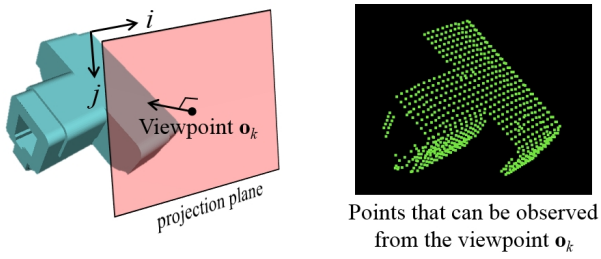


Fig. 2. An example of projection plane(left) and its observable points(right).

II. Basic method: Vector Pair Matching (VPM) [10]

A. Structure of 3D vector pair

In general, only three 3D points are necessary to determine rigid transformations of an object. We treat these three points as a pair of 3D vectors that have a common start point and different end points. Here, the vector pair $\mathbf{v}$ consists of a start point P and the end points Q_1, Q_2. Displacement vectors P-Q_1 and P-Q_2 are represented by $\mathbf{d}_{q1}$ and$\mathbf{d}_{q2}$, respectively. The angle between $\mathbf{d}_{q1}$ and $\mathbf{d}_{q2}$ is defined as θ. The vector pair's features $f(s_p, s_{q1}, s_{q2})$, which represent the distribution of normal vectors, are calculated by Equation 1.

$$s_p = \mathbf{n} \cdot \mathbf{n_p}, s_{q1} = \mathbf{n} \cdot \mathbf{n}_{q1}, s_{q2} = \mathbf{n} \cdot \mathbf{n}_{q2} \tag{1}$$

Here, $\mathbf{n}_p$, $\mathbf{n}_{q1}$, $\mathbf{n}_{q2}$ represent the normal vectors of P, Q_1, Q_2 and $\mathbf{n}$ represents the normal vector of $\triangle PQ_1Q_2$. The structure of a 3D vector pair is illustrated in Figure 1.

B. Extraction of distinctive 3D vector pairs

Vector pairs having features with low occurrence probability in the object model are used for the matching process. These vector pairs contribute to stable matching because they are laid on the distinctive shape of the object. We call these vector pairs "distinctive vector pairs".

First of all, vector pairs having the predefined parameters $|\mathbf{d}_{q2}|$, $|\mathbf{d}_{q2}|$ and θ are extracted from the object model. A 3-D co-occurrence histogram is generated by Equation 2 and 3.

$$h(s_p, s_{q1}, s_{q2}) = \sum_{n=0}^{N} \delta(f_n) \tag{2}$$

$$\begin{cases} \delta = 1 & when\{f(s_p) = s_p\} \cap \{f(s_{q1}) = s_{q1}\} \\ & \quad \cap \{f(s_{q2}) = s_{q_2}\} \\ \delta = 0 & otherwise \end{cases} \tag{3}$$

N is the number of extracted vector pairs. The number of vector pairs which have feature $f(s_p, s_{q1}, s_{q2})$ are accumulated to cell of co-occurrence histogram $h(s_p, s_{q1}, s_{q2})$. By using this histogram, occurrence probability of vector pairs Ph are calculated by Equation 4. Given a occurrence probability Ph, we denote $1 - Ph(s_p, s_{q1}, s_{q2})$ as the distinctiveness of the vector pair.

$$Ph(s_p, s_{q1}, s_{q2}) = \frac{h(s_p, s_{q1}, s_{q2})}{\sum_{s_p=0}^{L-1} \sum_{s_{q1}=0}^{L-1} \sum_{s_{q2}=0}^{L-1} h(s_p, s_{q1}, s_{q2})} \tag{4}$$

Finally, the vector pairs that have low occurrence probability (up to about 10% of the total) are extracted and used for the matching process. Details of this process are described in reference [10].

III. Extraction of vector pairs by using observability

A. Definition of observability

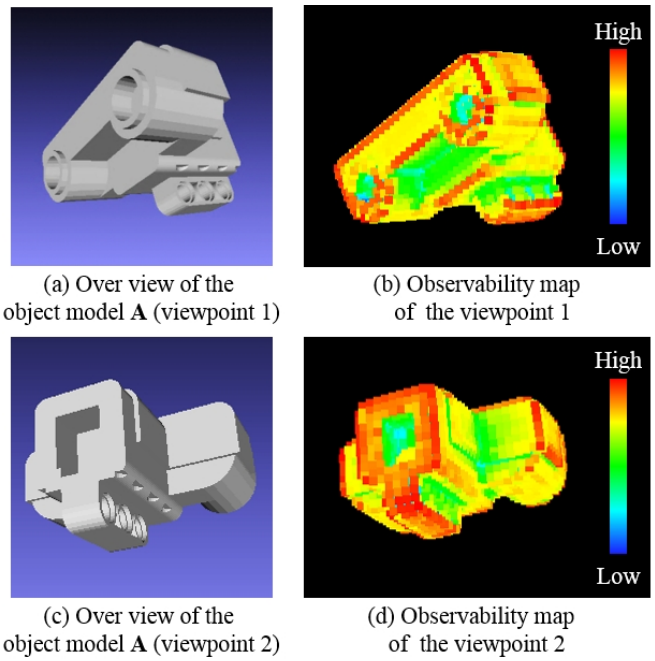
Using an object model represented by point cloud data, we defined "observable points" as points that can be observed from a certain viewpoint. The observability of an object's surface is affected by (1) the range measurement method, (2) the internal constitution of a range sensor, (3) the distance from the range sensor to the object, (4) the viewpoint direction, and (5) the object's shape. We did not consider (1) to (3) in this research because it is not so easy to get information about them before the recognition process is performed. Thus, we calculated the observability by using only the information we had about (4) and (5). Projecting the object model points to a projection plane that has a viewpoint vector as a normal vector, we determined the observable points as those which, out of those projected to the same plane coordinates, were nearest to the projection plane. Figure 2 shows example of projection plane and its observable points.

B. Calculation of observability

The observability of the object model is calculated with respect to each surface data point by using the viewpoints randomly disposed. The observability of point p is calculated by Equation 5.

$$Obs(p) = \frac{1}{K} \sum_{k=0}^{K-1} \delta(p, \mathbf{o}_k) \tag{5}$$

Here, $\mathbf{o}_n$ represents the direction of the viewpoint vector and the function δ returns '1' when point p is observed. K represents the number of viewpoint. Figure 3 shows views



(a) Over view of the object model **A** (viewpoint 1)

(b) Observability map of the viewpoint 1

(c) Over view of the object model **A** (viewpoint 2)

(d) Observability map of the viewpoint 2

Fig. 3. Object model and observability map views (blue: low observability; red: high observability).

of an object model calculated from 500 viewpoints and the corresponding observability map views.

Overviews of the object model are shown in (a) and (c) and the corresponding observability map views appear in (b) and (d). Blue and red indicate low and high observability, respectively. For the outside contour parts observability is high and for indented points it is low. The bottoms of the deeply indented parts have the lowest observability. We calculated $Obs(\mathbf{v})$, the observability of vector pair $\mathbf{v}$, by using the same approach as that used to calculate the observability of surface data points. As the probability of P, Q_1, and Q_2 being observed simultaneously, $Obs(\mathbf{v})$ is calculated by Equation 6.

$$Obs(\mathbf{v}) = \frac{1}{K}\sum_{k=0}^{K-1}\delta(P,\mathbf{o}_k)\delta(Q_1,\mathbf{o}_k)\delta(Q_2,\mathbf{o}_k) \quad (6)$$

C. Vector pair extraction by integrating observability factor and distinctiveness factor

Efficient vector pairs for stable recognition are selected by integrating observability factor $Obs(\mathbf{v})$ and occurrence probability $Ph(\mathbf{v})$. The integrated value is calculated by Equation 7 (It would seem that the parameters ω_1 and ω_2 should be explained, especially since they are referred to in subsection 5.3.). Vector pairs $Vm = \{\mathbf{v}_{mn}|n = 0, ..., N-1\}$ having a high integrated value $I(\mathbf{v})$ are used for matching. Here, N means the number of vector pairs used for matching. Figure 4 shows the overview of the algorithm to extract 3D vector pairs by integrating observability factor and distinctiveness factor.

$$I(\mathbf{v}) = \omega_1 Obs(\mathbf{v}) + \omega_2(1 - Ph(\mathbf{v})) \quad (7)$$

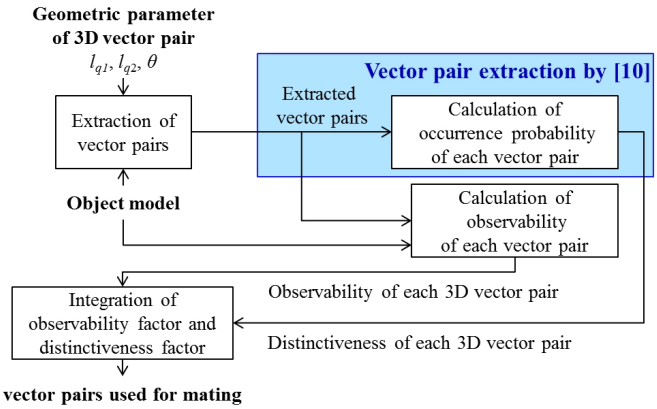


Fig. 4. Overview of algorithm to extract 3D vector pairs by integrating observability factor and distinctiveness factor.

IV. Voting Based Position and Pose Recognition by Vector Pair Matching

The position and pose of the object model is calculated by the following process.

Vector pair matching

1) Preprocessing
2) Extracting vector pairs
   All vector pairs $V_s = \{\mathbf{v}_{sm}|m = 0, ..., M-1\}$ are extracted from the input point cloud S.

[Repeat M times]

3) Calculation and voting of pose hypothesis
   a) Decision of correspondence points $c_k = (\mathbf{v}_m, \mathbf{v}_s)$
   b) Calculation transform parameters $T_k = (\mathbf{R}_k, \mathbf{t}_k)$
   c) Voting

[End repeat]

4) Clustering of pose hypothesis
5) Model matching

For efficient position and pose estimation, the proposed method uses voting based processing. All vector pairs extracted from the object model that are associated with offset vector $\mathbf{d}_c$, which represents the geometric relation from P to the center of the object model, are accumulated to the voting space.

1) Preprocessing: First, we uniformly down sample the input point cloud S and remove it from the plane. We call the reduced point cloud S^*.

2) Extracting vector pairs: All vector pairs of input point cloud $V_s = \{\mathbf{v}_{sm}|m = 0, ..., M-1\}$ having l_{q1}, l_{q2} and θ are extracted from S^*.

3) Calculation and voting of pose hypothesis: Correspondence points are found by using the extracted vector pair V_m and V_s. Before the processing described above, V_m is indexed into a hash table. Hash table bins represent the discretized features $f(s_p, s_{q1}, s_{q2})$ of the vector pair. We experimentally discretize each bin by 12 degrees.

a) Vector pair correspondence $c_k = (\mathbf{v}_m, \mathbf{v}_s)$ is generated by

using the feature of vs to access the hash table.
b) Rigid transformation $T(\mathbf{R}, \mathbf{t})$, which aligns $\mathbf{v}_m$ to $\mathbf{v}_s$, is computed for all correspondence c. In this process, a large number of c will be generated. For generating of pose hypothesis quickly, $\mathbf{R}$ is calculated by simple multiplication of matrices. (see Equation 8.)

$$\mathbf{R} = \mathbf{M}_m^T \mathbf{M}_s \tag{8}$$

Where, $\mathbf{M}_m$ and $\mathbf{M}_s$ represents $[Q_1 - P \; Q_2 - P \; (Q_1 - P) \times (Q_2 - P)]$ which are created from vector pair in the object model and the input scene. Relashionship between $\mathbf{M}_m$ and $\mathbf{M}_s$ are represented as $\mathbf{M}_m \mathbf{R} = \mathbf{M}_s$. Here, we set θ that is geometric parameter of the vector pair as 90 degree. Because we can change $\mathbf{M}_m$ and $\mathbf{M}_s$ into orthogonal matrices and $\mathbf{M}_m^{-1} = \mathbf{M}_m^T$. So, computational cost of calculating $\mathbf{R}$ will be decrease.
c) The position of the center of the object model in the scene is computed on the basis of $C = \mathbf{d}_c \mathbf{R} + \mathbf{t}$ and the voting value of C of the voting space is obtained.

4) Clustering of pose hypothesis: After the voting process, coordinates which got a voting value that exceeded the threshold are detected and T' is calculated by averaging the rigid transformations T, which voted for these coordinates. Outliers of T are then eliminated by simple clustering.

5) Model matching From multiple values of T', that which minimizes alignment error of the object model and the input scene is determined as a final recognition result. Errors are calculated by the sum of absolute distance of the transformed object model and S. By reducing the number of object model points and applying the sequential similarity detection algorithm (SSDA), the error calculation process is speeded up.

V. Experimental results

A. Accuracy of pose estimation

In order to investigate pose estimation accuracy of the proposed method, we tested the proposed method to input scenes have grand truth. Input scenes used in this experiment are created by randomly rigid transforming the object model. Rotation range is -30 to 30[deg], and translation range is -300 to 300[mm]. Point cloud's density of input scene is 3.0[mm/point]. Figure5 shows distribution of estimation error of rotation and translation. Error of translation is the difference value of the ground truth and estimated xyz translations. Error of rotation is the difference value of Euler angle of the ground truth and estimated xyz rotation angles.

According to the rotation, almost error of estimated result was within 5[deg]. If estimated error is 5[deg], error between the most distant point from object's centroid and the input point cloud is 2.9[mm]. According to the translation, all error of estimated result was within resolution of input scene. The proposed method performed high accurate pose estimation.

B. Experimental data

In this section, we will explain the methods used for object model and input scene generation.

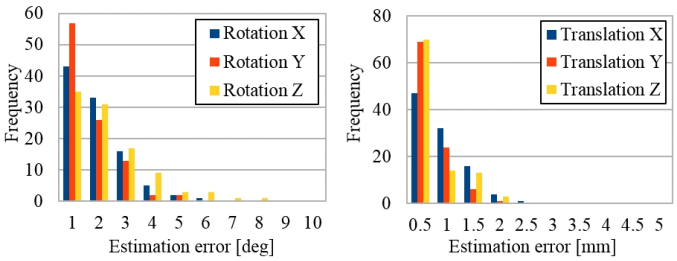


Fig. 5. Distribution of accuracy of estimated position and pose parameters. Left: Distribution of rotation error. Right: Distribution of translation error.

Object model: We used uniformly dense point cloud data surrounding the object. It was created by generating a point cloud in each triangular patch of STL or PLY format data.

Input scene: We prepared two kinds of data, for synthetic scenes (Figure (a) to (c)) and real scenes (Figure (d) . Synthetic scenes were created by using "Google SketchUp" modeling software and a physics simulator and were converted to point cloud data. Real scenes were captured by a laser scanner. The point cloud resolution is 0.4[mm/point].

C. Comparison of recognition performance

In order to estimate recognition performance of our method , we compared it with that of the following methods.

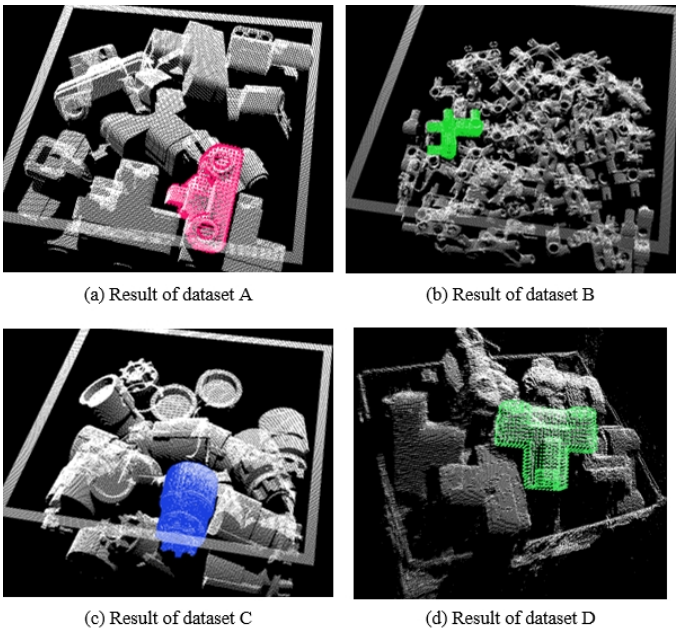
1) SHOT [6]
2) FPFH [5]
3) VPM (Distinctiveness) [10]
4) VPM (Observability) proposed method 1
5) VPM (Integration) proposed method 2

We implemented each method by using the Point Cloud Library [13]. Figure 6 shows example recognition results obtained with the proposed method. Recognized objects have been superimposed on input point clouds as colored point clouds. The relationship between recognition rate and processing time is shown in Table I. The algorithm presented in this paper was implemented in C++ and tests were performed on a desktop computer with an Intel Core i7 860 2.8GHz CPU and 12GB RAM.

SHOT descriptor + Correspondence Grouping method: Since the SHOT descriptor has a lot of (over 1,000) dimen-

TABLE I. Relationship between recognition rate $Pr[\%]$ and processing time $T[\text{sec}]$.

Feature	Matching method		A	B	C	D
SHOT[6]	Correspondence Grouping[11]	Pr	51.3	77.9	33.6	20.3
		T	27.78	19.46	33.69	22.48
FPFH[5]	RANSAC based model matching[12]	Pr	54.9	11.5	73.5	31.0
		T	0.67	3.97	1.07	0.91
Vector pair (Distinctiveness)[10]	Vector Pair Matching	Pr	40.7	31.0	47.8	33.6
		T	0.27	1.41	0.39	0.72
Vector pair (Observability)		Pr	97.3	100.0	94.7	85.0
		T	0.47	1.76	1.76	1.99
Vector pair (Integration)		Pr	99.1	95.6	92.0	85.8
		T	0.44	1.76	1.42	1.81



(a) Result of dataset A (b) Result of dataset B

(c) Result of dataset C (d) Result of dataset D

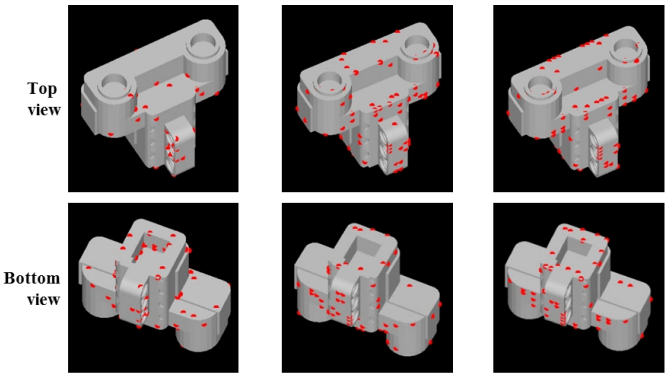
Fig. 6. Example recognition results for four datasets (superimposed on input point cloud).

sions, it has longer processing time than the other methods that were compared. In addition, because the input scenes consisted of densely stacked objects, data for multiple objects was easily included in the feature description area and consequently the recognition rate was inferior.

FPFH descriptor + RANSAC based model matching method: Since this matching method can calculate pose hypothesis at low cost, its processing time was fast. However, its recognition rate was inferior because its accuracy in estimating object model poses is not particularly high.

Vector Pair Matching: Figure 7 shows vector pairs extracted using methods 3) to 5). For visibility reasons, lines connecting feature points are not displayed. In the method that extracts vector pairs by using distinctiveness only, many vector pairs were extracted from indented parts (See bottom of Figure 7. (a)). Since this object model had few indented parts, many vector pairs were extracted from them. Adding observability to the extraction criteria reduced the number of vector pairs (see bottom right of Figure 7.). In contrast, it increased the number of vector pairs extracted from shapes with high observability, such as outside contour parts (see top right of Figure 7.).

This method achieved fast point correspondence by using low dimensional feature vectors for the matching process. As a result, it had faster processing time than the other methods. In the case of extracting vector pairs using distinctiveness only (method 3)), some extracted vector pairs were difficult to observe in the input scene. Thus, the recognition rate was inferior. This problem was solved by considering vector pair observability we proposed in this research (method 4), method5)), which increased the recognition rate. As a result, the proposed method had recognition rate and processing time superior to those of the other methods.



(a) Distinctiveness only[9] (b) Observability only (c) Observability + Distinctiveness

Fig. 7. Extracted vector pairs by three methods, distinctive only [10] (method 3)), observability only (method 4)) and integration of distinctiveness and observability (method 5)) .

D. Parameter of the VPM

To achieving high recognition performance, it is important to decide optimal value of each parameter. In this subsection, we examine relationship between recognition performance and two main parameters of the VPM, the number of vector pairs used for matching N_{np} and the weighting factors about distinctiveness and observability of Equation 7.

1) The number of vector pairs N_{vp}: The parameter N_{vp} represents the number of vector pairs used for matching. It affects the performance of recognition rate and processing time. Therefore, it is important to decide the optimal value of N_{vp}. Figure 8 shows measured relationship between Nvp and the recognition rate. In this experiment, We compared three vector pair extraction method, the distinctiveness of feature based method [10] (Distinctiveness), observability of the vector pair based method we proposed in this research (Observability), and integration of the distinctiveness and the observability method we proposed (Integration), too. In this figure, vertical axis shows recognition rate for 113 scenes, horizontal axis shows N_{vp} as percentage of all points of the object model.

In the case of $N_{vp} < 10\%$, recognition rate of the distinctiveness method was low. On the other hand, the observability method and the integration method achieved recognition rate of more than 90%. The reason why the observability method and the integration method report better result than the distinctiveness method is that they were acquired many correspondence points from the input scenes. In fact, the case of N_{vp}=10%, the number of correspondence point of recognized object of the distinctiveness method is 2.2. The observability method and the integration method got 8.1 and 6.3, respectively. Considering the observability of the points of object model contributes to increase the number of correct correspondence vector pairs.

In the case of $10\% < N_{vp}$, there ware not improvement of recognition rare. This results suggest that the optimal value of N_{vp} is 10% of all points of the object model.

2) Weighting factors of distinctiveness and observability: In extracting vector pairs, it is important to determine the weight of observability and distinctiveness. Accordingly, we

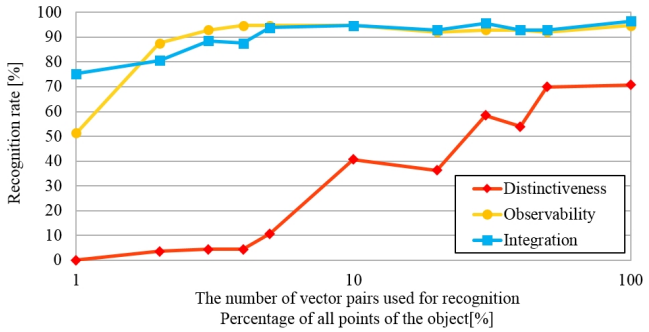


Fig. 8. Relationship between recognition rate and the number of vector pairs.

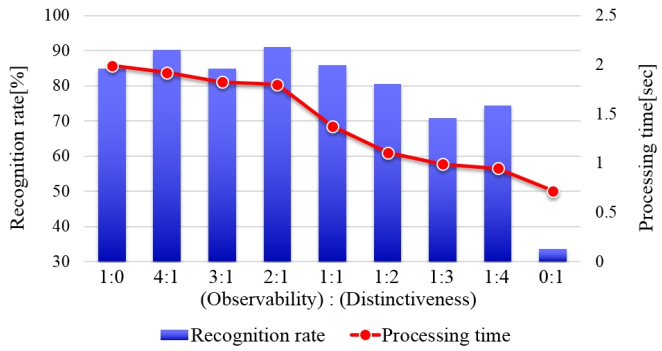


Fig. 9. Relationship between recognition rate and processing time when varying ω_1 and ω_2.

compared the recognition performance our method achieved when the parameters ω_1 and ω_2 in Equation 7 were changed. Figure 9 shows the relationship between recognition rate and processing time in this comparison. When (Observability) : (Distinctiveness) = 0 : 1, recognition rate was inferior. In contrast, recognition rate is increased by considering not only distinctiveness of vector pairs but also observability of ones. When the ratio of observability to distinctiveness is high, high recognition rate is achieved. However, processing time is increased because many pose hypotheses were generated. In contrast, when the ratio of distinctiveness to observability is high, fast recognition is achieved. However, the recognition rate is decreased because correspondence points sometimes do not appear in the input scene.

As a result, we can conclude that the balance between recognition rate and processing time is good when the observability-to-distinctiveness ratio is 1:1.

VI. Conclusion

We proposed a fast and reliable 3D object detection method that can recognize the position and pose of objects in complicated scenes consisting of randomly stacked objects. By integrating observability of the object surface and distinctiveness of features, the proposed method extracts 3D feature points. In recognition process, efficient pose estimation has been achieved by using voting based pose hypothesis clustering and fast hypothesis generation by using 3D vector pairs that consist of three 3D points.

Experiments confirmed that in comparison with the Correspondence method, the proposed method is about 19 times faster and increases the recognition success rate from 45.8% to 93.1%. The proposed method is fast enough for application to bin-picking robots because its processing time is shorter than general robot motion time. We also confirmed that the method maintains high-speed performance for a wide variety of object shapes.

In future work, we will attempt to improve the method's recognition performance by optimizing the geometric parameters of 3D vector pairs.

References

[1] H. Chen and B. Bhanu, "3D Free-form Object Recognition in Range Images using Local Surface Patches," *Journal Pattern Recognition Letters*, vol.28, issue 10, pp.1252–1262, 2007.

[2] T.Takeguchi and S.Kaneko, "Depth Aspect Images for Robust Object Recognition," *in Proc. SPIE Conference on Optomechatronic Systems IV*, vol.5264, pp.54–65, 2003.

[3] C. Dorai and A. K. Jain, "COSMOS-A Representation Scheme for 3D Free-Form Objects," *Trans. on IEEE Pattern Analysis and Machine Intelligence*, vol.19, issue 10, pp.1115–1130, 1997.

[4] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Aligning Point Cloud Views using Persistent Feature Histograms," *in Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp.3384–3391, 2008.

[5] R. B. Rusu, N. Blodow, M. Beetz, Fast Point Feature Histograms (FPFH) for 3D Registration, *in Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pp.3212–3217, 2009.

[6] F. Tombari, S. Salti, L. Di Stefano, "Unique Signatures of Histograms for Local Surface Description," *in Proc. 11th European Conference on Computer Vision (ECCV)*, pp 356–369, 2010.

[7] S. Winkelbach, S. Molkenstruck, F. M. Wahl, "Low-Cost Laser Range Scanner and Fast Surface Registration Approach," *in Proc. 28th DAGM Symposium*, pp.718–728, 2006.

[8] B. Drost, M. Ulrich, N. Navab and S. Ilic, "Model Globally, Match Locally: Efficient and Robust 3D Object Recognition," *in Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pp.998–1005, 2010.

[9] C. Choi, Y. Taguchi, O. Tuzel, M. Liu, and S. Ramalingam, "Voting-based Pose Estimation for Robotic Assembly using a 3D Sensor," *in Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pp.1724–1731, 2012.

[10] S. Akizuki, M.Hashimoto, "Fast and Reliable 3-D Object Recognition based on Surface Normal Distributions," *in Proc. International Symposium on Optomechatronic Technologies (ISOT)*, pp.1–9, 2013.

[11] F. Tombari and L. Di Stefano, "Object recognition in 3D scenes with occlusions and clutter by Hough voting," *in Proc. Fourth Pacific-Rim Symposium on Image and Video Technology*, pp.349–355, 2010.

[12] A. G. Buch, D. Kraft, J. Kamarainen, H. G. Petersen, N. Kruger, "Pose Estimation using Local Structure-Specific Shape and Appearance Context," *in Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2080–2087, 2013.

[13] R. B. Rusu and S. Cousins: "3D is Here: Point Cloud Library (PCL)," *in Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pp.1–4, 2011.