Recognition of the Multi Specularity Objects using the Eigen-Window

Kohtaro Ohba and Katsushi Ikeuchi

29 February 1996
CMU-CS-96-105

School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

This research was sponsored in part by the U.S. Advanced Research Project Agency under Army Research Office, the Department of the Army under Grant DAAH04-94-G-0006, and in part by the Office of Naval Research under Grant N00014-93-1-1220.

Keywords: Object Recognition, Specularity, Multi Objects, Eigen-space

Abstract

This paper describes a method for recognizing partially occluded objects for bin-picking tasks using the eigen-space analysis. Although effective in recognizing an isolated object, as was shown by Murase and Nayar, the current method cannot be applied to partially occluded objects that are typical in bin-picking tasks. The analysis also requires that the object is centered in an image before recognition. These limitations of the eigen-space analysis are due to the fact that the whole appearance of an object is utilized as a template for the analysis. We propose a new method, referred to as the "eigen-window" method, that stores multiple partial appearances of an object in an eigen-space. Such partial appearances require a large number of memory space. A similarity measure among windows is developed to eliminate redundant windows and thereby reduce memory requirement. Using a pose clustering method among windows, the method determines the pose of an object and the object type of itself. We have implemented the method and verify the validity of the method.

Contents

1. Introduction

Bin-picking, picking up one part from a large number of similar objects, is still one of the most challenging problems. Some of the earlier works in this domain include: [1-5]. Despite this long history, this bin-picking problem still provides a challenge to vision researchers. Some of the difficulty includes: real-time requirements, difficulty in segmentation, and difficulty in modeling objects.

Recently, visual learning methods [6-19] have shown a potential to solve some of these above-mentioned problems. These methods learn object models from a series of images taken in the same environment as in the recognition mode. Thus, this method by-passes the difficulty in modeling. Furthermore, since such a method stores an object model as a collection of appearance parameters, recognition speed is very rapid and it can achieve the real-time system.

The bin-picking problem requires the system to handle partial occlusion. Although powerful, the eigen-space analysis assumes that all the appearances are non-occlusions. In order to apply the eigen-space analysis to recognition of partially occluded objects, we propose to divide appearances into small windows, referred to as "eigen-windows" and to apply the eigen-space analysis to each eigen-window [21]. The basic idea is that, even if some of the windows are occluded, the remaining windows are still effective and can recover the object pose. In Section 2, we review the eigen-space analysis; discuss the limitations of the eigen-space analysis; and explain how to overcome these limitations using the eigen-window method. Since the total number of such small windows is very large and since storing all of them may require a prohibitive amount of memory space, we consider a method to automatically select only effective windows. Section 2.4 explains the method for selecting effective windows. Section 3 shows some of the experimental results, and Section 4 concludes this paper.

2. Eigen-Window Method

First, we will review the eigen-space technique and discuss the limitations of the technique under the image shift, occlusion and noise. Then, we will introduce the new method to overcome this problem using the eigen-window method.

2.1. Eigen-Space Technique

Let M be the number of the images in a training set. These images, $z_1\ z_2\ \ldots\ z_M$ are taken using the experimental setup as shown in Figure 1. Each image z_i, of which dimension is $N \times N$, has been converted into a column vector of the length N^2:

$$\left[z_1\ z_2\ \ldots\ z_M\right] \quad . \tag{1}$$

By subtracting the average brightness of the all images, we obtain the training matrix,

$$Z = \left[z_1 - c\ z_2 - c\ \ldots\ z_M - c\right] \quad , \tag{2}$$

where c is the average, and the size of the matrix is N^2 by M.

Then the covariance matrix Q, $N^2 \times N^2$, is obtained as:

$$Q = ZZ^T \quad . \tag{3}$$

This covariance matrix provides a series of eigenvalues λ_i and eigenvectors $e_i\left(i = 1, \ldots, N^2\right)$.

Each pair of eigenvalue and eigenvector holds:

$$\lambda_i e_i = Qe_i \quad . \tag{4}$$

Namely, the Q matrix can be decomposed into the N^2 orthonormal components, of which the eigenvalues are λ_i. Thus, each image set can be described with these eigenvectors and eigenvalues.

For the sake of memory efficiency, we will ignore smaller eigenvalues and corresponding vectors using a threshold value, T_s :

$$W_k = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{N^2} \lambda_i} \geq T_S \quad , \tag{5}$$

where k is sufficiently smaller than original dimension N^2 .

Once we can get these eigenvectors, we can construct the eigenvector matrix $E = \left[e_1\ e_2\ \dots\ e_k\right]$ to project an image, z_i (dimension N^2) into the eigen-space as an eigen point, g_i (dimension k).

$$g_i = E^T (z_i - c) \quad . \tag{6}$$

This eigen-space analysis can drastically reduce the dimension of the images, N^2 to the eigen-space dimension, k while keeping several of the most effective features to reconstruct the original images.

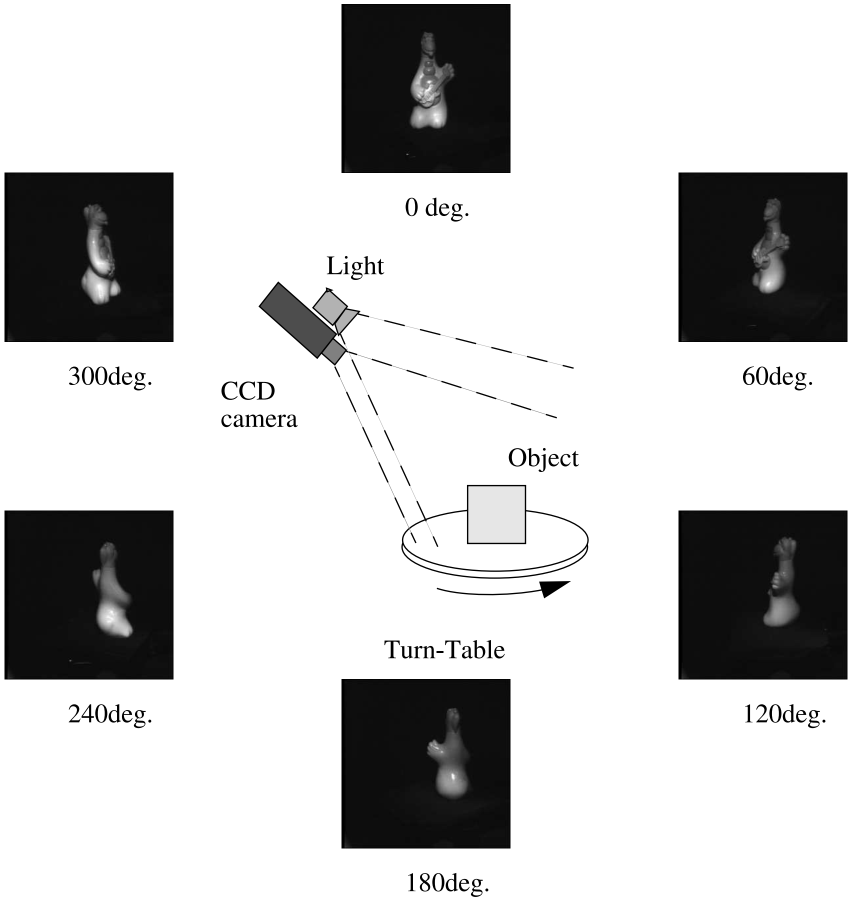


Figure 1. Experimental Setup

2.2. Limitations of the Eigen-Space Technique

An eigen-space representation, a collection of points in the eigen-space, is very sensitive to image conditions --- background noise, image shift, and occlusion of objects. For example, Figure 2 shows the effect of the image shift in the eigen-space. In this figure, one of the closed loops, denoted by “o”, depicts those points projected from a series of 128×128 *pixel* images of the object in Figure 1. Another closed loop, denoted by “*”, depicts those given by a series of images that were shifted by 16 pixel of the same object. This figure demonstrates that the image shift gives a significant effect on eigen-space representations. We have also evaluated the other factors, such as occlusion and noise in the eigen-space, and verified that each factor has a similar degree of effect on eigen-space representations.

As an effort to reduce these disturbance effects in the eigen-space, Murase and Nayar segmented only a window circumscribing the object using the movement of the object. Unfortunately, however, the usual bin-picking scenario does not provide such convenient clues for segmenting out a target region. Moreover, it often occurs that one window contains other objects due to the cluttered environment typical in a bin-picking scenario. Thus, we need a method to overcome these limitations.

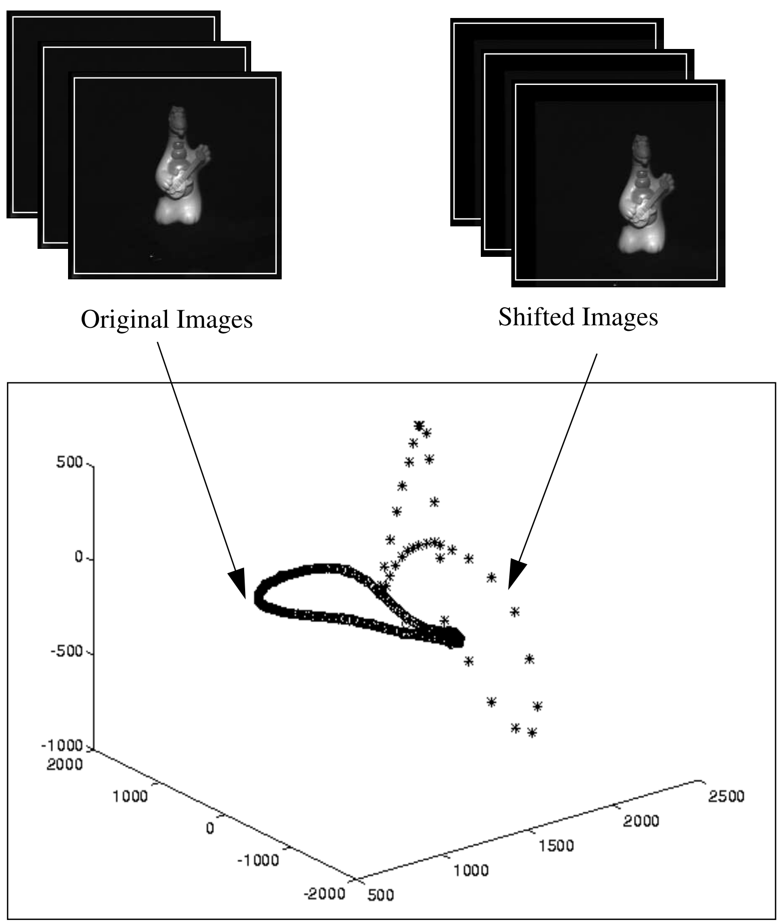


Figure 2 Disturbance Effect in the Eigen-space Given by Shifted Images

2.3. Eigen-Window Technique

To reduce the disturbance effects, we propose to apply small windows to the original images and to project all of them into the eigen-space. We refer to this method as the "eigen-window" technique. Figure 3 shows the overview of the technique.

2.3.1. Training Eigen-Windows

The training set of eigen-windows is given as:

$$Z = \begin{bmatrix} \zeta_1 & \zeta_2 & \dots & \zeta_M \end{bmatrix} =$$

$$\begin{bmatrix} \begin{bmatrix} z_1^1 - c & z_1^2 - c & \dots & z_1^{n_1} - c \end{bmatrix} & \begin{bmatrix} z_2^1 - c & z_2^2 - c & \dots & z_2^{n_2} - c \end{bmatrix} & \dots & \begin{bmatrix} z_M^1 - c & z_M^2 - c & \dots & z_M^{n_M} - c \end{bmatrix} \end{bmatrix} \tag{7}$$

where ζ_i denotes the collection of eigen-windows from the ith training image; z_i^j does the j eigen-window in the ith training image; n_i does the number of eigen-window in the i th image; c is the average value. In Figure 3, the white square denotes one of the training eigen-window.

The total number of eigen-windows in the training set is given by:

$$n_{total} = \sum_{i=1}^{M} n_i \quad . \tag{8}$$

Note that all the projected points of these eigen-windows are represented in the common eigen-space as shown in Figure 3. Each point in the space has the label of the original eigen-window and original training image (for example, eigen-window 1 in image 1, i.e., g_1^1 in Figure 3).

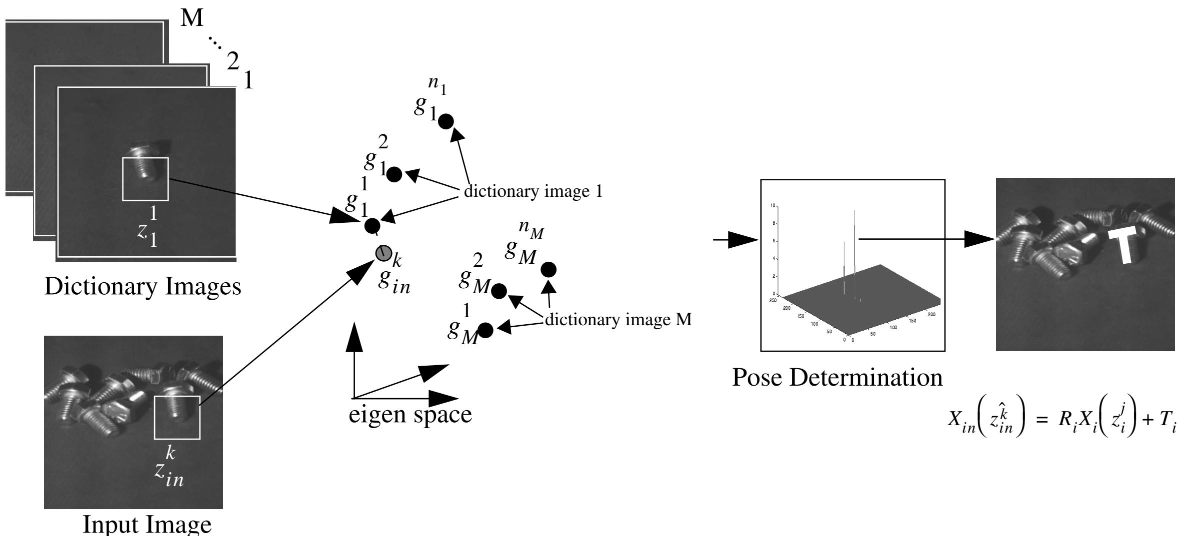


Figure 3 Eigen-Window Technique

2.3.2. Matching Operation

From an input image, a set of sub-window images is obtained:

$$Z_{in} = \begin{bmatrix} z_{in}^1 - c & z_{in}^2 - c & \dots & z_{in}^{n_{in}} - c \end{bmatrix} , \tag{9}$$

such as the white window in the lower left image in Figure3.

The similarity between training eigen-window and input eigen window is evaluated using the distance in the eigen-space. Here, the eigen-space point, g_{in}^k , from an input eigen-window, z_{in}^k , provides the maximum similarity to the point, g_i^j , from a training eigen-window, z_i^j :

$$\hat{g}_{in}^k = min_{i,j}\left(\left\| g_i^j - g_{in}^k \right\|\right) . \tag{10}$$

We will denote $\hat{z}_{in}^k$ as the corresponding training eigen-window to the input window, z_{in}^k and the content of $\hat{z}_{in}^k$ is z_i^j .

2.3.3. Voting Operation

The previous matching operation selects a set of training eigen-windows, $\hat{z}_{in}^{k}$ against input windows. We will group this set into a group of eigen-windows so that all the eigen-windows in one group have the same training image, in_i .

$$\hat{Z}_{in} = \left[\hat{\zeta}_{in_1}\ \hat{\zeta}_{in_2} \dots \hat{\zeta}_{in_M}\right] \tag{11}$$

$$\hat{\zeta}_{in_i} = \{z_i^{\exists j}\} \tag{12}$$

We prepare a pose space for voting from the correspondences. In this operation, we consider only the translation effect. Thus, the space is two dimensional. Here, the size of the pose space is twice the size of the input image size, i.e., 256×256 . Each pose space is prepared to each group, in_i .

Each pair, $z_{in_i}^{k}$ and z_i^{j} , in one group provides the positions of the input eigen-window, $X_{in}\left(z_{in_i}^{k}\right)$, and the training eigen-window, $X_i\left(z_i^{j}\right)$ Then, the difference, $X_{in}\left(z_{in_i}^{k}\right) - X_i\left(z_i^{j}\right)$ is calculated.

The corresponding cell in the two-dimensional pose space to this distance gets a voting. In order to absorb the digitization error, 5×5 cells around the center cell actually get votes from a single correspondence. We repeat this operation using all the correspondences in the group (all the correspondences from the same training image.)

2.3.4. Pose Determination

Some small peaks are due to noise; other prominent peaks are due to actual objects in an input image. By thresholding these peaks, the system eliminate noise peaks and extract actual prominent peaks.

The number of the prominent peaks in the space is equal to the number of objects that have roughly same rotation, but a different translation. By retrieving voted pairs, the system further divides the group into sub-groups so that each sub-group belongs to each prominent peak, and thus, isolated object in the input image.

Although we consider only the translation --- the training set is sampled along the rotation dimension, there is small rotation in object pose due to the sampling interval. To obtain this small rotation and the precise translation value, the system further employes the least square minimization using the pairs in each sub-group:

$$X_{in}\left(z^{k}_{in_i}\right) = R_i X_i\left(z^{j}_{i}\right) + T_i \quad , \tag{13}$$

where R and T denote the small rotation and translation, respectively.

2.4. How to Select Effective Eigen-Windows

One of the problems in the eigen-window technique is how to select the optimal set of eigen-windows. If all the eigen-windows are utilized, 1) the number of eigen-windows becomes very large and storing them requires a large amount of memory space, 2) due to the similarity among eigen-windows, the matching process becomes erroneous. In this section, we will consider the selection method. First, we shows that the local goodness, the traditional trackability criteria, can detect several appropriate corners in an image, without considering the similarity of these appearance. Then, we introduce a new global goodness method based on the similarity measure. Finally, we can get the optimal set of eigen-windows with these two methods.

2.4.1. Local Goodness: Trackability

The window selection may be considered as selecting feature points for object tracking. Some researchers proposed to use the 2×2 matrix as the trackability measure in a window, $X = \begin{bmatrix} x & y \end{bmatrix}^T \in R$ [20].

$$G = \sum_{X \in R} \left(\frac{\partial I}{\partial X}\right)\left(\frac{\partial I}{\partial X}\right)^T \quad . \tag{14}$$

This matrix G, has two eigenvalues λ_1, λ_2. The window is accepted as a good one, if the equation,

$$min(\lambda_1, \lambda_2) > \lambda \quad , \tag{15}$$

holds, where λ is a predefined threshold. This measure works well for detecting all important corners.

Unfortunately, however, the trackability measure does not guarantee the uniqueness of the window. In Figure 4, the window with a corner may be easy to track. However, the same window will be confused between the upper and lower corner. Figure 5 shows the results given by a trackability measure proposed in [20]. The left vertical edges are selected as good windows. Certainly, the confusion occurs among these windows along the edge. Thus, we can conclude that the trackability does provide local goodness around the feature but does not provide a global goodness over the entire image. Next subsection will discuss this global goodness.

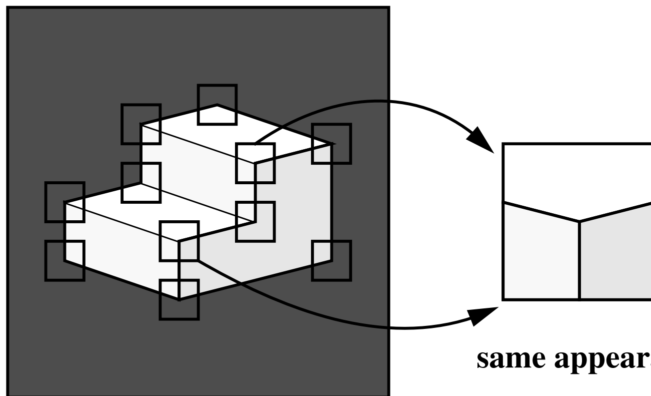


Figure 4   Corner Detection with Trackability

(a) Original Image   (b) Point Feature

Figure 5   Feature Detection with Trackability

2.4.2. Global goodness: Similarity

The global goodness of windows can be determined as the uniqueness of an eigen-window. This uniqueness may be measured using the similarity (difference) among windows. As was discussed in Section 2.3.2, the similarity between training and image eigen-windows was evaluated using the distance in the eigen-space, namely,

$$S_{l,m} = \|g_l - g_m\| \tag{16}$$

where $\|x\|$ denotes the norm of x *using* L1-norm or L2-norm. This similarity $S_{l,m}$ denotes how similar these two dictionaries g_l and g_m are in the eigen-space. We can use the same measure for evaluating the global goodness of the window, i.e., the similarity among training eigen-windows.

The similarity, $S_{l,m}$, between two training eigen windows, g_l *and* g_m, is evaluated using the equation. If this measure is less than a certain threshold T_{sim}, then these two eigen-windows g_l, g_m, are removed from the training set as shown in Figure 6.

$$S_{l,m} = \|g_l - g_m\| \le T_{sim} \tag{17}$$

This elimination of similar eigen-windows can make the size of a training set smaller than the original one. This operation also makes the matching process more robust. Because the set only contains unique eigen-windows, and the matching evaluation will not consider a sum of random contributions from a large number of similar windows.

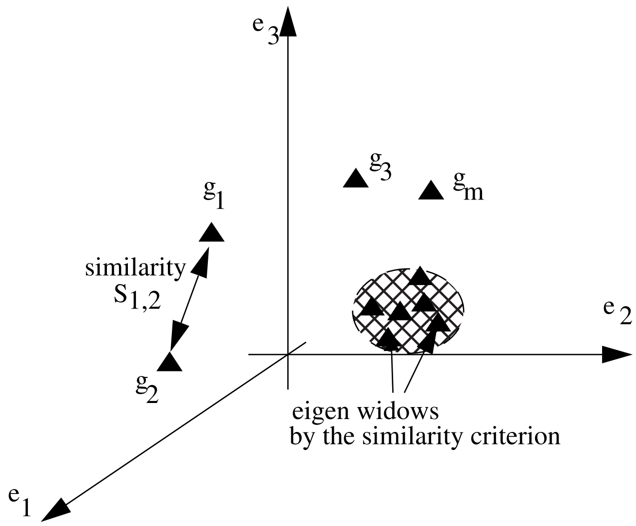
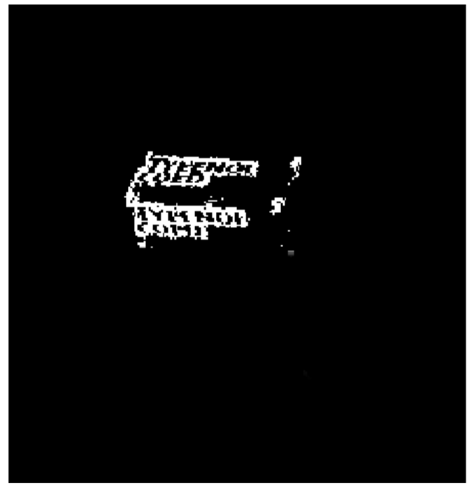
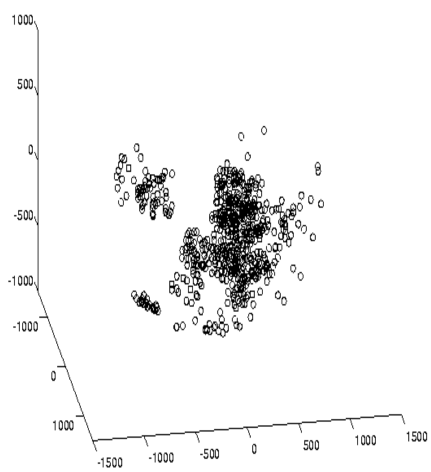


Figure 6 Eigen-space Distribution in 3D.

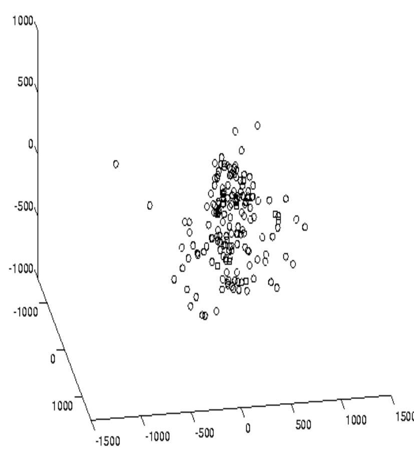
Figure 7 shows an example of the similarity evaluation in eigen-space. At first, the 637 eigen-windows in Figure 5(b), are selected using the local goodness measure, the trackability. Using the global goodness measure, the similarity in the eigen-space, the 178 eigen-windows are selected in Figure 7(a). Note that most of the redundant windows such as those in right corner edges in Figure 5(b) are eliminated in Figure 7(a). Figure 7(b) and (c) show the points projected in the 3D eigen-space. Notice that in Figure 7(c), the distribution of projection is more uniform than that in Figure 7(b).



(a) Eliminated Eigen-Windows



(b) Local Goodness Projection in 3D Eigen-space



(c) Global Goodness Projection in 3D Eigen-space

Figure 7 Eigen-Window Extract with Local Goodness Measure and Global Goodness Measure

3. Experimental Results

This section shows experimental results. Figure 8(a) and (b) show some kinds of the training objects' images and Figure 9 shows one of the input images from which the system determines poses of the objects and the types of objects itself. In order to emphasize the difficulty in modeling, we intentionally use specular objects in this experiment.

3.1. Training Mode

In training mode, a series of images of an object is taken at 10 deg. intervals each for each object using a rotary table and a CCD camera. The resolution of these images is 128×128 pixels and the intensity levels of images have eight bits.

Background noise is eliminated using a threshold value. Only the high intensity pixels remain in the image. In each image, we make eigen windows only on those high intensity pixels. We set the size of an eigen window as 15×15 pixels. Thus, the dimension of this eigen window is 225 (15×15 pixels), which is sufficiently smaller than the original image dimension 15360 (128×128 pixel). From each training image, we obtain 200 eigen windows on average.

The eigenvectors and eigenvalues are calculated from eigen windows using equation (4). Figure 10 shows the eigenvalues λ_i and W_k. This figure shows that up to the 20th dimension of the eigen-spaces are enough to recover the 80% of original image data.

We apply the global measure directly to those eigen-windows. Since specular features are all isolated edges and points, all features are good features in trackability. We did not apply the local measure. We eliminate redundant eigen-windows by evaluating the similarity among them. This process can select the eigen-windows that are very different from each other.

3.2. Run Mode

Figure 11 shows the results in run mode. Here, $\hat{z}_{in}$, depicts those eigen-windows whose counterparts are found in the training eigen-windows. The figures in $\hat{\zeta}_{in_i}$ depicts window positions that come from the same rotation angles. The next column depicts the voting results in the pose space, by the calculation of $X_{in}\left(z_{in_i}^k\right) - X_i\left(z_i^j\right)$. The number of votes represents the probability of the existence of the object in that rotation angle. From these voting results, we can calculate the rotation and translation of the objects as shown in Table 1.

The final column shows the reconvention results superimposed on the original input image. The system can identify seven bolts out of ten in the input image. In this case, the missing bolts are not bright enough to pass the intensity threshold operation for detecting input eigen windows.

3.3. Recognition for Various Objects

The algorithm also works well for the recognition of various objects, such as multi various objects as shown in Figure 14. In this case, the image training set is composed of each object's training set as shown in Figure 13(a) and (b). Then we can obtain the recognition result Figure 16, 17 and Table 2 with the eigen-window analysis.

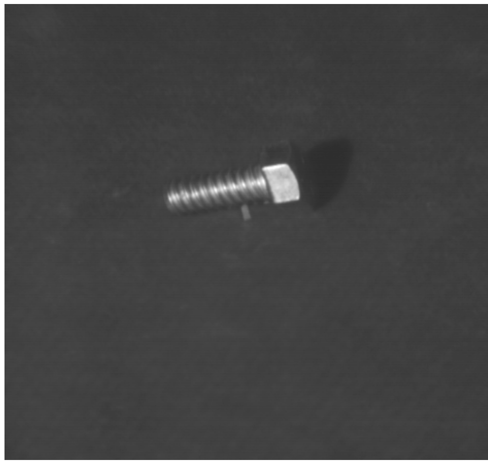
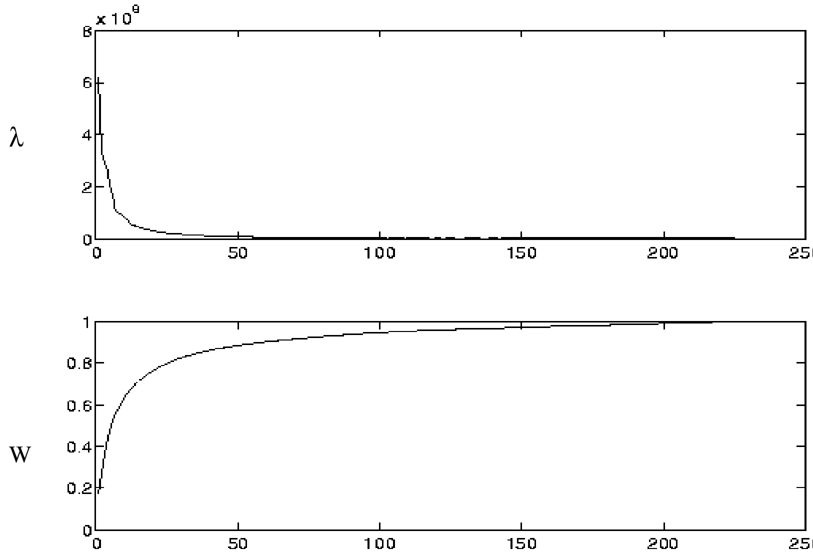


Figure 8 A Sample of Training Images. **Figure 9 Multi Objects Image.**



dimension of eigen-space.

Figure 10 Eigenvalue

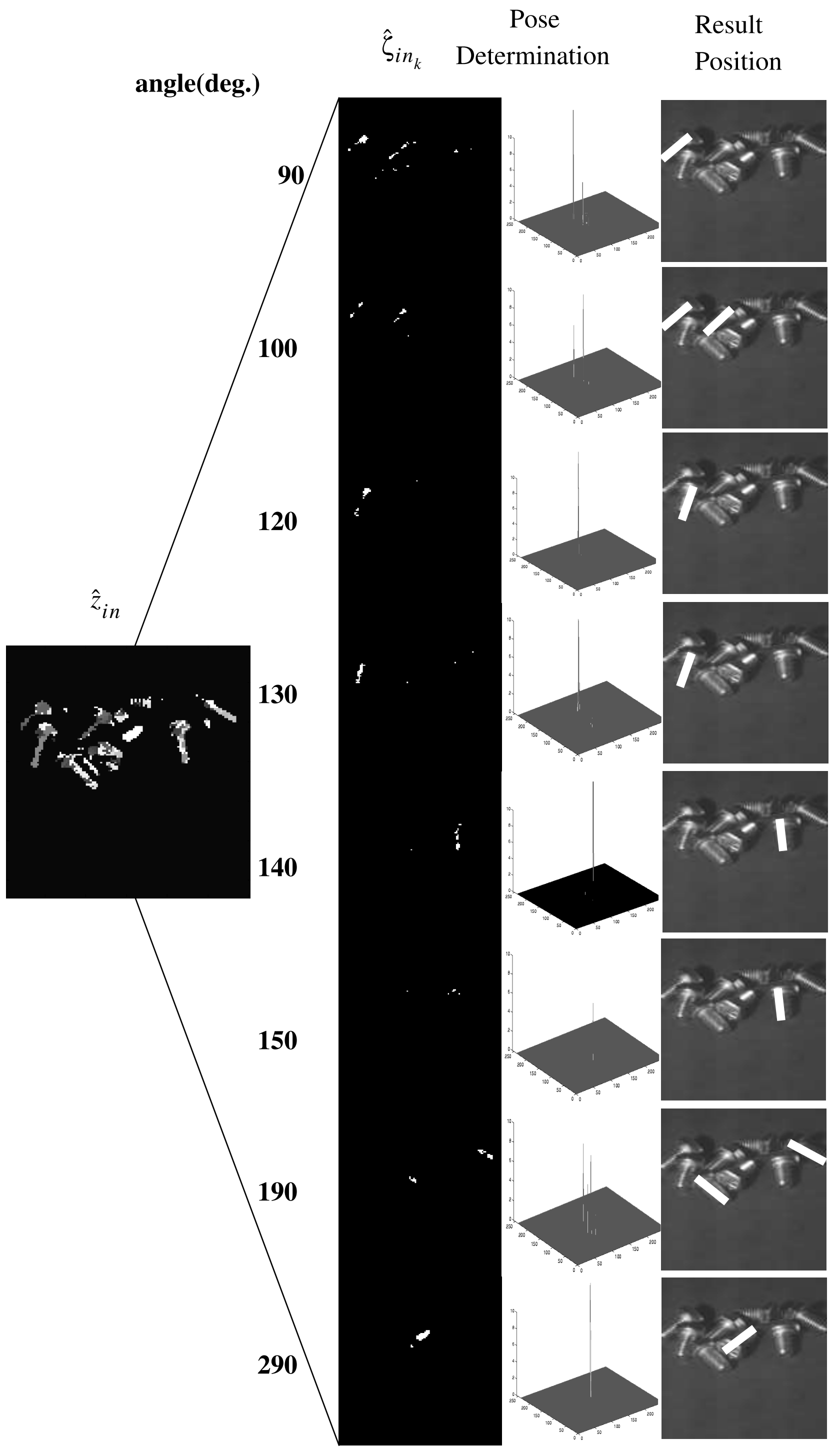


Figure 11 Components of the Recognition Result.

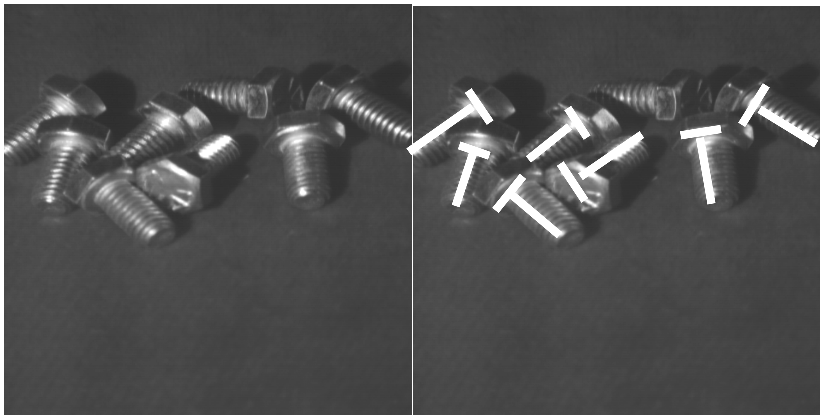


Figure 12 Recognition Results

Table 1: Position Parameters for each Objects

angle(deg.)	rotation		translation
90 -1	0.9809	-0.0273	-44.3796
	-0.0125	1.0223	17.1245
100 -1	1.0013	-0.0758	-12.4194
	-0.0905	1.0608	14.0588
100-2	0.4605	0.4365	-49.1039
	-0.3097	1.2900	14.8709
120-1	1.0660	-0.1010	-42.7534
	-0.0018	1.0081	4.5029
130-1	1.1684	0.0310	-60.9514
	0.1619	0.9606	-2.9618
140-1	1.0516	-0.0069	18.0010
	0.1367	0.9957	-1.9844
150-1	1.0000	-0.0000	20.0000
	-0.0000	1.0000	7.0000
190-1	0.5325	-0.2867	86.1911
	0.2337	1.1433	-8.0955
190-2	1.5127	-0.0851	-50.2733
	-0.1456	1.0785	11.2708
290-1	1.0431	-0.0423	-14.4051
	-0.1156	1.1287	-8.9405

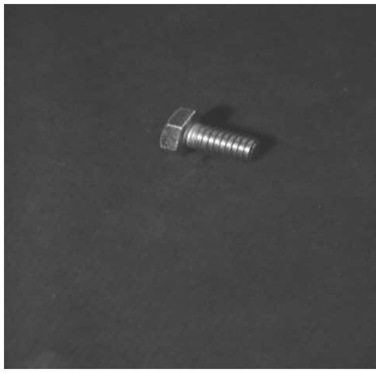


Figure 13 (a) A Sample of Training Bolt Images.

Figure 13 (b) A Sample of Training BNC Images.

Figure 14 Multi Objects Image.

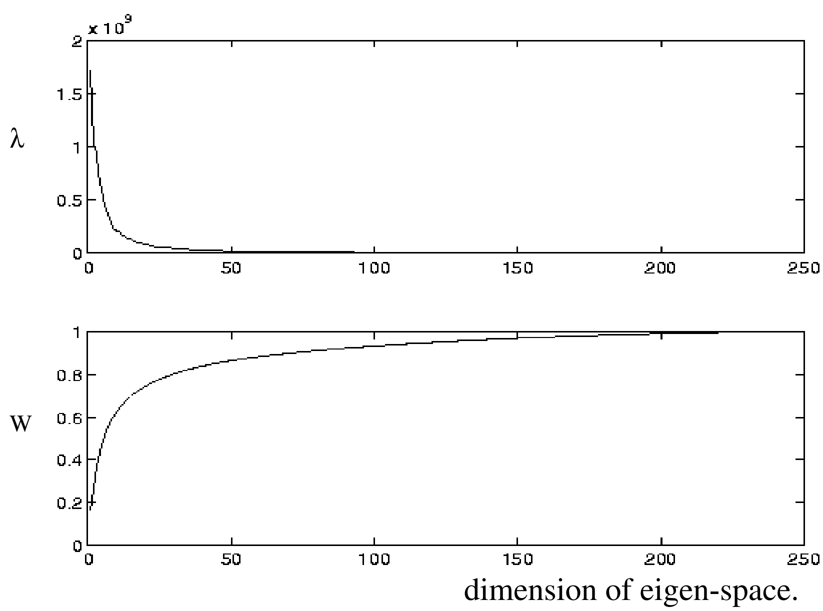


Figure 15 Eigenvalue

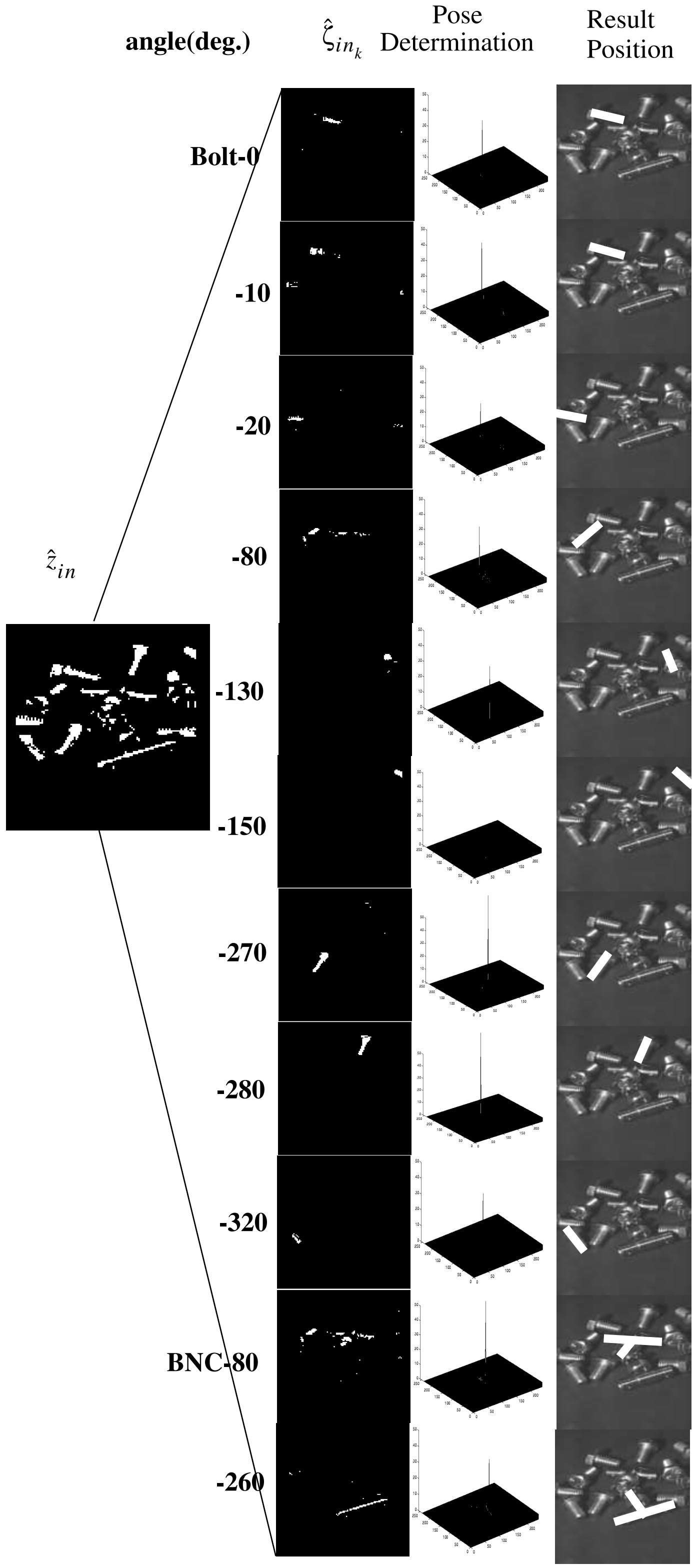


Figure 16 Components of the Recognition Result.

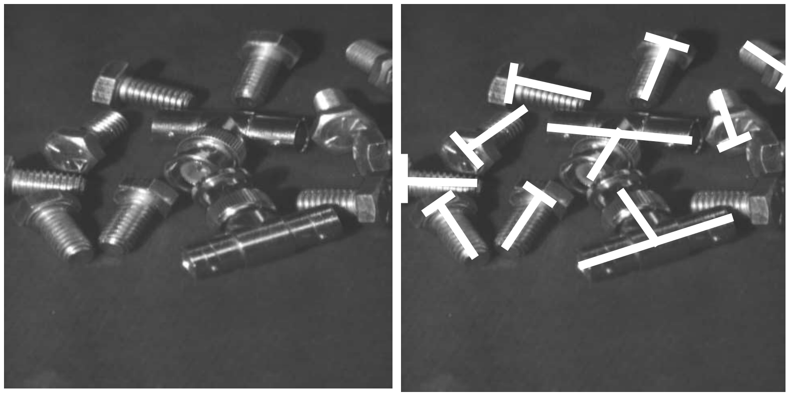


Figure 17   Recognition Results

Table 2: Position Parameters for each Objects

angle(deg.)	rotation		translation
Bolt -0	0.9751	-0.1494	-10.3179
	0.0059	1.0556	13.4182
Bolt -10	0.9427	0.0344	-21.7046
	-0.0411	1.0342	18.5854
Bolt -20	0.9128	-0.0419	-47.9302
	-0.0978	0.8219	9.9277
Bolt -80	1.2884	-0.3037	-34.0457
	0.0142	0.9635	9.8536
Bolt -130	1.0168	0.1180	22.4607
	0.0000	1.0000	16.0000
Bolt -150	0.8655	-0.0846	59.4585
	-0.0112	1.0776	28.1560
Bolt -270	1.0022	0.0453	-40.1660
	-0.0234	1.0555	-27.0799
Bolt -280	0.9482	0.0347	5.0386
	-0.0088	1.0011	24.5654
Bolt -320	1.0000	0.0000	-54.0000
	0.0000	1.0000	-29.0000
BNC -80	1.1579	-0.4820	29.8313
	0.0567	1.0709	-10.3385
BNC -260	0.8135	0.1992	-3.0655
	-0.0390	1.0683	-26.8598

4. Conclusions

This paper describes a novel method, referred to as eigen-window, to extend the eigen-space analysis to be able to recognize partially occluded objects. To reduce the redundancy among eigen-windows, a similarity measure among eigen-windows was developed. We have implemented the system and verified the validity of this method by experiments that involve multiple specularity objects.

Recently, several researchers are investigating in the invariance of objects features for the pattern matching problem. This kind of invariance allow you to recognize the object with input images which have some object rotation and scale. In future work, we would like to consider the invariant of the image in this eigen-window analysis.

References

[1] Y.Fukada, H.Doi, K.Nagamine, and T.Inari, “Relationships-Based Recognition of Structural Industrial Parts Stacked in Bin”, Robotica, Vol.2, pp.147-154, 1984.

[2] J.R.Birk, R.B.Kelly, and H.A.S.Martines, “An Orienting Robot for Feeding Workpieces Stored in Bins”, IEEE Trans. Systems Man Cybernet. SMC-11(2), pp.151-160, 1981.

[3] R.C.Bolles, P.Horaud, and M.J.Hannah, “3DPO: Three-Dimensional Parts Orientation System”, Proc. of the Intern. Joint Conf. on Artificial Intelligence, Karlsruhe, West Germany, pp.1116-1120, 8-12 August.

[4] B.K.P. Horn and K.Ikeuchi, “The Machanical Manipulation of Randomly Oriented Parts”, Scientific American, Vol.251, No.2, pp.100-111, 1984.

[5] S.A.Hutchinson and A.C.Kak, “{SPAR}: A Planner that Satisfies Operational and Geometric Goals in Uncertain Environments”, AI Magazine, Vol. 11, No.1, pp.31-61, 1990.

[6] G.J.Agin and O.R.Duda, “SRI Vision Research for Advanced Industrial Automation”, Proc. 2nd U.S.A-Japan Computer Conf., pp.113-117, 1975.

[7] Matthew A. Turk and Alex P. Pentland, “Face Recognition Using Eigenfaces”, Proc. CVPR 1991, pp.586-591, 1991

[8] Matthew A. Turk and Alex P. Pentland, “Eigenfaes for Recognition”, Journal of Cognitive Newroscience, Vol.3, No.1, pp.71-86, 1991

[9] Stan Sclaroff and Alex P. Pentland, “A Model Framework for Correspondence and Description”, Proc. 4th Int’l Conf. on Computer Vision, pp.308-313, 1993.

[10] Alex P. Pentland and Bradley Horowitz, “Recovery of Nonrigid Motion and Structure”, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.13, No.7, pp.730-742, 1991.

[11] Alex P. Pentland, Baback Moghaddam, Trad Starner, “View-Based and Modular Eigenspaces for Face Recognition”, IEEE Conference on Computer Vision and Pattern Recognition, 1994.

[12] Baback Moghaddam and Alex P. Pentland, “Face Recognition using View-Based and Modular Eigenspaces”, Automatic Systems for the Identification and Inspection of Humans, SPIE Vol.2277, 1994.

[13] Baback Moghaddam and Alex P. Pentland, “Probabilistic Visual Learning for Object Detection”, The 5th International Conference on Computer Vision, 1995.

[14] Stan Sclaroff and Alex P. Pentland, “Model Matching for Correspondence and Recognition”, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.17, No.6, pp.545-561, 1995.

[15] M. Kirby and L. Sirovich, “Application of the Karhunen-Loeve Procedure for the

Characterization of Human Faces", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.12, No.1, pp.103-108, 1990

[16] Hiroshi Murase and Shree K. Nayar, "Visual Learning and Recognition of 3-D Objects from Appearance", International Journal of Computer Vision, Vol.14, No.1, pp.5-24, 1995.

[17] Hiroshi Murase and Shree K. Nayar, "Learning and Recognition of 3-D Objects from Brightness Images", Working Notes of AAAI Fall Symposium Series, pp.25-29, 1995.

[18] M. Uenohara and T. Kanade, "Vision-Based Object Registration For Real-Time Image Overlay", International Journal of Computers in Biology and Medicine, Vol. 25, No. 2, pp. 249-260, March 1995.

[19] M. Uenohara and T. Kanade, "Vision-Based Object Registration for Real-Time Image Overlay", Proceedings of the First International Conference on Computer Vision, Virtual Reality and Robotics in Medicine, Nice France, April 1995.

[20] C. Tomasi and T.Kanade, "Shape and Motion from Image Streams: a Factorization Method-2. Point Features in 3D Motion", Technical Report CMU-CS-91-105, Carnegie Mellon University, Pittsburgh, PA January 1991.