Visual Learning and Object Verification with Illumination Invariance

Kohtaro Ohba
Mechanical Engineering Laboratory
MITI
Tsukuba, 305, JAPAN

Yoichi Sato
Institute of Industrial Science
The University of Tokyo
Tokyo, 106, JAPAN

Katsusi Ikeuchi
Institute of Industrial Science
The University of Tokyo
Tokyo, 106, JAPAN

Abstract

This paper describes a method for recognizing partially occluded objects to realize a bin-picking task under different levels of illumination brightness by using the eigen-space analysis. In the proposed method, a measured color in the RGB color space is transformed into the HSV color space. Then, the hue of the measured color, which is invariant to change in illumination brightness and direction, is used for recognizing multiple objects under different levels of illumination conditions. The proposed method was applied to real images of multiple objects under different illumination conditions, and the objects were recognized and localized successfully.

1 Introduction

Recently, visual learning methods based on the eigen-space analysis have shown a potential to realize the robust object recognition system, [1] and [2]. And also, it can solve some of these difficulties, such as requirement for real-time processing, difficulty in segmentation, and difficulty in obtaining appropriate object models. In the eigen-space analysis, object models are *learned* from a series of images taken in the same environment as in the recognition mode.

Although promising, the current eigen-space analysis is based on the assumption that objects are not occluded in images. Therefore, to apply the eigen-space analysis for partially occluded objects, we proposed to divide appearances into small windows, referred to as *eigen-windows* [4] and to apply eigen-space analysis to each eigen-window.

One drawback of the eigen-window method is that only a limited number of images can be used for learning object models, and therefore, all possible illumination directions cannot be taken into account.

In this paper, to overcome that drawback, we propose to use the color measurement *hue*, which is illumination invariant, in the eigen-window method. To demonstrate the effectiveness of the proposed method, we applied the method to real images taken under different illumination directions and brightness.

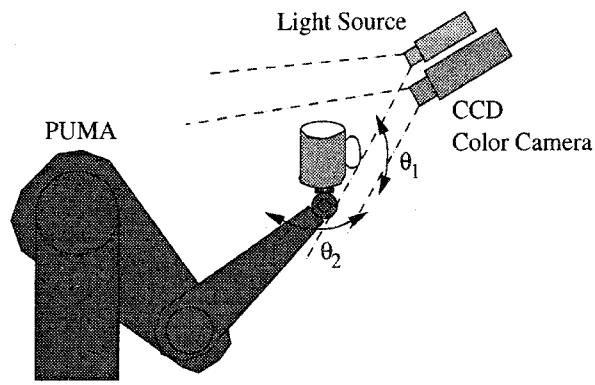


Figure 1: Experimental Setup

2 Eigen-Window Method

2.1 Eigen-Space Technique

Let M be the number of the images $z_1, z_2, \cdots, z_M$ in a training set related to each rotation of view points θ_1 and θ_2, as shown in Figure 1. Each image z_i, with dimensions $N \times N$, has been converted into a column vector of length N^2.

By subtracting the average brightness c of all the images, we obtain the training matrix of the size of N^2 by M,

$$Z = [z_1 - c, z_2 - c, \cdots, z_M - c]. \quad (1)$$

This covariance matrix $Q = ZZ^T$ provides a series of eigenvalues λ_i and eigenvectors $e_i (i = 1, \cdots, N^2)$, where each corresponding eigenvalue and eigenvector pair satisfies:

$$\lambda_i e_i = Q e_i. \quad (2)$$

For reducing memory requirement, we ignore eigenvectors corresponding to small eigenvalues $e_i (i > l)$. These eigenvectors do not affect object recognition results significantly. Once we obtain the remaining eigenvectors, we can construct the eigenvector matrix

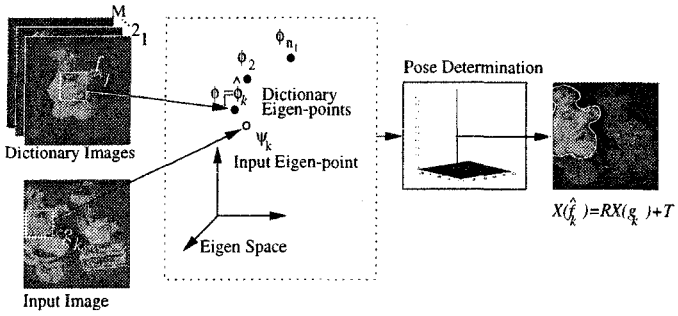


Figure 2: Eigen-Window Technique.

$E = [e_1, e_2, \cdots, e_l]$ which projects an image z_i (dimension N^2) into the eigen-space as an *eigen-point* g_i (dimension l).

$$g_i = E^T(z_i - c). \tag{3}$$

The eigen-space analysis can drastically reduce the dimension of the images (N^2) to the eigen-space dimension (l) while preserving enough dominant features to reconstruct the original images.

2.2 Eigen-Window Technique

To reduce the disturbance effects such as image shift and occlusion, we propose to select small windows in the original images. Each of the selected small windows is then analyzed by using the eigen-space analysis as described in the previous section. We call this method *the eigen-window method.* Figure 2 shows the overview of the method.

Training Eigen-Windows. The training set of eigen-windows is given as:

$$F = \left[F^1, F^2, \cdots, F^M\right], \tag{4}$$

where F^i denotes the collection of eigen-windows from the ith training image. Each F^i has the form $[f_1 - c, f_2 - c, \cdots, f_{n_i} - c]$, where f_j denotes the jth eigen window in the ith training image; n_i denotes the number of eigen-windows in the ith image; and c is the average intensity value across all eigen-windows in the whole training set. In Figure 2, the white square denotes one of the training eigen-windows.

Matching Operation. From an input image, a set of input eigen-window images is obtained:

$$G = [g_1 - c, g_2 - c, \cdots, g_n - c], \tag{5}$$

such as the white window in the lower left image in Figure 2.

The similarity between a training eigen-window and an input eigen-window is evaluated by using the distance between them in the eigen-space. Given an input eigen-point ψ_k projected from input eigen-window g_k using equation (3), we try to find a training eigen-point $\hat{\phi}_k$ from all training eigen-points ϕ projected from all training eigen-windows f. The training eigen-point is the one with the maximum similarity defined as:

$$\hat{\phi}_k = \arg\min_{\forall \phi}(\|\psi_k - \phi\|), \tag{6}$$

We denote the eigen-window that is projected to $\hat{\phi}_k$ as $\hat{f}_k$. The eigen-window $\hat{f}_k$ corresponds to the input eigen-window g_k.

Voting Operation. The previous matching operation selects a set of training eigen-windows, $[\hat{f}_1, \hat{f}_2, \cdots, \hat{f}_n]$ corresponding to input eigen-windows. We now sort the selected training eigen-windows into each groups, which contains windows that comes from the same training images, from a corresponding training image:

$$\left[\hat{F}^1, \hat{F}^2, \cdots, \hat{F}^M\right], \tag{7}$$

where

$$\hat{F}^i = \{\hat{f} | \hat{f} \; comes \; from \; training \; image \; i\} \tag{8}$$

We then prepare a pose space for voting from the established correspondences. In this operation, we consider only translation, and therefore the space is two dimensional corresponding to a $[x, y]$ location of eigen-windows. The size of the pose space is set to be twice that of the input image size, e.g., 256×256 in our examples. The pose space is prepared separately for each group $\hat{F}^i$.

Using each correspondence, we can compute the difference of the training eigen-window's location $X(\hat{f}_k)$ and the input eigen-window's location $X(g_k)$. ($X(f)$ represents $[x, y]$ location of the eigen-window f.) The difference is given as $X(g_k) - X(\hat{f}_k)$.

Then, in the pose-space, the cell that represents this distance, $X(g_k) - X(\hat{f}_k)$, gets a vote. To avoid the digitization error, all of the 5×5 neighbor cells around the center cell get a vote from a single correspondence. We repeat this operation, using all correspondences for a group $\hat{F}^i$ (all the correspondences from the same training image.) Then, we obtain a resulting pose-space for each of the groups $\hat{F}^i$.

Pose Determination. By retrieving voted pairs, we further divide the group $\hat{F}^i$ into sub-groups, each of which belongs to each prominent peak, i.e., an isolated object in the input image.

Since the training set is sampled along the rotation dimension, there exists a side effect of small object rotation due to the finite sampling interval. To obtain the rotation and the translation precisely, we refine the pose estimate via a least square minimization, using the pairs in each sub-group.

$$X(\hat{f}_k) = RX(g_k) + T, \tag{9}$$

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

2.3 Selection of Effective Eigen-Windows

In the eigen-window method, it is very important and also difficult to select an optimal set of training eigen-windows. In this section, we introduce three criteria for selecting the optimal set of eigen-windows: *detectability*, *uniqueness*, and *reliability*.

Detectability. For initial selection of eigen-windows in images, algorithms to select feature points for object tracking can be used. Tomasi *et.al.* proposed to use the following 2×2 matrix as the trackability measure for a window $X = [x, y]^T$ [3].

$$G = \sum_{X \in R} (\frac{\partial I}{\partial X})(\frac{\partial I}{\partial X})^T. \quad (10)$$

where I represents pixel intensities inside the window.

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

$$min(\lambda_1, \lambda_2) > \lambda. \quad (11)$$

holds, where λ is a predefined threshold. The measure works well for detecting most of the important corners.

Uniqueness. By using the detectability measure, we can select windows which contain important features. Unfortunately, however, the detectability measure does not guarantee the global uniqueness of the selected windows.

The uniqueness of each window can be measured by computing similarity among eigen-windows. As discussed in Section 2.2, the similarity between a training eigen-window and an eigen-window in an input image is computed by using the distance between them in the eigen-space.

We can use the same measure for evaluating the global goodness of selected windows, i.e., the similarity among training eigen-windows.

$$S_{l,m} = \|\phi_l - \phi_m\| \leq T_{sim}. \quad (12)$$

The similarity $S_{l,m}$ between two training eigen-points ϕ_l and ϕ_m, which are projected from two eigen-windows, is evaluated by using the equation. If the computed similarity is less than a certain threshold T_{sim}, then the two eigen-windows are discarded from the training set.

Reliability. The reliability of an eigen-window for object recognition can be inferred by measuring how much the eigen-point which corresponds to the eigen-window moves in the eigen-space while an object is moved slightly.

Please see more details about these criteria in [4].

3 Illumination Invariance

In our previous work [4], we have shown that the eigen-window method can successfully recognize and localize an object in b/w input images which contain multiple objects with specularity, even if the input images contain significant amount of noise, occlusion, image shifting and scaling change.

However, the method was based on an assumption that the location and brightness of a light source are fixed. Therefore, the method did not take into accounts shading variation such as highlights on object surfaces. For instance, if an object exhibits specularity, the object appearance can change drastically with different illumination directions, which confuses recognition and localization of the object.

To overcome this limitation, we propose to use an illumination invariant measure for the eigen-window method. By using the illumination invariant measure, the eigen-window method can be used successfully for recognizing and localizing multiple objects under different illumination conditions.

3.1 Illumination Invariance: Hue

Instead of black-and-white intensity images, we use RGB color images in the modified eigen-window method. Actually, several pieces of research works were done on the color indexing in the past [5] - [7], but we would like to use the *hue* criterion for its simpleness, and a color image measured in the RGB color space is converted to a HSV image (H: Hue, S: Saturation, V: Value). In these three parameters, the hue parameter is the value which represents color information, e.g., without brightness. Therefore, the hue is not affected by change of the illumination brightness and direction if the following two conditions hold: 1) the light source color can be expected to be almost white, and 2) a saturation value of object color is sufficiently large.

The original color of object X is transferred to be $X' = s \cdot X + t \cdot I$ by the change in diffuse shading and specularity as shown in Figure 3. s and t represent a relative strength of the diffuse reflection component and the specular reflection component of the color X', respectively. If the two conditions mentioned above are true, then the hue of X' remains the same as that of X.

In Figure 3, object color is represented by three color components S_1, S_2, and S_3. In the RGB color space, those three color components are Red, Green and Blue. Then, the light source color I is given as $I = (1, 1, 1)$. To define hue, saturation and intensity, one pair from three components, Red, Green, and Blue, have to be assigned to S_1' and S_2'. Usually, Red and Green are assigned as $S_1' = R$ and $S_2' = G$.

We conducted a simple experiment using a color test

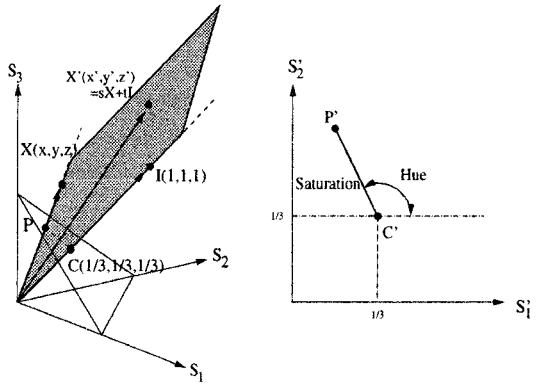


Figure 3: HSV Space.

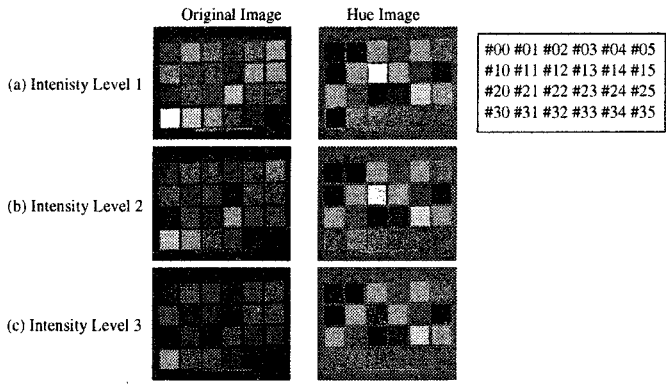


Figure 4: Illumination Constancy with a Color Test Pattern Image.

chart to see how hue is affected under different levels of illumination brightnesses. The result is shown in Figure 4 and Figure 5. In Figure 5 (b), we can see that hue remains almost constant over a wide range of illumination brightness for many color blocks.

However, for some color blocks, the value of hue does change with different levels of illumination brightnesses. For instance, the black-white color blocks in the last row of the color chart (color blocks #30-#35), red (color block #12) and magenta (color block #24).

That is because the saturation of color blocks #30-#35 is not sufficiently large, i.e., they are very close to gray. Also, hue has a discontinuity at 0 and 2π. That is the reason for unstable hue of the color blocks #12 and #24.

To obtain the value of hue reliably, we propose to use three criteria: *intensity value*, *saturation*, and *phase*.

Intensity Value. To eliminate the background noise, we apply a threshold value for the intensity value

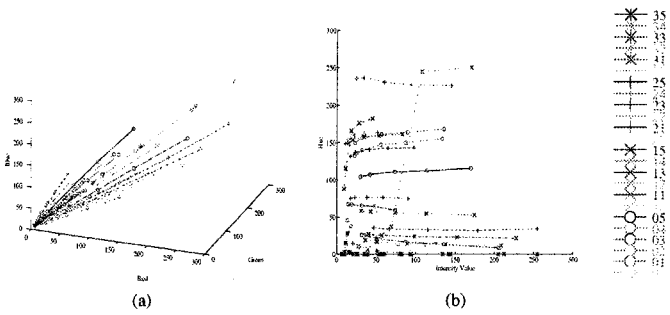


Figure 5: Color Elements. (a) RGB space; (b) Hue-Intensity space

as

$$if\ V < V_t\ then\ H = 0, \tag{13}$$

where V, V_t, and H are an intensity value, the threshold value, and a hue value, respectively. If measured color is not bright enough, the color is discarded. Then, the hue value is set to a predetermined value, i.e., 0.

Saturation. One of the problems shown in the example in Section 3.1 is that, if object color is close to gray, then hue value of the color is not stable. The reason is that, if the color is almost gray, the object color in $S'_1S'_2$ plane exists around the point C' in Figure 3. That means the hue angle cannot be determined robustly in the face of image noise. Therefore, measured color should be discarded if the saturation value is less than a certain threshold S_t:

$$if\ S < S_t\ then\ H = 0, \tag{14}$$

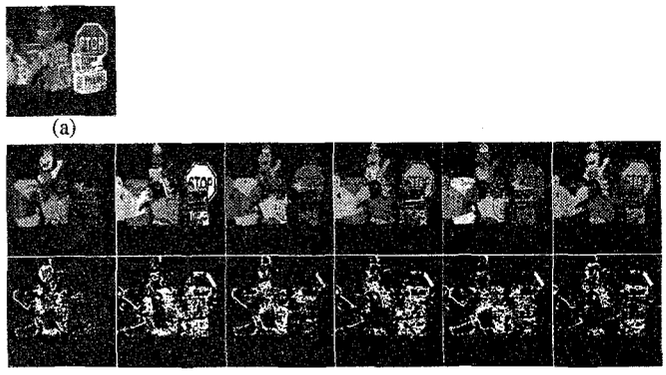
where S is the saturation value. Using the equation, measured color close to gray is discarded in the image.

Phase. The other problem shown in the example in Section 3.1 is that color close to red has a hue value near its discontinuity. The range of hue value is from 0 to 2π, and it has discontinuity at 0 and 2π. We avoid the discontinuity effect by using the phase threshold value ΔP_t as:

$$if\ H < \Delta P_t\ or\ ||H - 2\pi|| < \Delta P_t\ then\ H = 0. \tag{15}$$

In the examples shown in Figure 4 and Figure 5, the red color element may be neglected with this criterion.

It is important that the discontinuity of hue value depends on the selection of the color components S'_1 and S'_2. In the next section, we discuss how to select the color components S'_1 and S'_2 to be able to find more windows.



(a)

(b) RG 1002pts. (c) RB 2039pts. (d) GR 1295pts. (e) GB 1683pts. (f) BR 1525pts. (g) BG 1072pts.

Figure 6: Image Invariance and Window Selection. (a) Original Image; (b) Hue Image and Feature Points with RG; (c) RB; (d) GR; (e) GB; (f) BR; (g) BG

3.2 How to Select Color Components S_1' and S_2'

Usually, the two color components S_1' and S_2' are set to R and G. But if the R and G factors are used for the two color components, the discontinuity of hue appears around the color red as described in the previous section. Therefore, if red is the most important component for recognizing the objects, the use of R and G for S_1' and S_2' is not desirable.

In this section, we show how to choose the S_1' and S_2' from RGB components so that we can select more windows to be used as eigen-windows as described in Section 2.

There are six combinations for the selection of S_1' and S_2' from the RGB components. Figure 6 shows the result of window selection by using each combination of S_1' and S_2'. In the figure, RG represents that $S_1' = R$ and $S_2' = G$.

The windows in the hue images were selected by using the corner detector algorithm as described in Section 2.3. So if a hue image does not have enough contrast, fewer windows are selected. In this example, the largest number of windows was selected for the case of $S_1' = R$ and $S_2' = B$ in Figure 6. Intuitively, that result indicates that there are not many green color components in the example image.

4 Experimental Results

The proposed method was used for recognition and localization of objects in three test cases. In the first case, the same illumination condition was used both for training and for input images. In the second case, input images were taken under different levels of illumination brightnesses. In the last case, input images were taken with different light source locations.

4.1 Object Recognition and Localization with Hue Image

First, a set of training eigen-windows was obtained as described in Section 2. The training images were taken at $\theta_1 = [-20, 0, 20]$ and $\theta_2 = [0, 10, 20, \cdots, 350]$ for three different objects, *mug*, *bird*, and *tylenol*. We refer to the original images as $type(\theta_1, \theta_2)$. For example, the image $mug(-20, 60)$ denotes the image for the mug taken at the position $\theta_1 = -20deg$ and $\theta_2 = 60deg$. One hundred eight images were taken for each of the objects by using the experimental setup shown in Figure 1.

Then, eigen-windows were selected in each training image by using the detectability, similarity and reliability measurements as described in Section 2.3. The number of eigen-windows for each of the objects was initially more than 8,000. After the three measurements were applied, less than 2,000 of the training eigen-windows were finally obtained. Then, these eigen-windows were projected to produce eigen-points according to the equation (3).

One input image containing multiple objects was taken as shown in left hand side of Figure 7. In the input image, there are 7 objects, *duck*, *mug*, *barney*, *bird*, *stop-sign*, *tylenol*, and *tylenol-cold*. First, eigen-windows were selected in the input image by using the detectability measure. Then, we established correspondences between the input eigen-windows and the training eigen-windows by using the similarity between their eigen-points according to the equation (12).

The recognition and localization results are shown in figures in the middle column in Figure 7. The figures in the right column show the resulting pose spaces. Also, the obtained affine parameters and standard deviations in the pose space are shown. As we can see, each object's type, pose, and location were successfully obtained.

4.2 Effect of Illumination Brightness Change

The same training eigen-window set was applied to input images taken under a wide range of illumination brightness. Figure 8 shows the result.

The original color images are shown in the left column, and the computed hue images are shown in the middle column. The localization and recognition results are shown in the right column. The affine parameters and standard deviations of the pose space are also given in the figure.

The hue images did not change significantly with different levels of illumination brightness. The main difference between the hue image for the brightest illu-

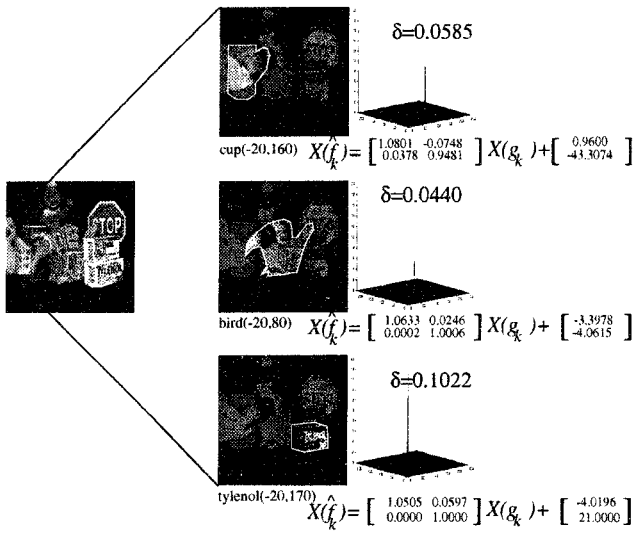


Figure 7: Recognition Result

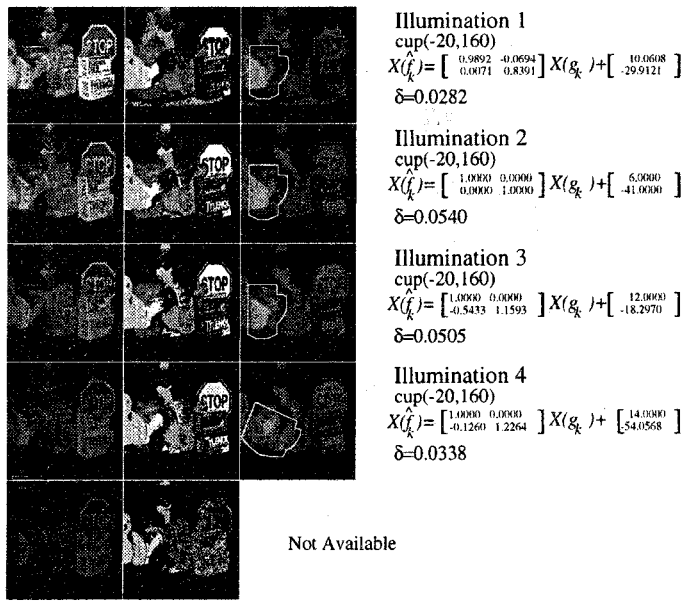


Figure 8: Object Recognition Results with Illumination Change

mination and that for the darkest illumination is that hue values were not computed over a large portion of the object surfaces. This is because intensity values were so small that hue values were set to zero as the background value according to the equation (14).

The experimental results show that the proposed method works even when input images are taken with different levels of illumination brightness. The object was recognized and localized successfully.

4.3 Effect of Different Light Source Positions

The proposed method was also applied to input images taken with different light source locations. As the light source position changes, the appearance of objects in input images changes drastically. Therefore, changing light source position makes recognition and localization of objects even harder than changing illumination brightness.

In this experiment, four different light source positions were used as shown in Figure 9. The left column images of Figure 10 show the input images taken with each of the four light source positions. The middle column images of Figure 10 show the obtained hue images. The right column images present the recognition and localization results. The affine parameters and standard deviations of pose-space are also shown in the figure.

Note that, in this experiment, there was no ambient illumination. Hence, the appearance of the objects change significantly with different light source positions. Nevertheless, the mug was correctly recognized and localized except in the input image for the light position 1. In this case, hue values were not obtained over a large portion of the object surface because of shadow casting on the surface.

5 Conclusion

In this paper, by using hue, which is an illumination invariant measure, the eigen-window method was extended further for recognition and localization of objects in images taken under changing illumination conditions. To use hue information of input images reliably, we introduced three criteria for computing hue values: intensity value, saturation, and phase.

The proposed method was applied to real images, and the method recognized and localized objects suc-

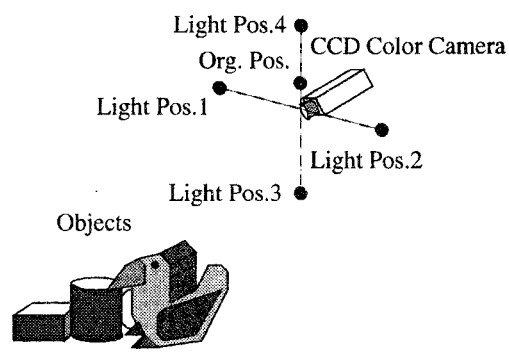


Figure 9: Light Source Position

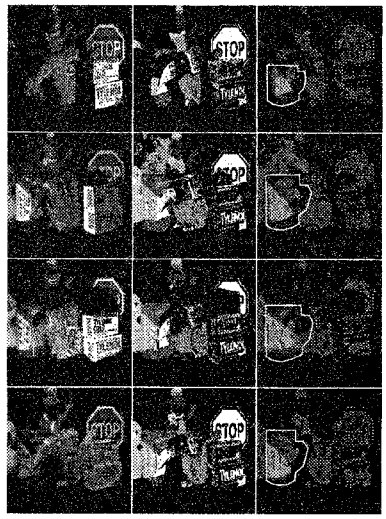


Figure 10: Effect of Light Source Direction

cessfully even in images taken under significantly different illumination conditions.

References

[1] M. A. Turk and A. P. Pentland, "Face Recognition Using Eigenfaces," Proc. CVPR 1991, pp.586-591, 1991.

[2] H. Murase and S. 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.

[3] C. Tomasi and T. Kanade, "Shape and Motion without depth," *Proc. of the Third International Conference in Computer Vision*, Osaka, Japan, December 1990.

[4] K. Ohba and K. Ikeuchi, "Recognition of the Multi Specularity Objects using the Eigen-Window," *Proceeding of International Conference on Pattern Recognition*, August 1996.

[5] G. Healey and D. Slater, "Global color constancy: recognition of objects by use of illumination-invariant properties of color distribution," *Journal of Optical Society of America*, Vol.11, No.11, pp.3003-3010, Nov. 1994.

[6] M. J. Swain, "Color Indexing," *International Journal of Computer Vision*, Vol.7, No. 1, pp.11-32, 1991.

[7] B. V. Funt and G. D. Finlayson, "Color Constant Color Indexing," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol.17, No.5, pp.522-529, May 1995.