

# Vehicle Classification System with Local-Feature Based Algorithm Using CG Model Images

Tatsuya YOSHIDA<sup>†</sup>, Shirmila MOHOTTALA<sup>†</sup>, *Nonmembers*, Masataka KAGESAWA<sup>†</sup>,  
and Katsushi IKEUCHI<sup>†</sup>, *Members*

**SUMMARY** This paper describes our vehicle classification system, which is based on local-feature configuration. We have already demonstrated that our system works very well for vehicle recognition in outdoor environments. The algorithm is based on our previous work, which is a generalization of the eigen-window method. This method has the following three advantages: (1) It can detect even if parts of the vehicles are occluded. (2) It can detect even if vehicles are translated due to veering out of the lanes. (3) It does not require segmentation of vehicle areas from input images.

However, this method does have a problem. Because it is view-based, our system requires model images of the target vehicles. Collecting real images of the target vehicles is generally a time consuming and difficult task. To ease the task of collecting images of all target vehicles, we apply our system to computer graphics (CG) models to recognize vehicles in real images. Through outdoor experiments, we have confirmed that using CG models is effective than collecting real images of vehicles for our system. Experimental results show that CG models can recognize vehicles in real images, and confirm that our system can classify vehicles.

*key words:* *eigen-window method, vector quantization, CG image, vehicle recognition, vehicle classification.*

## 1. Introduction

The main purpose of vehicle detection is to measure the number of vehicles at each sensing point for flow estimation and prediction. As a result, point-oriented sensors, i.e., ultrasonic sensors or loop detectors, have often been used. But these sensors can detect only whether a vehicle is present at a certain place; they cannot collect other important traffic information such as velocity, size and type of vehicles.

Recently, image-processing sensors have become available for ITS applications. Not only can those sensors measure the number of vehicles but they can also measure velocity; in addition, they have the potential to detect traffic accidents.

One of the common problems with image-based vehicle recognition systems is their inability to segment vehicle areas from input images. Various researches on updating background (e.g., [1]) have been conducted with the result that, by using the background subtraction method, it is easy to obtain moving object areas.

But in cases where vehicles are occluded by other vehicles, it is difficult to segment each vehicle area. We have already proposed an algorithm to recognize specified vehicles in infrared images based on local-features[9]. The algorithm can solve the problem of segmenting vehicles.

Despite the fact that a number of vehicle classification systems exist, most of them distinguish large vehicles such as buses, trucks, and normal passenger vehicles from one another. But in applications for systems requiring precise vehicle types and locations, e.g., electronic toll collection (ETC), mobile operation control systems (MOCS), commercial vehicle operations (CVO), and public transportation priority systems (PTPS), it might be necessary to classify vehicles into many groups.

In this paper, we propose a system that uses image-processing sensors for detecting vehicles and classifying their types with CG model images.

Our system is based on the algorithm of our previous work mentioned above. We have already shown that this system works very well, when a sufficient number of training images are available in advance. However, such a situation, although optimal, is not realistic. It is true that we can obtain real images with the camera in our system and that we can update our system every day, but before the update, we need to segment vehicle area, determine the orientation of each vehicle and classify its type. All tasks would be executed manually, which would be impossible. In short, collecting real images of the target vehicles is important for our system but it is generally a time consuming and difficult task. To reduce such tasks, we employ computer graphics (CG) as training images. Experimental results have confirmed that our system still works well using CG model images rather than real images.

It is true our system can be applied on any image from any camera angle, but in this paper we apply only the overhead view so that the system can serve as a verification system for ETC, where it is important to decide when communication should begin: if too soon, another vehicle may be traveling between the gate and the vehicle intended to be communicated with; if too late, the gate might not open in time. Our system can detect vehicle location correctly; therefore, it can enhance the ETC system. In order to avoid the occlusion problem, it is advisable to mount the camera on road,

Manuscript received February 28 2002

Manuscript revised May 25 2002

<sup>†</sup>The authors are with the Institute of Industrial Science, University of Tokyo.

facing down onto the road surface. This camera location is good for image processing but there still remain several problems such as casted shadows and changing illumination conditions, which is solved for limited number of vehicles with our system [10].

This paper is organized as follows. In the next two sections, we describe our algorithm. Our system classifies vehicles through the following two methods. The first one is the recognition with local features-based method described in section 2; the second one is the classification by using template matching described in section 3. After presenting the results of outdoor experiments in section 4, we close the paper with our conclusions.

## 2. Recognition Algorithm

We first recall our basic recognition algorithm, which is based on the eigen-window method originally developed by Ohba and Ikeuchi[5], [7]. Later, the algorithm was modified for the IMAV-vision board, which handles only integers using the vector quantization method [6]. The following flow shows our recognition method:

1. Make the database set in advance.
  - a. Make the set of training images.
  - b. Extract local feature points from each training image.
  - c. Compress the set of feature points.
  - d. Make a database set which consists of pairs of a compressed local feature and the location of the feature point in the training image.
2. Compare input images with the database set.
  - a. Extract local feature points from the input image.
  - b. Find the closest feature points in the database for each local feature point.
  - c. Make a vote such that two feature points in the input image are voted to the same point if, and only if, their relative position is the same in both the training image and the input image.
  - d. Detect according to the result of the vote.

The original algorithm is based on the eigen-space method [3], in which we calculate eigen values of a covariant matrix. The eigen-window method uses small windows as features for object recognition. This window method enables the algorithm to handle images that contain partially occluded objects.

## 2.1 Eigen-window Method

### 2.1.1 Eigenspace Technique

Here we summarize the eigen-space method. Let  $z_1, z_2, \dots, z_M$  be a set of training images of  $n \times m$ . Each  $z_i$  can be considered as a point in  $N = n \times m$  dimensional vector space. Let  $c$  be the average of all training images:  $c = \sum z_i / M$ . Then we have a  $N \times M$  matrix  $Z$  and its covariant matrix  $Q$  of  $N \times N$  as follows:

$$Z = \{z_1 - c, z_2 - c, \dots, z_M - c\}$$

$$Q = ZZ^t$$

Note that we can consider  $\tilde{Q} = Z^t Z$  instead of  $Q$  of  $M \times M$  if  $N \gg M$ .

Let  $\{\lambda_i\}$ ,  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_M \geq 0$ , be the eigenvalues of  $Q$ , and  $\{e_i\}$  be its eigen-vectors:  $\lambda_i e_i = \lambda_i Q$ . For a given threshold  $T$ , which is approximately 0.1, we can obtain such a number  $k$  that

$$\left( \sum_{i=1}^k \lambda_i \right) / \left( \sum_{i=1}^N \lambda_i \right) \geq T.$$

We can expect that  $k$  is sufficiently smaller than  $N$  if  $T$  is suitable. In general we can expect  $k$  is small enough, usually less than ten. The matrix  $Q$  is considered to be a diagonal matrix with respect to the base  $\{e_1, e_2, \dots, e_N\}$ . The contribution of  $e_{k+1}, \dots, e_N$  is relatively small; hence, we can reduce this linear map of  $Q$  as

$$E = [e_1, e_2, \dots, e_k].$$

Using this reduction, we identify any image  $p$  of  $n \times m$  as  $E^t(p-c)$ , which is a point in the  $k$ -dimensional vector space. If we put  $p = z_i$ , we obtain a reduced set of the training images.

In other words, we can reduce a set of points in  $N$ -dimensional vector space to a set of  $k$ -dimensional vector space. Usually  $k$  is about 10 or less, whereas  $N$  is over 1000; hence, it is much easier to match two images in  $k$ -dimensional vector space. We call the reduced  $k$ -dimensional vector space the eigen-space.

### 2.1.2 Local Feature Points

In this subsection, we explain how to select local features. We first extract the local features through the corner detector of Tomasi and Kanade[2]. Let  $I$  be the image intensity,  $\mathbf{x} = (x, y)$  be the coordinate and  $\mathcal{R}$  be the region of a window (e.g.,  $10 \times 10$  square). Consider the following matrix  $S$  of  $2 \times 2$ :

$$S = \sum_{\mathbf{x} \in \mathcal{R}} \begin{pmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{pmatrix} \begin{pmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{pmatrix}^T$$

Then  $S$  has two eigenvalues:  $\lambda_1, \lambda_2$ . We select the window as a local feature point if

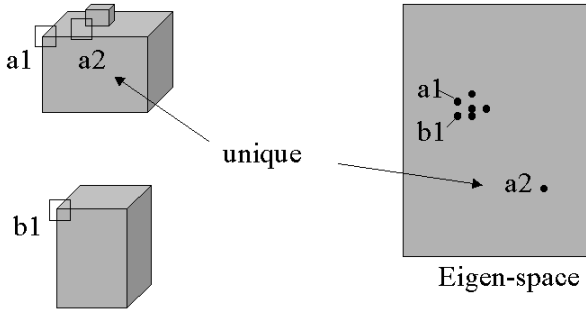


Fig. 1 Uniqueness criterion

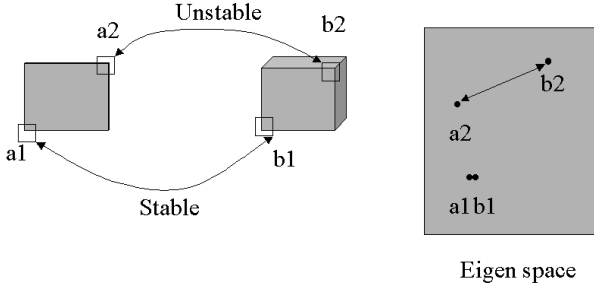


Fig. 2 Reliability criterion

$$\min(\lambda_1, \lambda_2) > \lambda$$

for a given threshold  $\lambda$ .

Secondly, when we make the database of training images, we reduce the local features to a lesser number of characteristic windows according to the criteria of uniqueness and reliability.

Let  $z_{i1}, z_{i2}, \dots, z_{iM_i}$  be the set of local features in a training image  $I_i$ , and  $z_{11}, z_{12}, \dots, z_{NM_N}$  be the set of all local features in the training images.

We define that a window  $z_{ij}$  is unique if it satisfies the following property:

**[Uniqueness criterion]**(Figure1) For all  $z_{kl} \neq z_{ij}$ ,  $\|z_{kl} - z_{ij}\| > \epsilon$ , where  $\epsilon$  is a given threshold.

If  $z_{ij}$  is not unique, it means that there exists a local feature  $z_{kl}$  which is similar to  $z_{ij}$ . This fact increases the cost of the vote because a window  $w$  which is similar to  $z_{ij}$  is usually similar to  $z_{kl}$  as well. In Fig. 1, the windows of  $a1$  and  $b1$  are similar, whereas the window of  $a2$  is unique. We remove the local features which are not unique, such as  $a1$  and  $b1$ .

A window  $z_{ij}$  is reliable if the following property is held:

**[Reliability criterion]**(Figure 2) Let  $I'$  be a image in which the target object in  $I$  is slightly moved, and  $z'_{ij}$  be the corresponding window of  $z_{ij}$  in  $I'$ . Then  $z'_{ij}$  is similar to  $z_{ij}$ :  $\|z'_{ij} - z_{ij}\| > \epsilon'$ .

Since a window which is similar to a non-reliable feature might not actually be similar to a part of the object recognized, we also remove the local features which are not reliable, such as  $a2, b2$  in Fig. 2.

We remove local features so that remaining local features satisfy both the uniqueness criterion and the

reliability criterion. We refer to the remaining local features as reduced local features.

### 2.1.3 Compression

We can apply the eigen-space technique to the set of reduced local features to obtain the set of compressed characteristic local features. The dimension of local-features is about  $10 \times 10$  and  $k$  is approximately five. We define the database as the set of all reduced local-features projected into the eigen-space, which is spanned by  $k$  local features.

In one case, the contribution of the first five eigen-values was 99.2%, which means that any local-feature is well-described as a linear combination of these five eigen-windows. We show examples of local-features and their eigen vectors in figure 4.

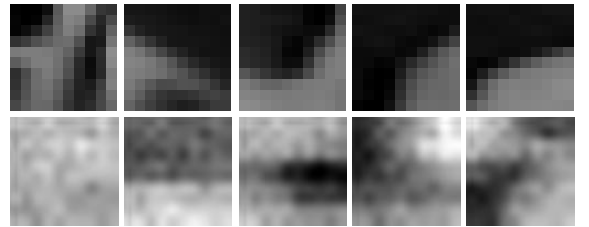


Fig. 4 Examples of local-features (upper line), The first five eigen-windows(lower line)

### 2.1.4 Recognition

For any input image  $J$ , we can extract local feature points  $\{w_l\}$  with the corner detector. Our system searches for locations where objects exist in  $J$  according to the following voting system(See Fig. 3).

Let  $D = \{z_{11}, z_{12}, \dots, z_{NM'_N}\}$  be the database. For simplicity, we rewrite  $D = \{z_1, z_2, \dots, z_M\}$  and assume that each  $z_i$  comes from the window located at  $(x_i, y_i)$  in the training image  $I_i$ .

The base space of our voting operation is  $\mathbf{Z} \times \mathbf{R} \times \mathbf{R}$ , where  $\mathbf{Z}$  corresponds to the number of training images, and the other two  $\mathbf{R}$ 's correspond to off-sets in the image.

Consider a local feature  $w \in \{w_l\}$ . If the location of  $w$  in the input image  $J$  is  $(x, y)$  and  $w$  is similar to some  $z_i$ ;  $\|z_i - w\| < \epsilon$ , then we put a vote onto  $(I_i, x - x_i, y - y_i)$ . If  $w$  is similar to another  $z_j$ , we also put another vote onto  $(I_j, x - x_j, y - y_j)$ .

It is easy to see that  $w_1$  and  $w_2$  are voted onto the same point if, and only if, there exist  $z_i$  and  $z_j$  such that

- (1)  $w_1$  is similar to  $z_i$ , and  $w_2$  to  $z_j$ .
- (2)  $I_i = I_j$ .
- (3)  $x_i - x_j = x_1 - x_2$  and  $y_i - y_j = y_1 - y_2$ .

If the number of votes on a point  $(I, x, y)$  is  $r$ , it means that there are  $r$  local features in a training image

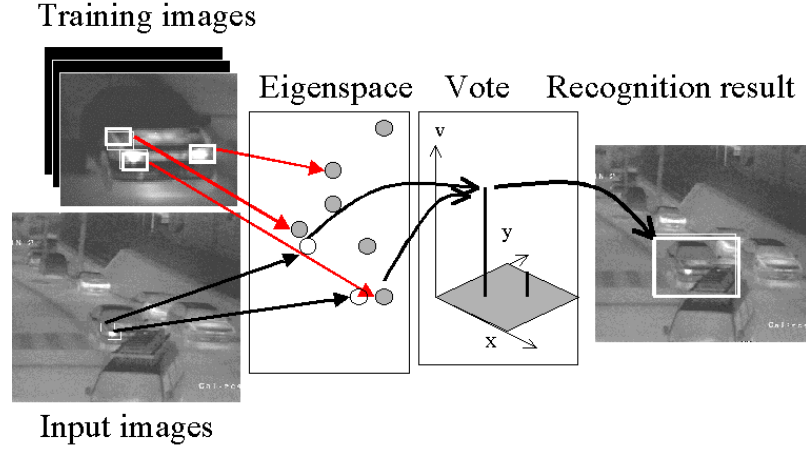


Fig. 3 Eigen-window method

$I$  such that their relative position in the training image is the same as that in the input image. Hence, for each point in the base space  $(I, x, y)$  on which the number of vote is large enough, our system tells that there is an object in the image  $I$  with the off-set  $(x, y)$ . Note that the off-set  $(0,0)$  means that the location in the input image is the same as that in the training image. Because of this voting operation, our algorithm has the following properties:

1. It might recognize occluded objects when there is a large enough number of local-features which are not occluded in input images.
2. It can recognize all non-occluded objects in input images.
3. It can detect the objects even if the location in an input image is different from that of the training image.
4. It does not require us to segment the vehicle area from the input images.

## 2.2 Vector-Quantization Algorithm

The vehicle recognition system using this algorithm has been reported in [9]. Although we have confirmed that the eigen-window method works well in detecting specified vehicles, its processing time is too long for practical use. Thus, we decided to implement our method on the image processing board, the IMAP-vision board. Since the hardware supports only integers, we modified our algorithm so that it is based on vector-quantization[6] instead of on the eigen-windows.

### 2.2.1 Local Feature Points

First, the binary image is obtained through an edge detector from the original image, and then the stable windows are selected as described below.

Let  $B(I; x, y; b)$  be the window of size  $(2b + 1) \times (2b + 1)$  pixels around  $(x, y)$  in a binary image  $I$ . The Hamming distance  $D_H$  between two binary vectors of the same dimension is defined as the sum of absolute distance of each corresponding coordinate value. Thus, the Hamming distance of two binary images is equal to the number of unequal elements in corresponding positions.

We can define a stable window of size  $(2b + 1) \times (2b + 1)$  pixels around  $(x, y)$  in an image  $I$  as those of which value  $r(I; x, y)$  is small.

$$r(I; x, y) := \min_{\substack{-d \leq d_x \leq d \\ -d \leq d_y \leq d \\ (d_x, d_y) \neq (0,0)}} (D_H[B(I; x + d_x, y + d_y; b), B(I; x, y; b)])$$

For each training image  $M_i$ , we can select the first  $n$  windows of size  $(2b + 1) \times (2b + 1)$  centered around  $(x_{ij}, y_{ij})_{1 \leq j \leq n}$ , such that  $r(M_i, x_{ij}, y_{ij})$  is small.

### 2.2.2 Compression Method

Let  $z_1, z_2, \dots, z_M$  be a set of binary training images of  $n \times m$ . They are considered to be points in  $N = n \times m$  dimensional vector space  $V$  on  $\{0, 1\} = \mathbf{Z}/2\mathbf{Z}$ . Then, for given  $G$ , we can obtain such  $G$  segments  $\{g_i\}_{i=1}^G$  in  $V$  that any  $g_i$  includes  $c$  training images, where  $c$  is  $N$  divided by  $G$ . Instead of the eigen-vectors used in the previous method, we can use  $\{v_i\}$ , the center of each segment:

$$v_i = \sum_{z_k \in g_i} z_k / c$$

In this way, we obtain only  $G$  kinds of images instead of  $N$  local features, and this  $G$  kinds of images are called the code features.

Note that the distance between two points in  $V$  is equal to the exclusive OR operation of two points.

### 3. Classification Algorithm

Our system recognizes vehicles in input images by using the algorithm described in section 2. We have already proposed a simple classification algorithm [8], but this algorithm is not sufficiently accurate to distinguish between two similar classes as shown in 4.1 below. For that reason, we propose our new vehicle-classification algorithm using template matching after recognition.

The following flow shows our classification method:

1. Make templates of target vehicles in advance.
  - a. Make binary edge images of target vehicles using the edge detector.
  - b. Make templates, extending binary edge images by quantized Gaussian function.
2. Select class candidates from recognition results.
  - a. Label class data on each training image.
  - b. Select class candidates by evaluating the maximum number of votes in recognition results.
3. Select class of the recognized vehicle by using template matching.

In the remainder of this section, we describe the vehicle classification method.

#### 3.1 Selection of Class Candidates

First, in the recognition phase (mentioned in Sect. 2), we label a particular class of each training model image; thus, the class label of the training image which shows maximum vote is the most likely class of the recognized vehicle in the input image. But features extracted in the recognition phase are suitable only for recognition, not for classification. Hence, the class label shown in the results of recognition, is not reliable for classification. For this reason, we select  $N_c$  class candidates in the recognition phase. Each training model is evaluated by the maximum number of votes, and then we can select first  $N_c$  class candidates such that maximum number of votes is large.

#### 3.2 Template Matching

##### 3.2.1 Making Templates

From selected candidates, we decide the class of the recognized vehicle by using the template matching method. First we prepare templates of the vehicles. Since our recognition method uses binarized edge images, we also perform template matching on binarized images. For binarized input images, we prepare edge templates extended by Gaussian function.

$m_3$	$m_3$	$m_3$	$m_3$	$m_3$	$m_3$	$m_3$
$m_3$	$m_2$	$m_2$	$m_2$	$m_2$	$m_2$	$m_3$
$m_3$	$m_2$	$m_1$	$m_1$	$m_1$	$m_2$	$m_3$
$m_3$	$m_2$	$m_1$	$m_0 = Max$	$m_1$	$m_2$	$m_3$
$m_3$	$m_2$	$m_1$	$m_1$	$m_1$	$m_2$	$m_3$
$m_3$	$m_2$	$m_2$	$m_2$	$m_2$	$m_2$	$m_3$
$m_3$	$m_3$	$m_3$	$m_3$	$m_3$	$m_3$	$m_3$

Fig. 5 The mask used to extend edge images

The templates are made from the edge images; first, the binary images are obtained from the training image through an edge detector, and then the binary edge images are extended by Gaussian function. Since we implemented our vehicle recognition system on IMAP-vision board, which supports only integers, we modified the Gaussian function so that it is suitable for the IMAP board. We use the mask shown in Fig. 5 as a quantized gaussian function in order to extend edge images, in this mask  $Max = m_0 > m_1 > m_2 > m_3 > 0$ , and  $Max \geq 8m_1 + 16m_2 + 24m_3$ . We calculate the value of templates to put on the mask at each point of the binary images.

Let  $e(x, y)$  be the value of the binary image at the point  $(x, y)$  ( $e(x, y)$  takes only 0 or 1), and  $t_i(x, y)$  be the value of the template  $t_i$  at same position defined as follows:

$$t_i(x, y) = \min(Max, \sum_{d_x, d_y=-3}^3 m_{||d||} e(x + d_x, y + d_y))$$

where we denote  $||d|| = ||(d_x, d_y)||_\infty = \max(|d_x|, |d_y|)$ .

Note that  $t_i(x, y) = Max$  if  $e(x, y) = 1$ .

In our experiment, we set that  $m_1 = 13, m_2 = 6, m_3 = 2, Max = 255$ , and we normalize the value  $[0, 255]$  into  $[0, 10]$  in order to be fit for the IMAP-vision board.

##### 3.2.2 Template Matching

In the recognition phase, our system can detect the position of a vehicle; thus, template matching might be done only around the detected position.

Let  $\{t_1, t_2, \dots, t_M\}$  be the templates of the candidate class  $C$ ,  $t_i(x, y)$  be the value of the template  $t_i$  at the position  $(x, y)$  in  $t_i$ , and  $I(x, y)$  be the binary value of the input image at position  $(x, y)$  in the input image. We put the template  $t_i$  at the position  $(x_i, y_i)$  in the input image; thus, the evaluation value  $Ev(t_i; x, y)$  of the template  $t_i$  at the position  $(x, y)$  in the input image is defined as

$$Ev(t_i; x, y) = \frac{\sum_{x=x_i, y=y_i}^{X+x_i, Y+y_i} t_i(x - x_i, y - y_i) I(x, y)}{\sum_{x=x_i, y=y_i}^{X+x_i, Y+y_i} I(x, y)}$$

where  $X \times Y$  is the size of the template. We obtain the

evaluation value  $Ev_C$  of the class  $C$  as follows;

$$Ev_C = \max_{\substack{-d \leq d_x \leq d, -d \leq d_y \leq d \\ 1 \leq i \leq M}} Ev(t_i; x + d_x, y + d_y)$$

We calculate  $Ev_C$  of all candidate classes, and select the candidate which has the highest value of  $Ev_C$  as the class of the recognized vehicle. In our experiment, we set that  $d = 5, M = 6$ .

#### 4. Experiments

In this section, we show the result of our outdoor experiment.

##### 4.1 Comparison Between Models From CG Images and Real Images

One purpose of this paper is to confirm that our algorithm is effective for systems which require accurate vehicle location; hence, we installed the overhead type of camera above the road. We had already developed a simple classification algorithm [8], thus, at first we applied it to the images taken by our camera. Making a class template of a sedan from CG model images, our system detected 25 sedans in 28 real images in which there was a sedan-type vehicle. As for the class of wagon, it detected 19 in 20 images. We will explain how to make templates from CG models in the following subsection.

We have already had results for real images. As described in [8], where the camera is located at the roadside, monitoring traffic flow, our system can detect vehicles of class sedan. In that case, we have achieved a recognition ratio of 91% for 280 images.

These results show that employing CG model images instead of real images is effective.

Thanks to CG model images, we can easily obtain many training images for various kinds of vehicles. As a result, we can perform experiments with several vehicle classes. But unfortunately, our simple algorithm does not work very well if we have two or more similar classes: the system often confuses similar classes, a problem which does not occur if their is only one class. For that reason, we need to make the algorithm more sophisticated so that it can distinguish among similar classes, as described above.

##### 4.2 Training Images of Computer Graphics

As we have already mentioned, it is a very time-consuming, but nevertheless important task, to collect real images of all target objects. We here propose to use CG images as training images to reduce the performance time of tasks without loss of accuracy. Recent computer graphics technologies provide CG images enough reality to enable their use as training model images for recognizing vehicles in real images. Model CG

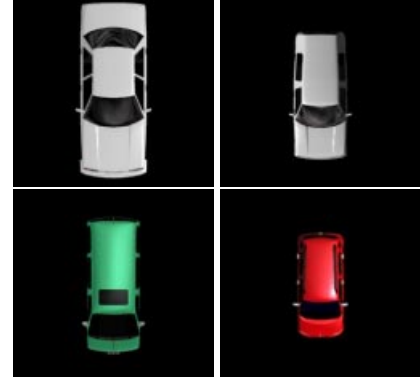


Fig. 6 CG training images

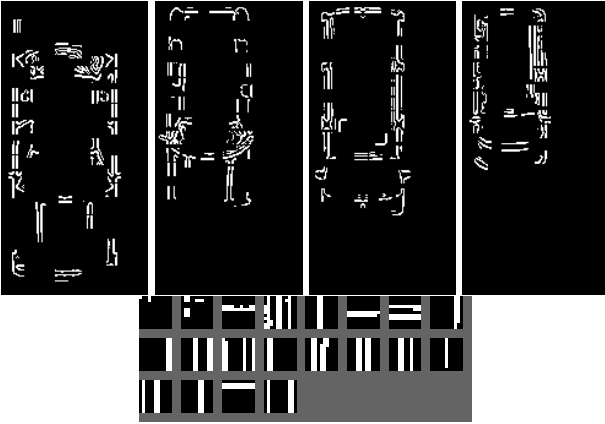
images can be created easily by using a 3-dimension computer graphics tool. We have confirmed that our system using CG images as training images is sufficiently effective for recognition of vehicles in real images, performing outdoor experiments. Sample images of CG training images are shown in Fig. 6.

##### 4.3 Outdoor Experiments

We have performed outdoor experiments to confirm the effectiveness of our vehicle classification system. Our system is based on the vector quantization algorithm mentioned above. In these experiments, our purpose was to detect vehicles in input images and to classify them into particular classes. We took several image sequences using a fixed camera, and we categorized vehicles in these images into 4 classes: sedan, wagon, mini van and hatchback. For each class of vehicles, we prepared training CG images and made models for recognition and templates for classification in advance; then we experimented to determine whether our system can recognize vehicles' locations. and classify to which class they belong. In the recognition phase, we selected 2 candidates for classification ( $N_c = 2$ ).

Table 1 shows the parameters of our system in experiments. We prepared 6 CG training images per each class to cover variation of appearances of vehicles in one class. We set the window size to  $11 \times 11$  and selected 26-42 local features on each image in proportion to their size to obtain 20 code features ( $2b + 1 = 11, G = 20$ ). Figure 7 shows extracted features from images shown in Fig. 6 and calculated codes.

To make the recognition system more robust, the vote space was filtered using  $15 \times 15$  mask whose element values are all 1 except the center, whose element value was 2. In addition, we used background subtraction in order to reduce excessive edges. Since we used a fixed camera, the background image did not change frequently. We prepared the initial background image and it was updated according to input images. The subtraction was performed after binarization. Because illumination often changes in outdoor environments, a



**Fig. 7** Extracted features from images shown in Fig. 6 and calculated codes

no. of classes	4	no. of models	6 (par class)
candidates	2	no. of features	26-42 (par model)
no. of codes	20	window size	11 × 11

**Table 1** parameters of our system in outdoor experiments

background subtraction between gray-scale images does not work well in such cases.

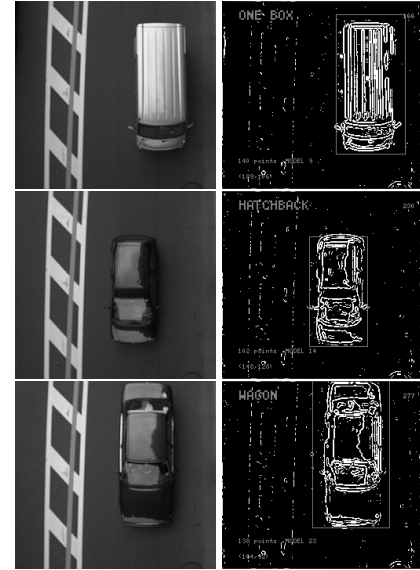
Let  $I_i(x, y)$  be the input image after binarization,  $I_b(x, y)$  be the binarized background image and  $I_s(x, y)$  be the image after subtraction, then we define a binarized background subtraction as follows;

$$I_s(x, y) = \begin{cases} 1 & (I_i(x, y) = 1, I_b(x, y) = 0) \\ 0 & (\text{other cases}) \end{cases} \quad (1)$$

#### 4.4 Results

We processed 87 input images; 28 of them included sedan type vehicles, 20 included wagon type vehicles, 20 included mini van type ones, and other 17 included hatchback type vehicles. The dimension of each image is  $256 \times 240$  pixels. Figure 8 shows several examples of input images and processed images. In processed images, recognized vehicles are shown as rectangles at the detected position, and classification results are displayed at the upper left of the images. Table 2 shows the confusion matrix of experiments; in this table, columns represent the true class of the vehicle in the input image decided by a human, and lines mean the classification results of our system. Our system can detect all vehicles at their exact positions and can classify them at 54 % accuracy. Compared with other methods [4], [11], our system shows good results.

The reason why our system works better than others is essentially the difference of the algorithms. Our algorithm is view-based and has a huge database of viewing for each camera. If we change the camera, we need to make a new database for it. But once the database is created, our system works well. In the case of [4], the algorithm is model-based and the recognition



**Fig. 8** Examples of input images and processed images

Class as	True Class			
	sedan	wagon	mini van	hatchback
sedan	9	2	4	1
wagon	15	14	2	3
mini van	0	1	11	0
hatchback	4	3	5	13
total (true)	28	20	22	17
accuracy	54%			

**Table 2** Confusion matrix of experiments

algorithm strongly depends on the model. In the case of [11], the inductance pattern is not sufficiently stable for vehicle identification. We think that any algorithm could be better if we had more real data, but, it is usually very hard to obtain such data and to analyze it. In our system, we can come around to collect a large amount of real training images by using CG images instead.

Most failures in this experiment were caused by the confusion between sedan vehicles and wagon vehicles. Actually, if we unite two classes of sedan and wagon into one class, the total recognition ratio becomes 74%. The reason why our system confuses both classes is as follows. As in figure 6, we can obtain a wagon vehicle occluding trunk of sedan vehicle. Thus, it is natural that our system would often confuse classes of sedan and wagon. For a practical system, we are developing a new algorithm which can distinguish two classes such that the template of one class is included in that of another one.

## 5. Conclusion

We have confirmed through outdoor experimentation that our local feature-based method using CG training images is effective for vehicle classification. In our experiment, CG models of vehicles categorized into 4

classes are given in advance and our system detects the location of the vehicles in input images. One of the common problems in model-based recognition systems is the difficulty of collecting model images of target objects. We can easily create each vehicle model by using a 3-D CG rendering tool; this reduces the trouble of collecting real images of the target vehicles. In classification experiments, the accuracy of our system is almost 55%. This is a good result compared with results of other methods [4], [11], but for a practical system, higher accuracy is required.

We plan to develop a more accurate vehicle classification method, which has enough classification accuracy for practical use, and also plan to implement a classification system.

## References

- [1] N.Seed, A.Houghton, "Background Updating for Real Time Image Processing at TV Rate," *Image Processing, Analysis, Measurement, and Quality*, vol.901,SPIE, 73, 1988.
- [2] 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 Univ., Jan. 1991.
- [3] H. Murase and S. K. Nayar, "Visual Learning and Recognition of 3D Objects from Appearance'," *International Journal of Computer Vision*, vol.14, no.1, pp.5-24,1995.
- [4] J. M. Ferryman, A. D. Worrall, G. D. Sullivan and K. D. Baker, "Visual Surveillance Using Deformable Models of Vehicles," *Proc. of International Symposium on Intelligent Robotic Systems*, Pisa, pp. 295-302, July 1995.
- [5] K. Ohba and K. Ikeuchi, "Recognition of the Multi Specularity Objects Using the Eigen Window," *Proc. of International Conference on Pattern Recognition*, Vienna, August 1996.
- [6] J. Krumm, "Object Detection with Vector Quantized Binary Features," *Proc. of Computer Vision and Pattern Recognition '97*, pp.179-185, 1997.
- [7] K. Ohba and K. Ikeuchi, "Detectability, Uniqueness, and Reliability of Eigen-Windows for Stable Verifications of Partially Occluded Objects," *IEEE Pattern Analysis and Machine Intelligence*, vol.19, no.9, pp.1043-1048, 1997.
- [8] M.Kagesawa, A.Nakamura, K.Ikeuchi, H.Saito, "Local-feature Based Vehicle Class Recognition in Infra-red Images Using IMAP Parallel Vision Board," *Proc. of 2000 International Conference on Intelligent Transportation Systems (CD-ROM)*, Oct. 2000.
- [9] M. Kagesawa, S. Ueno, K. Ikeuchi and H. Kashiwagi, "Recognizing Vehicles in Infrared Images Using IMAP Parallel Vision Board," *IEEE Trans. on Intelligent Transportation Systems*, vol.2, no.1, pp.10-17, March 2001.
- [10] T. Yoshida, M. Kagesawa, T. Tomonaka and K. Ikeuchi, "Vehicle Recognition with Local-Feature Based Algorithm Using Parallel Vision Board" *Proc. of 8th World Congress on Intelligent Transportation Systems*, Sydney, Sep. 2001.
- [11] S. M. Tabib, "Inductance-Pattern Recognition for Vehicle Re-Identification," *Proc. of 8th World Congress on Intelligent Transport Systems*, Sydney, Sep. 2001.

**Tatsuya Yoshida** He received the B.E. degree in information and communication engineering from the University of Tokyo, Tokyo, Japan, in 2000., M.S. degree in information and communication engineering from the University of Tokyo, Tokyo, Japan, in 2002. He now works for Matsushita Electric Industrial Co., Ltd., Osaka, Japan.

**Shirmila Mohottala** She was born in Colombo, Sri Lanka. She received her B. Eng. degree in Information and Communication Engineering from the University of Electro-Communications, Tokyo, Japan in 2001. She is currently a graduate student at the Institute of Industrial Science, the University of Tokyo. Her research interests include image processing, object recognition and image-based sensing in ITS.

**Masataka Kagesawa** He received the BS degree in mathematics in 1986 from Chiba University, Japan, the MS degree in mathematics from Tokyo Metropolitan University in 1988. He was a doctor course student at Tokyo Metropolitan University from 1988 to 1990. From 1990 to 1994, he was a technical associate at the Institute of Industrial Science, the university of Tokyo. He is now a research associate at the same institute. His research interests include traffic simulation with dynamic information, traffic management systems and sensing systems on Intelligent Road Traffic Systems.

**Katsushi Ikeuchi** He is a Professor at the Institute of Industrial Science, the University of Tokyo, Tokyo, Japan. He received the Ph.D. degree in Information Engineering from the University of Tokyo, Tokyo, Japan, in 1978. After working at the AI Lab., MIT for three years, the ETL for five years, and the School of Computer Science, CMU for ten years, he joined the university in 1996. He has served as the program/general chairman of several international conferences, including 1995 IEEE-IROS, 1996 IEEE-CVPR and 1999 IEEE-ITSC. He is on the editorial board of the *International Journal of Computer Vision*, and the *Journal of Computer Vision, Graphics*. He has been a fellow of IEEE since 1998. He is selected as a distinguished lecture of IEEE SP society for the period of 2000-2001 He has received several awards, including the David Marr Prize in computational vision, and IEEE R&A K-S Fu memorial best transaction paper award. In addition, in 1992, his paper, "Numerical Shape from Shading and Occluding Boundaries," was selected as one of the most influential papers to have appeared in *Artificial Intelligence Journal* within the past ten years. He is a fellow of the IEEE.