Recognizing Vehicles in Infra-red Images Using IMAP Parallel Vision Board

KAGESAWA Masataka, UENO Shinichi, IKEUCHI Katsushi, KASHIWAGI, Hiroshi

Abstract— **This paper describes a method for vehicle recognition, in particular, for recognizing a vehicle's make and model. Our system is designed to take into account the fact that vehicles of the same make and model number come in different colors; to deal with this problem, our system employs infra-red images, thereby eliminating color differences. Another reason for the use of infra-red images is that it enables us to use the same algorithm both day and night. This ability is particularly important because the algorithm must be able to locate many feature points, especially at night. Our algorithm is based on configuration of local features. For the algorithm, our system first makes a compressed database of local features of a target vehicle from training images given in advance; the system then matches a set of local features in the input image with those in training images for recognition. This method has the following three advantages: (1) It can detect even if part of the target vehicle is occluded. (2) It can detect even if the target vehicle is translated due to running out of the lanes. (3) It does not require us to segment a vehicle part from input images.**

We have two implementations of the algorithm. One is referred to as the eigen-window method, while the other is called the vector-quantaization method. The former method is good at recognition, but is not very fast. The latter method is not very good at recognition but it is suitable for an IMAP parallel image-processing board; hence, it can be fast.

In both implementations, the above-mentioned advantages have been confirmed by performing outdoor experiments.

Keywords— **Vehicle Recognition, IR image, Parallel Image Processor**

I. Introduction

Traditionally, 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, are often used. Those point-oriented sensors collect only binary information such as whether a vehicle exists at a certain point at a certain place; the sensors, however, cannot collect other important traffic information, such as space velocity, or size of vehicles. It is also true that those sensors do not work well at merging points because many vehicles switch lanes, and the sensors cannot detect such vehicles.

Recently, image-processing sensors have become practically available in ITS applications. Not only can those sensors measure the number of vehicles but they can also measure velocity; they also have the potential to detect traffic accidents. For example, Momozawa and Nomura developed an accident recognition system[6].

This paper proposes an algorithm for recognizing target vehicles, that is, for detecting types of vehicles, e.g., type A of company B, by using image-processing sensors. It is true that there are other methods to detect vehicle types, e.g.,axle-counting equipment [8]. But such systems actually detect the weight, the size, and/or the wheel-base of vehicles, not the vehicle model. Our method is vision-oriented, and can recognize specified vehicles on roads.

One of the common problems with image-based vehicle recognition systems is segmenting vehicle area from input images. Various research on updating background (e.g., [2]) has been done; hence, with the method of background subtraction, 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. Because the algorithm in our method uses local features, segmentation is not necessary in our system.

In principal, if we had detailed training images of all types of existing vehicles, and if we had a large amount of resources such as memory and time, our system would be capable of detecting the types of any vehicles on any road in the world.

Although some other automatic vehicle identification (AVI) systems, including two-way communication and number plate reading systems([3]) exist, they have several practical problems. For example, two-way communicaltion systems are expensive and not yet common; and data from number-plate reading systems should be treated very carefully due to the privacy issue. Thus, we prefer the vision-based system.

II. Recognition Algorithm

A. Overview

Our basic algorithm is based on the eigen-window method, originally developed by Ohba and Ikeuchi[12], [14]. Later, the algorithm is modified for the IMAP-vision board, which handles only integers using the vector quantization method proposed by Krumm[13]. The following procedure describes our 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 [9], in which we calculate eigen values of a covariant matrix. The eigen-window method uses small windows as features for object recognition. Due to this window method, the algorithm can handle images that contain partially occluded objects.

In the remainder of this section, we describe the eigen-window method([14]).

B. Eigen-window Method

B.1 Eigenspace Technique

We summarize the eigen-space method here. Let $z_1, z_2, \ldots, z_M$ 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:

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

$$Q = ZZ^t$$

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

Let $\{\lambda_i\}, \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M \geq 0$, be the eigen-values of Q, and $\{\mathbf{e}_i\}$ be its eigen-vectors: $\lambda_i \mathbf{e}_i = \lambda_i Q$. For a given threshold T, which is approximately 0.1, we can obtain such 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. The matrix Q is considered as a diagonal matrix with respect to the base $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N\}$. The contribution of $\mathbf{e}_{k+1}, \ldots, \mathbf{e}_N$ is relatively small; hence, we can reduce this liner map of Q as

$E = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k]$.

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 into 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.

B.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[4]. Let I be the image intensity, $\mathbf{x} = (x, y)$ be the coordinate and $\mathcal{R}$ be the region of a window (e.g., 10×10 square). Consider the following matrix S of 2×2:

$$S = \sum_{\mathbf{x} \in \mathcal{R}} \left(\frac{\partial I}{\partial \mathbf{x}}\right) \left(\frac{\partial I}{\partial \mathbf{x}}\right)^T$$

Then S has two eigenvalues: λ_1, λ_2. We select the window as a local feature point if

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

for a given threshold λ.

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

Let $z_{i1}, z_{i2}, \ldots, z_{iM_i}$ be the set of local features in a training image I_i, and $z_{11}, z_{i2}, \ldots, z_{NM_N}$ 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 ϵ is a given threthold.

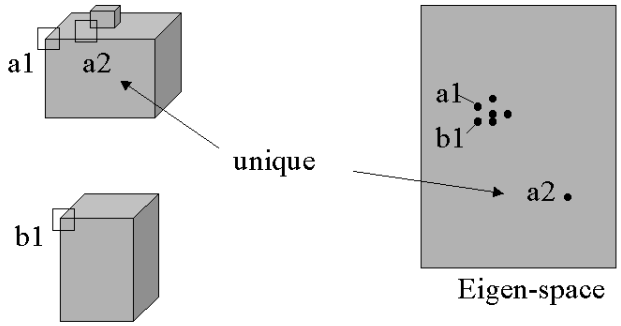


Fig. 1. Uniqueness criterion

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 vote because a window w which is similar to z_{ij} is usually similar to z_{kl} as well. In Figure 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:

[Reliablity 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}: $\|z'_{ij} - z_{ij}\| > \epsilon'$.

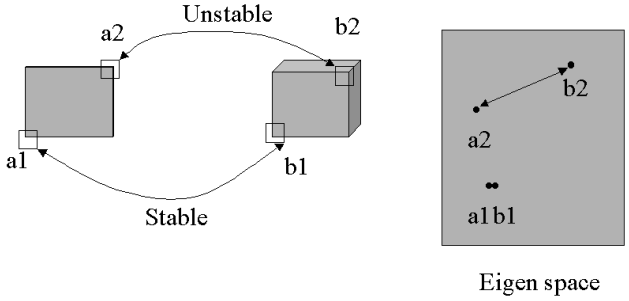


Fig. 2. Reliablity criterion

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 Figure 2.

B.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 10×10 and k is approximately five.

We call the set of local features the database.

B.4 Recognition

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

Let $\{z_1, z_2, \ldots, z_M\}$ be the database. For each z_i, we write that it comes from the training image I_i and its location in I_i is (x_i, y_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 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 system has the following properties:

1. It might recognize occluded objects when there is a large enough number of their local-features which are not occluded in input images.
2. It can recognize all 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.

III. Basic Experiment

Under two different sets of conditions, we conducted vehicle detection experiments with the eigen-window method, and confirmed that the method works well even if part of the specified vehicle is occluded.

A. Indoor Experiment

First we experimented in detecting model cars using indoor images. We prepared 48 training images, 24 for each model, rotating each model by increments of 15 degrees. Figure 6 shows some of the training images, Figure 7 examples of input images.

The dimension of each image is 512×480, and the number of local feature points in an image is approximately 100–200.

The dimension of each local feature is 10×10, and some of them are shown in Figure 4. The eigen-windows of the local features are considered as their principal components. The first five components are shown in Figure 5. In this case, the contribution of the first five eigen-values is 99.2%, which means that any local feature is very well-described as a linear combination of these five eigen-windows.

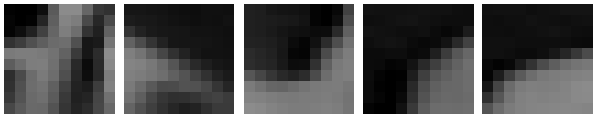


Fig. 4. Examples of local feature windows

Fig. 5. The first five eigen-windows

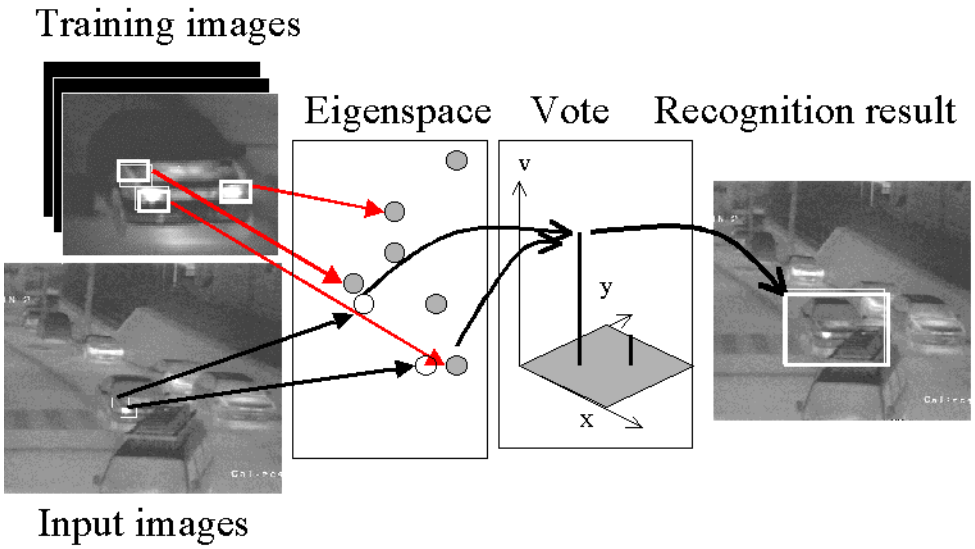
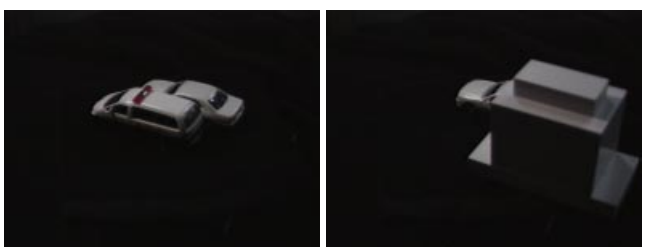


Fig. 3. Eigen-window method

Fig. 6. Examples of training images

The table I shows some of the voting results. In the table, S–15 denotes that the type of model is a sedan and the rotated degree is 15. In the same way, the training image of a model wagon rotated by 30 degrees is described as W–30. The numbers in the table show the number of votes for the training image on the left-hand side. We do not show incidences of less than 6 votes, and the symbol $^{\times}$ means wrong detection. In the case of #7, our system might not have determined the pose of the wagon but this is to be expected because the actual pose was rotated by about 280 degrees. In other cases, our system was somewhat confused with the pose rotated by 180 degrees. These results show that we do not need many poses for a vehicle, and that, with the exception of a rotation of 180 degrees, our system can recognize the vehicles very well.



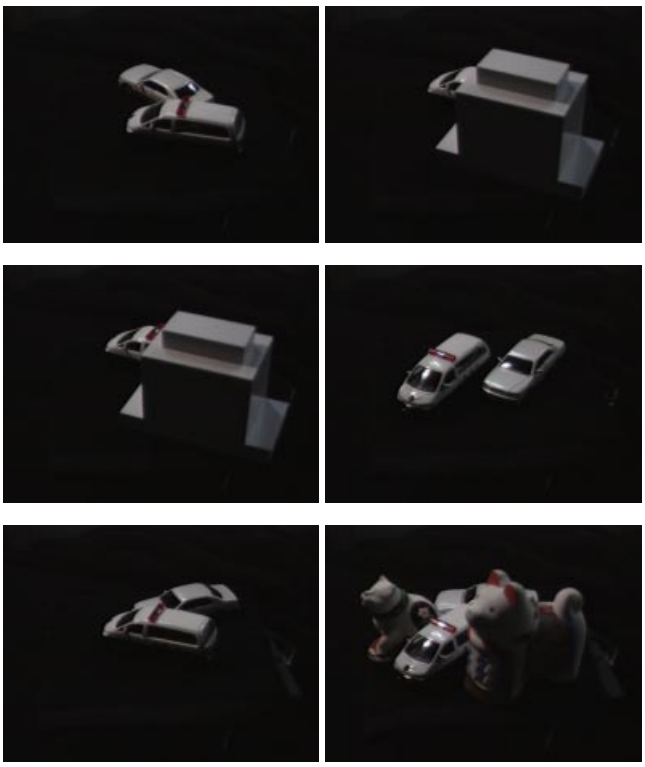


Fig. 7. Examples of input images(#1 – #8)

We processed 20 images and there were no false alarms. Except for the figure in the right-bottom corner of Figure 7, in which most of the vehicles were occluded, our system could recognize the vehicles and was able to detect their pose and location correctly.

This experiment confirmed that local feature-based detection is an effective method for detecting vehicles.

Input images Training image	#1	#2	#3	#4
S–255			32	
S–270	56	100		
W–270	77			65
W–285			32	
Input Images Training Images	**#5**	**#6**	**#7**	**#8**
S–000		135		
S–165				$7^{\times}$
S–300				7
S–315			10	
S–330				$7^{\times}$
W–270	87		$13^{\times}$	
W–285			15	
W–345		8		7

TABLE I
VOTING EXAMPLES OF INDOOR EXPERIMENT

B. Outdoor Experiment

Our second experiment was conducted to confirm the effectiveness of our method in outdoor environments. Our eigen-window method requires many local feature points. Unfortunately, however, we could not expect a large enough number of local feature points at night,if we employed only the usual optical images. In order to increase those features, particularly in night time images, we decided to use infra-red (IR) images. Another reason why we employed IR images is to solve the color problem, i.e., vehicles of the same make and model come in different colors, but we would like to recognize them as one model.

Our IR camera can detect waves of 3–5 μm length; through the outdoor experiment, we have determined that this camera can sense a temperature range that of about 5–40 degrees centigrade.

We taped several sequences, in daytime and at night on March 10, 1998. All the images in this paper are taken from the same videotape.

We prepared two kinds of training images, taken both in daytime and at night. These training images were manually segmented from the background.

Fig. 8. Training images(left:day, right:night)

We prepared 30 input images for detection, all of which were different from any of the training images. The figure 9 shows some of the input images. The results are shown in the table II. The symbol "–" means that the system can not detect the vehicle, even if the specified vehicle is in the input image. This kind of error was not observed in the indoor experiments. Our system had one false alarm, and failed to detect in 5 images; in four of these, the vehicle was mostly occluded. The other image in which our system failed to detect was actually a different image. It is true that the specified vehicle is running in the image, but it appears to be a different one because it has been just started and its engine is not yet warm.

These results confirm that our method works for partially occluded vehicles in outdoor environments.

Fig. 9. Examples of outdoor IR images(D–3,4, N–3,4)

Input Images Training Images	D–1	D–2	D–3	D–4
Day	–	–	98	
Night				
Input Images Training Images	**N–1**	**N–2**	**N–3**	**N–4**
Day				
Night	7	–	22	40

TABLE II
SOME VOTING RESULTS OF OUTDOOR EXPERIMENT

IV. PARALLEL IMPLEMENTATION ON THE IMAP-VISION BOARD

Although we have confirmed that the eigen-window method works very well in detecting specified vehicles, about 140 seconds on a Sun SPARCstation–20 are required to process one input image. This processing time is too long for practical use of our method. Thus, we decided to implement our method on an image processing board. We selected the IMAP-vision board because it has 256 processors with 1DC language development kit. Since the hardware supports only integers, we modified our algorithm so

that it is based on vector-quantaization[13] instead of on the eigen-windows. The dimension of the images is also reduced to 256×240 pixels, one-quarter of the previous system, in order to maximize the performance of the IMAP-vision board.

The hardware image processing board does not support floating-point calculation and in the eigen-windows method, we need to calculate eigen-values. Hence, we changed implementation of our algorithm from the eigen-window method to the vector quantization method ([13]) so that we need only integer calculation.

A. Vector Quantization Method

A.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.

$$\begin{aligned} r(I;x,y) &= \min\{D_H[B(I;x+d_x,y+d_y;b),B(I;x,y;b)]; \\ &\quad -d\le d_x\le d,-d\le d_y\le d,(d_x,d_y)\neq(0,0)\} \end{aligned}$$

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

A.2 Compression Method

Let $z_1,z_2,\ldots,z_M$ be a set of binary training images of $n\times m$. They are considered as 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 devided 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, which are called the code features.

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

B. New Implementation

We selected 25 local feature points whose stable windows did not overlap one another. Then we computed the centroid of the set of the $M(=25\times n_m)$ local feature points to obtain $20(=G)$ code features $\{v_i\}_{i=0}^{20}$, where n_m is the number of the models. In the manner of the previous subsection, we reduced each local feature point to a pair of the closest code features and the coordinate of the center of the window in its training image. We obtained the compressed model of $25\times n_m$ local features $(c_{k_{ij}},x_{ij},y_{ij})_{1\le i\le n,1\le j\le 25}$, where $c_{k_{ij}}$ is the closest code feature to $B(M_i;x_{ij},y_{ij};b)$ in $\{v_i\}$. The figure 10 shows the local feature points in training images, and the figure 11 shows our code features $\{v_i\}$.

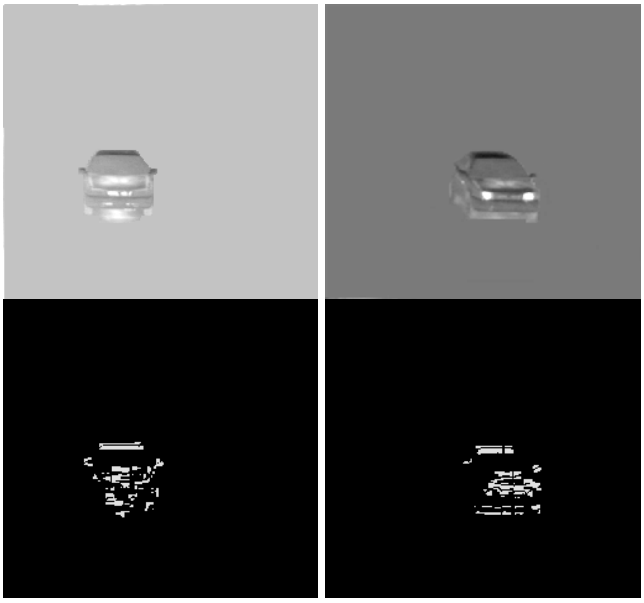


Fig. 10. Training images and feature points

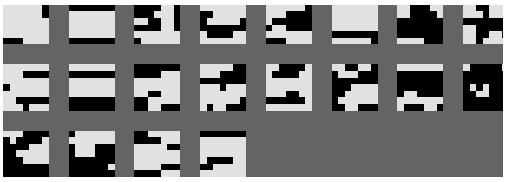


Fig. 11. Code features

C. Detection

We have already described our voting operation in the algorithm section. We here discuss the threshold of the voting operation because the maximum number of votes at a point is very limited in this method.

Our system tells that the specified vehicle exists at the location of offset (d_x,d_y) if the number of votes for $(n,d_x,d_y)\in\mathbf{Z}\times\mathbf{R}\times\mathbf{R}$ is greater than or equal to the given threshold V.

We can estimate the threshold V on probability calculation. The probability of false alarm $p_{fa}(V)$ with the threshold V is calculated as follows: Let (n_m,n_x,n_y) be the dimension of the voting base space. Then

$$p_{fa}(V) = 1 - (1 - p_f)^{n_m n_x n_y}$$

where p_f is the possibility that the number of votes is equal to or greater than V at a location in a training image. At first we assume that any local feature of background, which is not any part of the target vehicles, is uniformly distributed; this assumption makes calculation easier but approximate. Under this assumption, the possibility that a local feature can be equal to a code feature is $1/G$, where G is the number of the code features. Hence, p_f is calculated with binominal distribution:

$$p_f = \sum_{i=V}^{n_f} \left(\begin{array}{c} n_f \\ i \end{array} \right) \left(\frac{1}{G}\right)^i \left(1 - \frac{1}{G}\right)^{(n_f - i)}$$

where n_f is the number of local features in an image. We now have the explicit expression of p_{fa} as follows:

$$p_{fa}(V) = 1 - \left(1 - \left(\sum_{i=V}^{n_f} \left(\begin{array}{c} n_f \\ i \end{array} \right) \left(\frac{1}{G}\right)^i \left(1 - \frac{1}{G}\right)^{(n_f - i)}\right)\right)^{n_m n_x n_y}$$

Setting that $n_f = 25, G = 20, n_m = 2, n_x = 127, n_y = 120$, we obtain:

$$\begin{aligned} p_{fa}(8) &= .449, \\ p_{fa}(9) &= .057, \\ p_{fa}(10) &= .005. \end{aligned}$$

The function $p_{fa}(V)$ is monotonically decreasing; the threshold V should be nine or ten. We realized that actual probability is much smaller than a theoretical probability, probably because our assumption that local background features are uniformly distributed was not true. On the other hand, the smaller the threshold V, the better our system could recognize occluded vehicles. We therefore set the threshold to nine.

D. Implementation on IMAP-vision board

The IMAP-vision board is a parallel image processing board with 256 processors (PE). Each chip contains 1KB of memory, and 16MB external memory is shared with all processors on board. We assigned each processor the task of processing a vertical line. Because most of our method needs only local information, it was easy to implement the method on the board. But we lacked a good algorithm for our voting operation, the part which is still slow and dominant.

The eigen-window system on a Sun SPARCstation 20 needs about 140 seconds to process one image of the size 512×480; hence, it takes about 35 seconds for processing an image of the size 256×240. Our system can process one input image in less than 1 second.

E. Results

We have processed about 400 input images, all of which are different from any of the training images. Half of them are images in which the target vehicle appears to be almost the same size as the training image; the other half are not, that is to say, in these images, there might be several vehicles but no target vehicle, the vehicle which we would like to recognize, or there is a target vehicle whose scale is different from that of any training image. As we discussed, we fixed the threshold to nine votes throughout the experiment.

Our system can detect the target vehicle if its appearance size in the input image is almost the same as that in the training image. Figure 12 shows several examples of input images and processed images. Model 1 means the model in daytime, and Model 2 is the model at night. The table III shows the result of detection. The upper line shows the result of images in which no specified vehicle appears, but other vehicles do appear. The lower line shows the result of images in which the specified vehicle runs and its size in the image is almost the same as that in the training image. Detection failure of ten cases in the lower line was caused by occlusion; however, we have input 50 occluded images in total. In other words, our system has correctly detected 80% of partially occluded images.

We have again confirmed that our system can detect the specified vehicle even if it is partially occluded.

Input Im.\ Result	Failure	Success	Ratio
No specified veh.	0	200	100%
Exists specified veh.	18	182	91%

TABLE III
RECOGNITION RESULTS

For a practical system, we still have several issues that must be solved. One issue is the system's robustness to the vehicles' environments, e.g., we have to take into account such conditions as different seasons, different weather. Moreover, even for vehicles of the same make and model, we need to consider such things as different body colors and different temperature distributions according to running time.

We have captured many infra-red images of the same vehicle in different seasons to test our system's performance. For instance, we have made our model from training images in spring, and included summer and winter videos in our system. In this case, there were no false alarms, but the recognition ratio was decreased. The number of votes to any point was not large enough for our system to recognize the vehicle, although there were votes to the correct point. In other words, the differences of the various environments made input images noisy, which decreased the number of votes. On the other hand, in the second version of our system with an IMAP board, we could treat images of the dimension 256×240, and the dimension of vehicle area would be too small to obtain a large number of feature points. Our system would perform much better if we

had a large number of feature points.

We noted that, in the case of spring and summer, the camera positions were not exactly the same. Thus, we realized that our system is robust for rotation of the vehicle.

There is a possibility that our system might be confused by similar-looking makes and models; however, the fact that our system seldom has few false alarms indicates that our feature selection method is good. We expect that our system can recognize vehilce make and model as well as human beings do.

Another problem is that of scale. It is true that our system often fails to recognize the vehicle if its scale is different from any scale of the training images. In order to solve this problem, we can prepare multiple scale images for each vehicle. However, in the field of ITS, the roadside camera is fixed; hence, there is a limited area where the vehicles run in view of the camera. As a result, the number of multiple images is small.

Fig. 12. Vehicle detection on IMAP-vision board

V. Conclusion

We have confirmed that our local-feature based method is effective not only for vehicle detection but also for vehicle recognition. In paticular, our system has the advantage in cases where the target vehicles are partially occluded by other vehicles. At first we applied the eigen-window method to perform both indoor and outdoor experiments. We have also implemented the method based on the vector quantization on a parallel image processing board, the IMAP-vision board. Our system can recognize the target vehicle in less than 1 second. The detection speed is still not fast enough because vehicles move 15 meters in one second and our detecting area is small. The accuracy of our system is over 90% both in the eigen-window method and in the vector quantization method.

We believe that our algorithm is general enough to be applied to outdoor visible images on a CCD camera, if there are large number of feature points in the images. In paticular, visible images are required if we need to specify colors or painting. i.e., signs, on vehicles. The indoor experiment suggests that our algorithm can cooperate with outdoor visible images.

We plan to increase the speed of our system. We also plan to extend the system so that it can treat many training images while cooperating with the host computer. Finally, we are going to apply our method to recognize the classes of vehicles, such as trucks, buses, sedans, and so on.

References

[1] M.Yachida, M.Asada, S.Tsuji,"Automatic Analysis of Moving Images," IEEE Trans. Pattern Analy. & Mach. Intel., PAMI–1,12,1981.

[2] 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.

[3] A.Shio,"Character recognition in scene images," Proc. of CASA/SME AUTOFACT '89, pp.10-33 – 10-44, 1989.

[4] 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.

[5] T.Kanade and C.Tomasi,"Detection and tracking of point features," Technical Report CMU–CS–91–132, Carnegie Mellon Univ., Jan. 1991.

[6] M.Momozawa, M.Nomura, "Accident Vehicle Automatic Recognition System by Image Processing Technology," IEEE International Conference on Image Processing, 1992.

[7] Jianbo Shi, Carlo Tomasi,"Good Featurers to track," Proc. of Computeter Vision and Pettern Recognition (CVPR) '94,pp.593–600, 1994.

[8] T.Ueda, K.Takemura, S.Ogata, T.Yamashita,"Optical Axle-counting Equipment,"ITSC '97,1997.

[9] H.Murase and S.K. Nayar,"Visual Learning and Recognition of 3D Objects from Appearance'," Int'l J. Computer Vision, vol.14, no.1, pp.5-24,1995.

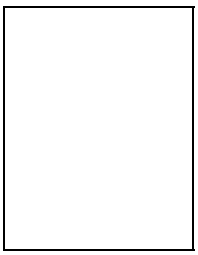
[10] S.K.Nayar and H.Murase,"Learning, Positioning, and tracking Visual Appearance," Proc. IEEE Int'l conf. Robotics and Automation, San Diego, Calif, May 1994.

[11] H.Murase and S.K.Nayar,"Image Spotting of 3D Objects Using parametric Eigenspace Representation," Proc. 9th Scandinavian Conf. Image Analysis, pp.325–332, June 1995.

[12] 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.

[13] John Krumm, "Object Detection with Vector Quantized Binary Features," Proc. of Computer Vision and Pattern Recognition '97, pp.179–185,1997.

[14] Kohtaro Ohba and Katsushi 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.
[15] M.Kagesawa, S.Ueno, K.Ikeuchi and H.Kashiwagi, "Recognizing Vehicle in Infra-red Images Using IMAP Parallel Vision Board,"Proc. of 1999 IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems (ITSC '99), pp.2–7, Oct.1999.
[16] M.Kagesawa, S.Ueno, K.Ikeuchi and H.Kashiwagi, "Vehicle Recognition in Infra-red Images Using Parallel Vision Board," Proc. of the 6th World Congress on Intelligent Transportation Systems, Nov. 1999.



KAGESAWA Masataka recieved 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.

UENO Shinichi will be filled later.

IKEUCHI Katsushi will be filled later.

KASHIWAGI Hiroshi will be filled later.