Journal of Science and Technology Volume 46, Number 5A, 2010 pp. 135-145

OBJECT DETECTION BASED ON EDGE, CORNER AND HoG FEATURE

BAO NGUYEN THIEN, TADASHI MATSUO

ABSTRACT

In this paper we propose a simple approach for detecting object. Our method is followed by the way an artist sketches out objects. He draws very fast with very few lines, but once we look at these lines we can easily recognize what the object is. This method is described detail in the part two. In this part we also survey techniques for detecting corner and edge in an still image. In the next part, we continuously present our experiment on applying this method for detecting some common objects such as car, bicycle, airplane, train. In part four, we present result and a comparative evaluation of different methods versus our method..

Key words: object detection, edge and corner feature, HOG, computer vision.n

1. INTRODUCTION

Object detecting and recognition is now still a difficult problem for computer vision, although different solution for this matter have been developed over the past few years. These approaches based on features of an image including both local and global ones for detecting object. But almost these approaches based on features which with human eyes it is difficult to see. Only computer can detect and recognize these features. So, in our method, we base on features that is easily seen by naked-eye and very close with natural detecting by human. When an artist draw an object, he outlines some basic lines in some directions. These lines meet each other at some specific points. This arrangement makes human easily recognize what the object is. From this viewpoint, we propose an simple approach for detecting object in a still image base on edge, corner and orientation of edge. We believe that our method works efficiently because it resembles with the way that human being detect objects, and our performance for some-object-detecting shows that it is true.

The paper is organized as follow: in Section 2, we introduce our propose method, which base on edge, corner and HoG feature. We also explain the important role of these feature for recognizing object with human being. In section 3, the more detail in technical about how to detect edge, corner and HoG are described. After that, the implementation of detecting five general object in PASCAL-VOC contest, include: side view car, front/rear view car, bicycle, train, and aero plane is display in this section. The result and evaluation of this method is in section 4. As in common, we discuss about the advantage, disadvantage and future work of our method in the final section.

2. DETECTING OBJECT BASED ON EDGE, CORNER AND ORIENTATION

It is easy to see that the most important features for recognition object are the overall shape of that object and all boundaries which separate main parts of objects. Shape or boundary is basically composed by *edges*. These edges are arranged in an specific order and meet each other at some points called *corners*.

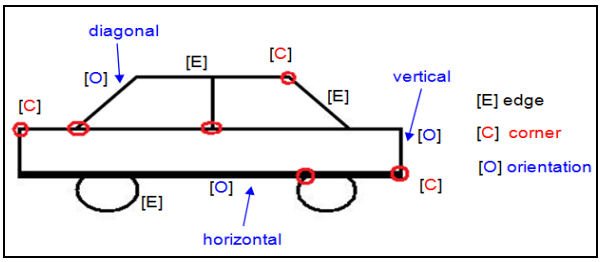

Figure 1. Three main components *(edge, corner and orientation)* make object can be figured.

Beside, edge orientation also play an prominent role for figuring object. With the same number of edges, a talent artist can organize in different order and different orientation to make viewer imagine different objects. Fig. 1 describes how these principal components make object imagination. Based on this, we propose an approach for object detecting which focuses on edge, corner and edge orientation. More detail is presented in the next diagram:

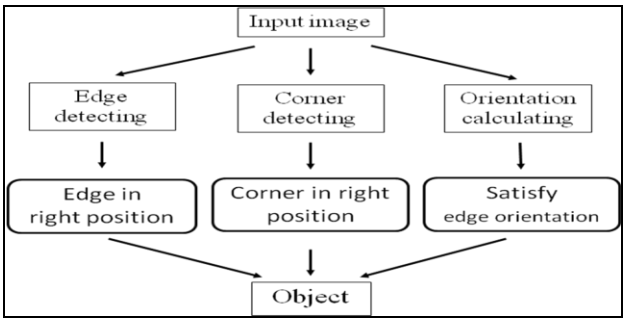

Figure 2. A model for object detecting based on edge, corner and edge orientation.

But there is still an issue in this model. That is how can we know that edge/corner is located at right position or not? In order to answer this question, we must point out the place where edge/corner must belong to, before we carry out detecting object. Actually, there are many way to specify location of edge/corner, such as using prior knowledge, or machine learning… In our implement, we choose machine learning approach because this method is more general for a lot of objects and the most important reason is that when using this method, user doesn't have to know more detail about object that they want to detect.

2.1. Edge and Corner Detection

Up to present, there are a lot of implementation of edge/corner detection and get the result with a high accuracy rate. Many of them have been reported in public. Zuniga and Haralick fit a

continuous surface over a small neighborhood of each point and consider the rate of change in gradient direction [7]. Kitchen and Rosenfeld proposed a cornerness measure based on the change of gradient direction along an edge contour multiplied by the local gradient magnitude [9]. Moravec defined ”points of interest” as points where there is a large intensity variation in every direction [8]. Harris and Stephens used image derivatives to estimate the autocorrelation of the image [1]. Rangarajan, Shah and Brackle found an optimal function representing the corner detector which when convolved with the gray level function yields a maximum at the corner point [12]. Rafajlowicz, E. in [10] uses the idea of vertically weighted regression and in its simplest form it leads to interpreting SUSAN in terms of a box sliding on the surface of an image. This modification of the SUSAN algorithm is still simple, robust against errors and provides thinner edges, without further efforts on additional thinning. Canny [11] introduces the notion of non-maximum suppression, which means that given the pre-smoothing filters, edge points are defined as points where the gradient magnitude assumes a local maximum in the gradient direction. And in 2007, Sonya Coleman *et al* [3] proposed a new method for enabling edge and corner detection to be integrated with a significantly reduced computation time.

In our performance, we have tried with many method for edge/corner detecting. And it shows that Canny detector gives the best result for edge detection, while Harris detector works well with corner detection.

2.2. Edge Orientation Detection

Specifying orientation of each edge is not an easy task. Because, with all edge detection method above, we only get the total edge image not each separate edge in image. So, it is almost impossible to know the orientation of each edge due to we don’t specify each edge individually. Instead of that, we can calculate the domain orientation of all edge in a specific sub-image. From this, we can define relatively the edge orientation. The domain orientation of a region can be calculated with HOG descriptor method (Histogram of Oriented Gradient). HOG descriptor is a local statistic of the orientations of the image gradients around a keypoint. HOG descriptor was initially proposed by Lowe [9] in his Scale Invariant Feature Transform (SIFT) [9]. Several HOG-based algorithms have been recently presented [4, 5] and combined with technologies such as boosted classifiers [2]. Navneet Dalal and Bill Triggs (INRIA) have used this method for human detection with a high accuracy rate [2]. David Monzo *et al* [6] compare HOG-EBGM vs. Gabor-EBGM and show the result that HOG has a better performance.

3. IMPLEMENTATION

We apply our approach for some detecting some objects including car (front/rear view, side view, bicycle, airplane). We choose these objects to detect because this task has been done by many researcher around the world, especially in PASCAL-VOC contest which is held in every year. So we can easily compare the result of our method with others to evaluate it exactly. As mention on part two, before detecting object we must know the position of each edge or corner. When applying this approach for detecting, we use machine learning method to mark locations

of edge or corner. Set of position of edge is called edge map, and ones of corner is corner map. Our object detecting system includes two main steps: making edge map and corner map by using machine learning with images from training database; and detecting object based on edge, corner and orientation.

3.1. Making Edge Map and Corner Map

To make edge map and corner map, we use same method as Zhenfeng Zhu *et al* in [11]. For all images in the training-image set, after detecting edge/corner we accumulate them in one temp image T_e or T_c. Then edge map and corner map will be made from these temp images respectively.

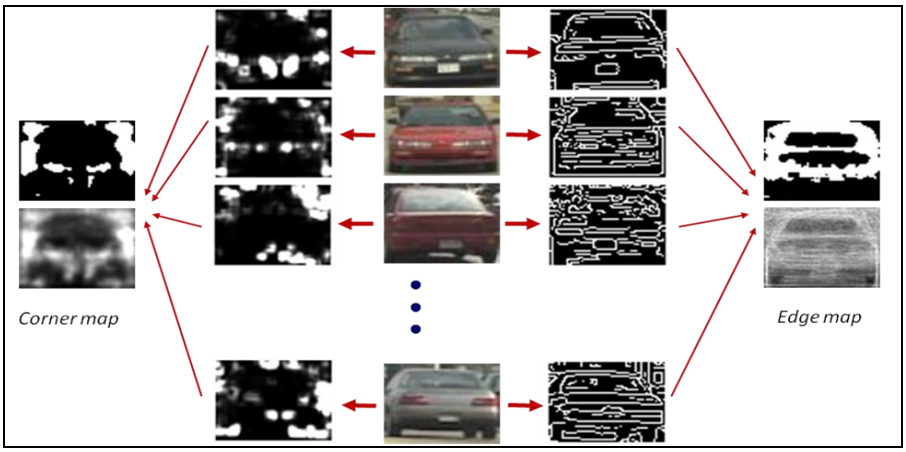

Figure 3. Edge map and corner map from training set.

However, [15] makes edge map and corner map be binary images. Our performance shows that with gray scale image, the result will be better. Because with a binary map, the location of edge or corner must be fixed at that position. But, due to variant of scale or view point, the location of edge or corner cannot be fixed at one specific point, other while it can be swung in a small region. It means that we should give a weight for every edge/corner pixel. That weight is the degree of how much the edge/corner pixel is in right position. So, gray scale map can work better in this case.

The edge map and corner map are constructed from T_e and T_c as follow:

$$M_E(x,y) = \frac{1}{\theta_1 * N} T_e(x,y), \qquad M_C(x,y) = \frac{1}{\theta_2 * N} T_c(x,y) \quad (1)$$

where N is the number of images in the training set, and θ_1, θ_2 are thresholds. Here, we choose $\theta_1$to equal 25 and θ_2 to equal 35. $M_E(x,y)$ and $M_C(x,y)$ denotes for edge map and corner map respectively.

3.2. Detection Object

With an input image, we will make edge image- I_E and corner image-I_C. To extract edges for making edge image, we use canny edge detector with 3x3 structure.

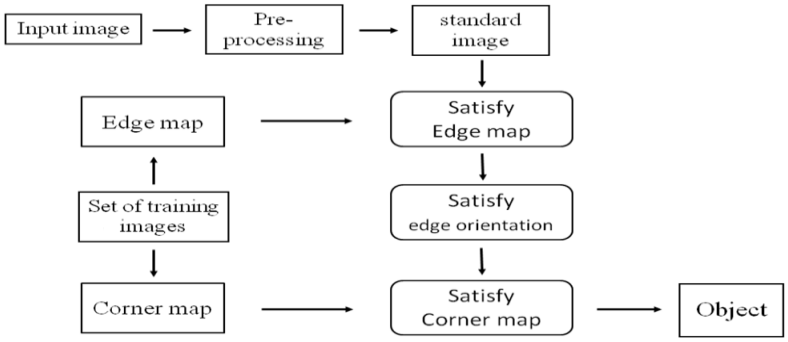

Figure 4. Two stages of object detection system based on edge, corner and orientation.

The position of corners is localized by using Harris corner detector with 5x5 structure. Beside, Gaussian filter is applied on input image before extracting edge and corner features to subtract background

Satisfying edge map. Edge image after detecting edge from input one is denoted as I_E. Let's define two parameters: n_1^e is number of edges in the input edge image I_E and n_2^e is number of edges matching between I_E and M_E. Degree of satisfying M_E of I_E is determined by some conditions as bellow:

$$C_1^E(I_E) = \begin{cases} 1 & n_2^e > N_1^e \\ 0 & else \end{cases}. \tag{2}$$

$$C_2^E(I_E) = \begin{cases} 1 & \frac{n_2^e}{n_1^e} > \theta^e \\ 0 & else \end{cases} \qquad C_3^E(I_E) = \begin{cases} 1 & n_1^e < N_2^e \\ 0 & else \end{cases}. \tag{3}$$

where N_1^e, N_2^e are constants and θ^e is a given threshold. $C_1^E(I_E)$ (2) guaranties that edges in the input image must be in correct position to construct object like edges in M_E. While both $C_2^E(I_E)$ and $C_3^E(I_E)$ (3) are to avoid situation that there are so many edges in the input image. So, if

$$Q_E = C_1^E(I_E) \cap C_2^E(I_E) \cap C_3^E(I_E) \tag{4}$$

is true, then I_E satisfies M_E, and the image will be passed to the next corner test step.

But, as we mention in part two, after detecting edge we cannot know exactly individual edge. It also means that we cannot calculate n_1^e and n_2^e. However, instead of counting number

of edges, we can sum of pixels which are marked as edge. Thus, n_1^e and n_2^e can be known approximately as

$$n_1^e \approx \frac{1}{255}\sum I_E(x,y) \qquad n_2^e \approx \frac{1}{255\times 255}\sum I_E(x,y)\cdot M_E(x,y)\,. \tag{5}$$

Here, because both I_E and M_E are gray scale ones, we should divide the sum to 255 for each one.

Satisfying corner map. Similarity to the previous step, in this we also define three conditions

$$C_1^C(I_C) = \begin{cases} 1 & n_2^c > N_1^c \\ 0 & else \end{cases}\,. \tag{6}$$

$$C_2^C(I_C) = \begin{cases} 1 & \frac{n_2^c}{n_1^c} > \theta^c \\ 0 & else \end{cases} \qquad C_3^C(I_C) = \begin{cases} 1 & n_1^c < N_2^c \\ 0 & else \end{cases}\,. \tag{7}$$

where N_1^c, N_2^c are constants, θ^c is a given threshold, n_1^c is the number of corners in the input corner image I_C and n_2^c is the number of corner matching between I_C and the corner map image M_C.

$$n_1^c \approx \frac{1}{255}\sum I_C(x,y) \qquad n_2^c \approx \frac{1}{255\times 255}\sum I_C(x,y)\cdot M_C(x,y)\,. \tag{8}$$

Then, qualification of whether the input image satisfies corner map or not is

$$Q_C = C_1^C(I_C) \cap C_2^C(I_C) \cap C_3^C(I_C)\,. \tag{9}$$

Satisfying orientation. From the input image I(x,y), we find the orientation of each pixel by using HOG descriptor method as in [2]. The magnitude m(x, y) and the orientation θ(x, y) are computed by

$$\begin{aligned} m(x,y) &= \sqrt{g_x(x,y)^2 + g_y(x,y)^2} \\ \theta(x,y) &= \tan^{-1}(g_y(x,y)/g_x(x,y)) \end{aligned}\,. \tag{10}$$

where $g_x(x, y)$ and $g_y(x, y)$ denotes the x and y components of the image gradient in x and y direction calculated by 1-D centered mask [−1, 0, 1]

$$g_x(x, y) = I(x + 1, y) - I(x - 1, y)\,. \tag{11}$$

$$g_y(x, y) = I(x + 1, y) - I(x - 1, y)\,. \tag{12}$$

After that, we specify some sub regions of object where the orientation is so strong such as horizontal, vertical and diagonal.

Figure 5. Sub regions with strong orientation, from left to right: edge image, horizontal orientation, diagonal orientation and vertical orientation.

The orientation is ranged from [0 − 2π], we divide into 16 bins *h[0]... h[15]*. For each sub regions, we calculate histogram of orientation in that region by quantizing the orientation $\theta(x, y)$ for all pixels falling to *h[i]* bins, $i = \overline{0,15}$, weighted by its magnitude $m(x, y)$.

Condition for input image satisfies orientaiton is that

$$H(S) = \begin{cases} 1 & \dfrac{h[0]+h[15]+h[7]+h[8]}{\sum_{i=0}^{15} h[i]} > \sigma_1 \\ 0 & else \end{cases} \quad (13)$$

$$V(S) = \begin{cases} 1 & \dfrac{h[3]+h[4]+h[11]+h[12]}{\sum_{i=0}^{15} h[i]} > \sigma_2 \\ 0 & else \end{cases} \quad (14)$$

$$D(S) = \begin{cases} 1 & \dfrac{h[1]+h[2]+h[9]+h[10]}{\sum_{i=0}^{15} h[i]} > \sigma_3 \\ 0 & else \end{cases} \quad (15)$$

where σ_1, σ_2, and σ_3 are specific thresholds. If all sub-regions satisfy condition of the strong orientation respectively then input image satisfies orientation.

4. RESULT AND EVALUATION

Our proposed scheme is evaluated on five object-categories, including front/rear view car, side view car, bicycle, train and airplane. We use CALTECH image databased for testing with front/rear view car, bicyle, and airplane. While side view car database is downloaded from UIUC's webpage. With train object, we have to select manualy from internet. Beside, in order to evaluate the result more exactly, we also get object image from the officcial website of PASCAL-VOC 2009. Morover, an additional about 1500 negative examples are collected randomly from many websites and added to the database. Thus with each object-category, the whole database contains about more than 1500 negative images and 1500 positive images (including 1000 images for training to make edge map and corner map, and 500 images for testing).

Image databases contain natural images that are taken from several sources and include occlusion and cluttered backgrounds. With side view car, images used for training have the same size 40x100 as in original database. Otherwhile, all images in the training set of front/rear view car, we resize manually into 60x80 to eliminate the influence from background. The basic size of bicycle image is set to 40x80 and, airplane's is 50x150. Train object is specially, because train has no specific size. The length of train object changes very much. This is one of our obstacle when testing with train object. Because if train object has no fix-size, then it can't make Edge

map and Corner map. The safest way we choose is that trying to detect one coach in stead of whole train. After detecting some coachs, the post-processing will be done to make the whole train from these coachs. Size of one coach is 40x150. With a testing image, we preprocess it by resizing it into four times as size of training image corresponding with their object. After detecting at this scale, we reduce it size and continue detect until it's size is equal to training image. This work is to be able to detect all object in the input image even if the size of object in it is small or large. Some of our results for each object-category are shown as bellow.

Figure 6. Example of front/rear view car detection result.

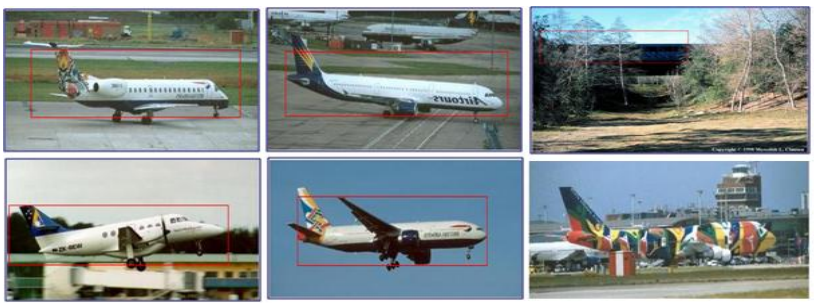

Figure 7. Example of areoplane detection result.

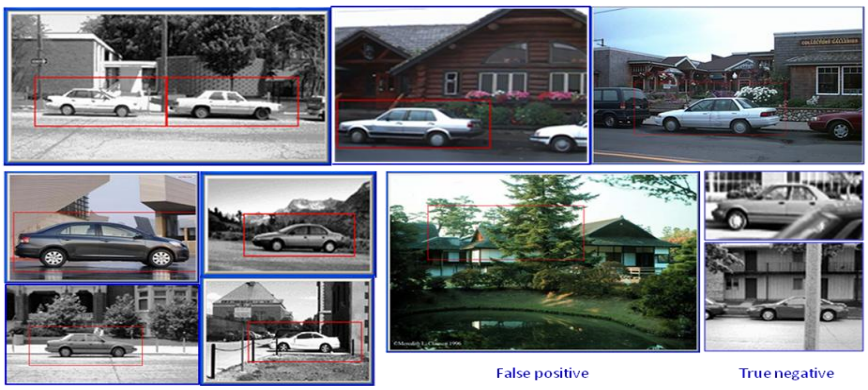

Figure 8. Example of front/rear view car detection result.

The result of our system is shown in Table 1. The performance of Zhenfeng Zhu [11] for car detection with the same UIUC database, reachs the highest accurate rate at 90.5%, while the lowest accurate of our sytem for car is about 93.24%. This shows that our shema has a better perfomence. For space consideration only some examples of correct detection are shown in Fig. 6 and Fig. 7. In each figure we present both correct detection (true positive) and misdetection (true negative and false positive) of our implementation. It shows that if the input image is so blur or is cut-edge some part of the object, then our shema cannot detect and it leads to true negative. Otherwhile, if there are edges and corners which are arranged similar to the object then our shema knows that is object wanted to detect (false positive).

Table I. Result of object detection

(a) our system

Object	Correct rate	False rate
Front/rear car	93.24%	6.76%
Side car	93.31%	6.69%
Bicycle	85.29%	14.71%
Train	81.82%	18.18%
Plane	95.21%	4.79%

(b) PASCAL 2009[1]

Object	Correct rate	False rate	Authors
Car	72.50%	27.50%	NECUIUC_CLS-DTCT
Bicycle	68.60%	31.40%	NECUIUC_CLS-DTCT
Train	86.00%	14.00%	NECUIUC_CDCV
Plane	88.10%	11.90%	NECUIUC_CDCV

We know that these result is not enough to evaluate our system better than other or not. Because we only test on five categories, while in PASCAL every author's system must run on twenty categories. But anyway, the result of our system is more accurate and that encourage us to continue testing with other categories. In train object, our system has a lower result because the size of train is various. So we cannot create edge map and corner map correctly. Beside, to increase accuracy rate, we should combine our present approach with some out-of-the-art feature such as SURF/SIFT to classify the image before detecting. If we can combine classification and detection, the result is surely higher.

5. CONCLUSION

We propose a method for gerneric object detection. Our method is similar to the way that human being recognize object which is based on edge, corner and histogram of oriented gradient. We implement on five categories including front/rear view car, side car, bike, railtrain and areoplane. The result shows that our propose method not only is easy to understand with human, but also gets the higher accuracy rate compared with state-of-the-art methods. Another advantage of our method is that running time is not consume a lot. However, because this method compare the edge/corner version of input image with edge/corner map which was formed from training database image, if there are some occlusion especially in some important position of object (strong corner or strong edge) then our system can not detect object. In order to overcome this problem, we should combine both local and global information to evaluate more precise the score function of detecting object.

[1] http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2009/results/index.html

REFERENCES

1. C. Harris and M. Stephens - A combined corner and edge detector, proc. of the the 4th Alley Vision Conference, Manchester, 1988, pp. 147–151.
2. Navneet Dalal and Bill Triggs - Histograms of Oriented Gradients for Human Detection, proc. of the IEEE CVPR Vol. **II**, San Diego, CA, USA , June 2005, pp. 886-893.
3. Sonya Coleman, Bryan Scotney, Dermot Kerr - Integrated Edge and Corner Detection, proc. of the Conference on Image Analysis and Processing, Modena, Italy, Sep 2007.
4. H. Bay, T. Tuytelaars, L. V. Gool - Surf: Speeded up robust features, proc. of the 9th European Conerence on Computer Vision, Graz, Austria, May 2006.
5. K. Mikolajczyk, C. Schmid - A performance evaluation of local descriptors, IEEE Trans. on PAMI, Vol. **27** (10), Oct. 2005, pp. 1615–1630.
6. David Monzo, Alberto Albiol, Jorge Sastre, Antonio Albiol - HOG-EBGM vs. GABOR-EBGM, proc. of the 15th IEEE ICIP, San Diego, CA, Oct. 2008, pp. 1636-1639.
7. O.A. Zuniga, R.M. Haralick - Corner detection using facet model, proc. Conference on Pattern Recognition and Image Processing, 1983, pp. 30-37.
8. H.P. Moravec - Towards automatic visual obstacle avoidance, proc. of the 5th International Joint Conference on Artificial Intelligent", 1977, pp. 584.
9. D. G. Lowe - Distinctive image features from scale invariant keypoints, International Journal of Computer Vision, 60(2), 2004, pp. 91–110 .
10. Rafajlowicz - SUSAN edge detector reinterpreted, simplified and modified, International Workshop on Multidimensional (nD) Systems, Portugal, p.69-74, June 2007, pp. 69-74.
11. Zhenfeng Zhu, Hanqing Lu Hu, J. Uchimura - Car detection based on multi-cues integration, proc. of the 17th ICPR, Vol. **II**, Cambridge, UK, Aug. 2004, pp. 699-702.
12. K. Rangarajan, M. Shah and D.V. Brackle - Optimal corner detector, Computing Vision, Graphics, and Image Processing, Vol. **48**, 1989, pp. 230-245.
13. E. Rosten, T. Drummond - Machine learning for high-speed corner detection, proc. of the 9th ECCV, Graz, Austria, May 2006, pp. 430-443.
14. L. Kitchen, A. Rosenfeld - Gray level corner detector, Pattern Recognition Letters, Vol. **I**, 1982, pp. 95-102 .
15. J. F. Canny - A computational approach to edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI **8**(6), Nov 1986, pp. 679–698.
16. H. Harzallah, F. Jurie, C. Schmid - Combining efficient object localization and image classification," proc. of the 12th ICCV, Kyoto, Japan, Sep. 2009.

Address: 520-0054 Kusatsu Shi, Shiga Ken,
Higashi yagura 2-10-2, Japan.

Received Dec 12, 2009

Bao Nguyen Thien,

Computer Vision Lab, Human and Computer Intelligence Department, Ritsumeikan, Japan

Tadashi Matsuo,

Computer Vision Lab, Human and Computer Intelligence Department, Ritsumeikan, Japan