A SIMPLE APPROACH FOR EFFICIENTLY OBJECT DETECTING

Bao Nguyen Thien
Yoshiaki Shirai

Ritsumeikan University

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. Then we continuously present our experiment on applying this method for car detecting in part three. In part four, we present our result and a comparative evaluation of different methods versus our method.

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 car-detecting shows that it is true.

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.

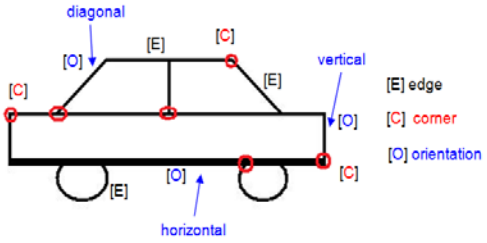

Fig. 1 Three main components make object be figured.

Shape or boundary is basically componented by *edges*. These edges are arranged in an order and meet each other at some points called *corners*. 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 for these principal components for making 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:

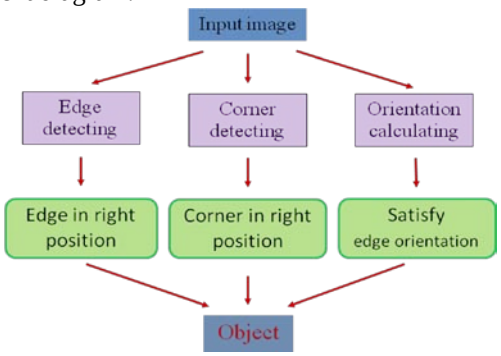

Fig. 2 A model for object detecting based on edge, corner and edge orientation.

But there is still an issue in this model. That is how do we know that edge/corner is at right position or not? In order to answer this question, we must point out the place where edge/corner must belong to before 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 one 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]. 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 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 show 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 ON CAR DETECTING

We apply our approach for car detecting. We choose car detection because this task has been done by many researcher around the world. So we can easily compare the result of our method with other 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 car 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. Then, our car detecting system includes two main steps: making edge map and corner map by using machine learning; and detecting car 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. However, [11] 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. So, gray scale map can work better in this case.

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

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

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

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. Car detecting base on edge, corner and orientation

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. And to consider the position of corners, Harris corner detector is used with 5x5 structure. Beside, Gaussian

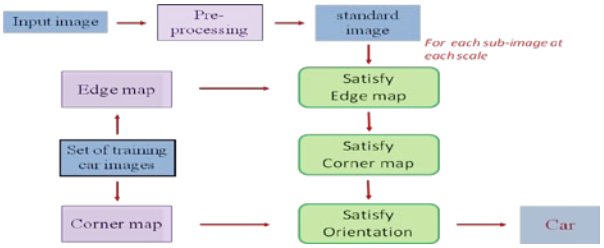

Fig. 4 Car detection based on edge, corner and orientation

filter is applied on input image before extracting edge and corner features to subtract background.

3.2.1. Satisfying edge map

Edge image after detecting edge from input one is denoted as I_E. Let's define two parameters:

n_1^e : number of edges in the input edge image I_E.

n_2^e : 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:

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

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

where N_1^e, N_2^e are constants and θ^e is a given threshold. $C_1^E(I_E)$ 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 $C_3^E(I_E)$ is to avoid situation that there are so many edges in the input image. So, if $Q_E = C_1^E(I_E) \cap C_2^E(I_E) \cap C_3^E(I_E)$ (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

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

here, because both I_E and M_E are gray scale, we should divide the sum to 255 for each image.

3.2.2. Satisfying corner map

Similarity to the edge test step, in this stage we also define three conditions:

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

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

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.

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

Then, qualification of whether the input image satisfies corner map or not is $Q_C = C_1^C(I_C) \cap C_2^C(I_C) \cap C_3^C(I_C)$ (9)

3.2.3. 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

$$m(x,y)=\sqrt{g_x(x,y)^2+g_y(x,y)^2}$$
$$\theta(x,y)=\tan^{-1}(g_y(x,y)/g_x(x,y)) \quad (10)$$

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]

$$g_x(x, y) = I(x + 1, y) − I(x − 1, y) \quad (11)$$
$$g_y(x, y) = I(x + 1, y) − I(x − 1, y)$$

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

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

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

$$H(S)=\begin{cases}1 & \frac{h[0]+h[15]+h[7]+h[8]}{\sum_{i=0}^{15}h[i]} > \sigma_1 \\ 0 & else\end{cases} \quad (12)$$

$$V(S)=\begin{cases}1 & \frac{h[3]+h[4]+h[11]+h[12]}{\sum_{i=0}^{15}h[i]} > \sigma_1 \\ 0 & else\end{cases} \quad (13)$$

$$D(S)=\begin{cases}1 & \frac{h[1]+h[2]+h[9]+h[10]}{\sum_{i=0}^{15}h[i]} > \sigma_1 \\ 0 & else\end{cases} \quad (14)$$

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

4. RESULT AND EVALUATION

Our proposed scheme is evaluated on both front/rear view car and side view car. We use CALTECH image databased for testing with front/rear view car, while side view car database is downloaded from UIUC's webpage. Beside, an additional 165 negative examples collected from website are added to the database. Thus, with

front/rear view car, the whole database contains 665 negative images and 1553 positive images including 253 images for training to make edge map and corner map. With side view car, we have 665 negative images and 828 positive images containing 250 training images.

Both of two databases also contain natural images that are taken from several sources and include occlusion and cluttered backgrounds. With side view, 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. With a testing image, we preprocess it by resizing it into four times as size of training image. After detecting at this scale, we reduce it size and continue detect until it's size is equal to training iamge. This work is to be able to detect all car in the input image even if the size of car in it is small or large.

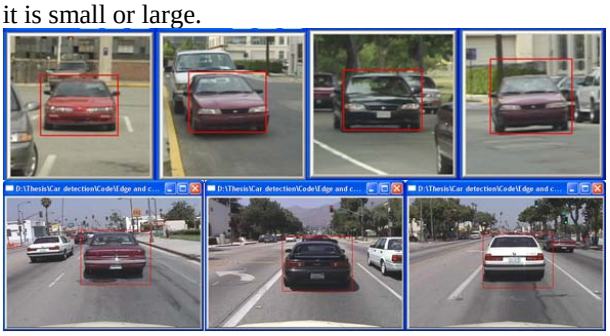

Fig. 6 Example of correct detection with front/rear view.

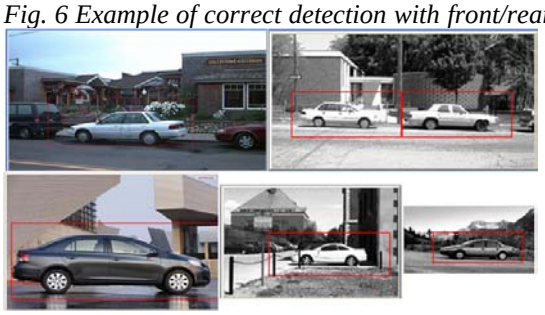

Fig. 8 Example of correct detection with side view.

Fig. 9 Example of false detection: negative false of front view, positive false, negative false of side view.

The result of our system is shown in Table 1. The performance of Zhenfeng Zhu [11] reachs the highest accurate rate at 90.5%, while the lowest accurate of our sytem is about 91.67%. This shows that our shema has a better perfomence. For space consideration only some examples of correct detection are shown in Fig.7 and Fig. 8. In Fig. 9 we present negative false and positive false of our implementation. It is show 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 negative false. Otherwhile, if there are edges and corners which are arranged similar to the object then our shema knows that is object wanted to detect.

Table 1. Result of our car detecting system

Set	No. of image	Correct	In-correct	False rate	Accuracy rate
Calt	504	481	23	4.57%	95.43%
Calt-09	526	475	41	8.95%	92.05%
Non- car	665	624	41	6.17%	93.83%
Detection result of front/rear view car					
UIUC1	300	275	25	8.33%	91.67%
UIUC2	278	264	14	5.04%	94.96%
Non-car	665	638	27	4.06%	95.94%
Detection result of side view car					

In this paper we present a simple appoach for object detecting. Our implementation on car detection shows that our shema can work with high accurate rate. For our future scopes, we will continue testing with other objects and improve our system to be better.

References

[1] C. Harris and M. Stephens, "A combined corner and edge detector," *proc. of the the 4th Alvey Vision Conference*, Manchester**,** pages 147–151, 1988.

[2] Navneet Dalal and Bill Triggs, "Histograms of Oriented Gradients for Human Detection," *proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. II, San Diego, CA, USA , pages 886-893, June 2005.

[3] Sonya Coleman, Bryan Scotney, and Dermot Kerr, "Integrated Edge and Corner Detection," *proc. of the International Conference on Image Analysis and Processing*, Modena, Italy, Sep 2007.

[4] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," *Proc. of the 9th European Conerence. on CV*, Graz, Austria, May 2006.

[5] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. on PAMI*, vol. 27, no. 10, pages 1615–1630, Oct. 2005.

[6] David Monzo, Alberto Albiol, Jorge Sastre, Antonio Albiol, "HOG-EBGM VS. GABOR-EBGM," *proc. of the 15th IEEE International Conference on Image Processing*, San Diego, CA, pages 1636-1639, Oct. 2008.

[7] O.A. Zuniga and R.M. Haralick, "Corner detection using facet model", *Proc. Conference on Pattern Recognition and Image Processing*, pages 30-37, 1983.

[8] H.P. Moravec, "Towards automatic visual obstacle avoidance", *proc. of the 5th International Joint Conference on Artificial Intelligent*", pages 584, 1977.

[9] D. G. Lowe, "Distinctive image features from scale invariant keypoints," *International Journal of Computer Vision, 60(2)*, pages 91–110, 2004.

[10] Rafajlowicz,"SUSAN edge detector reinterpreted, simplified and modified", *International Workshop on Multidimensional (nD) Systems*, Portugal, p.69-74, June 2007.

[11] Zhenfeng Zhu Hanqing Lu Hu, J. Uchimura, K, "Car detection based on multi-cues integration," *proc. of the 17th International Conference on Pattern Recognition* (ICPR), vol. II, Cambridge, UK, pages 699-702, Aug. 2004.

[12] K. Rangarajan, M. Shah and D.V. Brackle, "Optimal corner detector," *Computing Vision, Graphics, and Image Processing*, Vol. 48, pages 230-245, 1989.