International Journal of Information Technology, Vol. 12 No. 2 2006

An MRF Model Based Algorithm of Triangular Shape Object Detection in Color Images

Yangxing LIU[†], Satoshi GOTO[‡], Takeshi IKENAGA[‡]

2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Graduate School of Information, Production and Systems, Waseda University, 808-0135, Japan
lyx@ruri.waseda.jp[†]
{GOTO, IKENAGA}@waseda.jp[‡]

Abstract

Triangular shape object detection in color images is an important issue in computer vision. However, there are few reports on this matter. In this paper, we proposed a novel approach, which combines a global contour based line segment detection algorithm and Markov Random Field (MRF) Model, to extract triangular objects from color images. First, we use an elaborate edge detection algorithm to obtain image edge map and accurate edge pixel gradient information. Then line segments are extracted from the edge map and some neighboring parallel segments are merged into a single line segment. Finally all segments lying on the boundary of unknown triangular objects are labeled via an MRF Model built on line segments. Experimental results show that our method is robust in locating multiple triangular objects simultaneously with respect to different size, orientation and color.

1 Introduction

Triangular shape object detection (TSOD) is one of the fundamental tasks of computer vision and is especially important in some applications such as object tracking, visual inspection, and man-made object detection, which frequently has shapes with triangular edges. For example, some traffic signs, which indicate danger or a threat, often have equilateral triangular shapes. Since the shapes of traffic signs do not vary regardless of weather or lighting conditions, it is a robust feature that we can exploit to detect traffic signs in driver support system.

Shape-based object detection is one of the difficult tasks in computer vision systems. Although it has been studied for dozens of years, a robust, accurate and high performance approach is still a great challenge today.

Many methods have been previously used to extract geometric primitives from image data. The most popular methods are variations on the Hough transform (HT) [1]. Conventional HT is very time-consuming and expensive in memory, especially for triangle detection, which has six unknown parameters to be estimated. The Randomized HT [2] was originally designed mainly for analytical curve extraction, such as line, circle etc. Its application to direct triangle detection is rarely investigated.

Roth et al [3] proved that the shape detection problem can be formulated in term of an optimization problem, so the genetic algorithm is proposed to solve this problem [4]. However it is only applied to binary image and its optimization process is time-consuming. Moreover, it does not go far enough to detect multi-object simultaneously.

Currently, few papers about TSOD from color images have been published. In this paper, we propose a novel TSOD approach, targeted towards being robust with respect to diverse kinds of triangular object appearances, including object size, orientation, and color. The method is significantly different from conventional TSOD techniques, which directly estimate the 6 parameters (three vertexes) of an arbitrarily oriented triangle to extract triangular objects with high time complexity and space requirement. The basic idea of our algorithm is to use an MRF model built on detected line segments in image edge map to label certain line segments, which lie on the boundary of triangular objects. First, with an elaborate edge detection algorithm based on differential geometry, we obtain image edge map and precise gradient direction of edge pixel. Second we link edge pixels with similar orientation into straight line segments and group neighboring parallel segments into a single line segment. Eventually, an MRF model is built to label line segments belonging to different triangular objects. The randomness is used to model the uncertainty in the assignment of the labels.

The rest of this paper is organized as follows. Section 2 describes the color image edge extraction technique used in our method. The line segment detection and combination algorithm is discussed in section 3. Section 4 discusses our MRF model and its application to triangle detection. Experimental results are explained in section 5. Conclusion remarks are given in the last section.

2 Image edge Extraction

First, we use an accurate isotropic edge detector based on multi-dimensional gradient analysis to obtain image edge information. The edge detector we adopted is different from most existing edge detectors, which cannot provide accurate edge direction information [5]. Some ideas about the detector have been described in [6].

2.1 Edge detector

Given a color image, the difference vector DV in rgb color space induced by moving an infinitesimal step in the image plane in the direction {dx,dy} is : DV=(dx dy) J_c^T , where the matrix J_c is the Jacobian matrix of the image.

The Euclidean squared magnitude of DV is: $DV^2=(dx\ \ dy)\, M_c\, (dx\ \ dy)^T$, where

$$M_c = J_c^T J_c = \begin{bmatrix} M_{xx} & M_{xy} \\ M_{xy} & M_{yy} \end{bmatrix},$$

$$M_{xx} = r_x^{\,2} + g_x^{\,2} + b_x^{\,2},$$

$$M_{xy} = r_x \times r_y + g_x \times g_y + b_x \times b_y,$$

$$M_{yy} = r_y^{\,2} + g_y^{\,2} + b_y^{\,2},$$

where r_x, r_y, g_x, g_y,b_x and b_y, are the six first order derivatives of the three color channels with respect to x and y. Asking for the direction of {dx,dy} maximizing this magnitude is an eigenvalue problem. We can obtain the magnitude extremum in the direction of the eigenvector of the matrix M_c and the extremum value is the corresponding eigenvalue. The larger eigenvalue of M_c is

$$V=(M_{xx}+M_{yy}+\sqrt{(M_{xx}+M_{yy})^2-4\times(M_{xx}\times M_{yy}-M_{xy}^2)})/2$$

and its corresponding eigenvector is $\{M_{xy}, V-M_{xx}\}$. The gradient direction angle θ is defined by eigenvector:

$$\theta = \operatorname{arctg}\frac{v - M_{xx}}{M_{xy}}. \tag{1}$$

The square root of the larger eigenvalue and its corresponding eigenvector direction are the equivalents of the gradient magnitude and gradient direction at any given point.

2.2 Edge Detection algorithm

We exploit image pixel gradient information, spatial relation and region property to obtain image edge map. A selection-and-verification scheme is performed. First we select edge pixel candidates according to image pixel eigenvector direction and eigenvalue magnitude. Then spatial information and region based analysis are integrated to verify all candidate edge pixels.

1) Candidate edge pixel selection

First of all, the matrix M_c is computed for each pixel on the image and we will get a series of eigenvalues V and eigenvectors E for all pixels.

Edge pixels are those points with local maximum gradient magnitudes in their gradient directions. So a pixel is kept as a candidate edge pixel only if it has a larger gradient magnitude than that of its neighbors located in the direction closest to its gradient direction, i.e., the eigenvalue of an edge pixel must be greater than that of both two neighboring points, which are closest to its eigenvector direction.

2) Edge pixel verification

To avoid detecting false edge pixels, such as noise points, the following two operations incorporating spatial information and region based analysis are performed.

- First if a potential edge point has no potential edge point in its 8 neighboring points, it must be treated as noise and removed from edge point collection.
- Second we also noticed that the gradient magnitude variance of two adjacent points in the same region is not apparent. So if the corresponding eigenvalue difference between a candidate edge pixel and its eight neighboring labeled edge pixels is always very large, the candidate edge pixel will not be selected as an edge pixel. Inequality (2) is defined to evaluate the difference.

$$\left(\sum_{0}^{Num(i,j)-1}\left|V(i,j)-V(i+p,j+q)\right|-\sum_{0}^{Num(i,j)-1}V(i+p,j+q)\right)>0 \tag{2}$$

Given a candidate edge pixel (i,j), Num(i,j) is the count of its neighboring labeled edge pixel (i+p,j+q) and Num(i,j) value could be 1,..,7, with p and q $\in$ {-1, 0, 1}. If (2) is true, the pixel (i,j) will be labeled as a non-edge point.

3 Line segment detection and combination

After we get the edge map, we first need to link neighboring edge pixels with similar orientation to a set of straight line segments to prepare data for later triangle detection.

3.1 Line segment detection algorithm

In this stage, we group neighboring edge pixels with similar orientation into straight line segments. Because edge pixels are an unordered list of points in 2-dimensional Cartesian space, like HT we fist map each pixel(x,y) into a curve $\rho = \mathrm{x}\cos\vartheta + \mathrm{y}\sin\vartheta$ in $\rho - \vartheta$ plane, where ρ is the distance of the desired line from the origin and ϑ is the angle that the normal to the line makes with positive x-axis.

Although HT can find straight lines on which the image edges lie, a straight line in an image has two ends and HT does not find end-points. So we will store the two end coordinates corresponding to a point (ρ_0, ϑ_0) in $\rho - \vartheta$ plane, which can be mapped into a line $\rho_0 = \mathrm{x}\cos\vartheta_0 + \mathrm{y}\sin\vartheta_0$ in x-y plane. Furthermore, edge pixel orientation obtained in previous stage can greatly reduce computation burden because each edge pixel only votes for one bin in the accumulator array with ϑ fixed by gradient direction.

To fit a straight line to a set of points, we first quantize ρ and ϑ with $\delta\vartheta = 2$ and $\delta\rho = 1$. Then the following steps are performed to detect line segments from the edge map.

- Scan the image from left to right and from top to bottom.
- For each edge pixel (i,j), we compute its polar coordinate $\rho(\mathrm{i,j}) = \mathrm{i}\times\cos\theta(\mathrm{i,j}) + \mathrm{j}\times\sin\theta(\mathrm{i,j})$ in $\rho - \vartheta$ plane, where $\theta(\mathrm{i,j})$ is its gradient direction angle. After this, we will group (i,j) according to following two cases:
  1) If no line segment lying along the vertical direction defined by $\theta(\mathrm{i,j})$ is connected with (i,j), then a new line segment is generated with its start and end labeled with (i,j), else
  2) The starting and ending pixels of the line segment, which has the same oriention and connects with (i,j), are updated by computing the Manhattan distance between two ends and (i,j).

3.2 Line segment combination

Due to effect of noise or illumination variation, line segments detected in former procedure may be highly fragmented and a grouping process is necessary. Moreover, line segments combination will reduce the time complexity of later triangle boundary detection process. Therefore, we merge some straight line segments into a single segment according to several perceptual grouping criteria.

- First, the merged line segments of course, must have the same orientations.
- Second, the distance between line segments should be very small.
- Third, parallel line segments should not overlap a significant portion when they are projected in the vertical direction of line direction.

4 Triangle detection algorithm

After obtaining all line segments, we select line segments belonging to different triangular objects in image through an MRF model. MRF[7] has been demonstrated to be successfully applied to many computer vision tasks. Because the probability of a line segment being one triangle side mainly depends on that of its neighbors, we can build an MRF model to label line segments being triangle sides or not.

Let $L = \{l_1, l_2 \dots l_n\}$ be the line segment set we obtained in previous process. The set of sites $d = \{1, 2, \dots n\}$ indexes L. Let $F=\{F_1, F_2 \dots F_n\}$ be a family of random variables, in which each random variable F_i takes its value from 0 to 1 and indicates whether a line segment l_i lies on the boundary of a triangular shape object or not. When $F_i = 1$, l_i is regarded as one triangle side.

Because neighborhood selection is an important issue in MRF, we will first introduce the neighborhood system used in our algorithm.

4.1 Neighborhood system

Let $N(l_i)$ be the set of all line segments in L that are neighbors of l_i such that

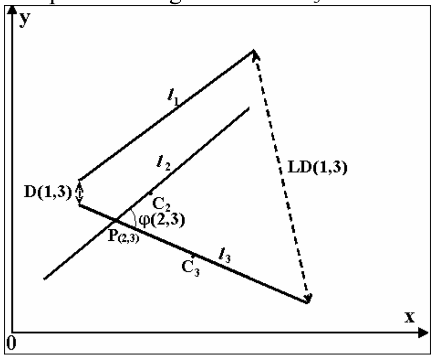
$$l_i \notin N(l_i) \text{ and if } l_j \in N(l_i), \text{ then } l_i \in N(l_j).$$

In our algorithm, one line segment l_j will be regarded as one neighborhood of l_i only if it meets the following three requirements.

- First, the distance D(i,j) between l_i and l_j should not be too large.
  D(i,j) equals to the minimum distance between each end of one line segment and that of another line segment. In this case, D(i,j) should be smaller than twenty percent of each line segment length, i.e. $D(i, j) < (\min(LEN_i, LEN_j) \times 0.2)$, where LEN_i is the length of a line segment l_i.
  In Fig. 1, D(1,3) is the distance between line segment l_1 and l_3.
- Second, the angle $\varphi(i, j)$ between l_i and l_j should be greater than the TH^L and not exceed the threshold TH^U. TH^L is experimentally set to 0.17, about 10 degrees and $TH^U = PI - TH^L$. TH^L and TH^U can also be set to different values depending on different applications.

In Fig. 1, if $\varphi(2,3) < TH^L$ or $\varphi(2,3) > TH^U$, l_2 and l_3 will not be regarded as neighboring line segments in our method.

- Third, if l_i and l_j intersect in a single point $P_{(i,j)}$, $P_{(i,j)}$ should not close to the middle point of each line segment.
  In figure 1, l_2 and l_3 intersect in $P_{(2,3)}$, point C_2 and C_3 are middle points of l_2 and l_3. Only if $\left|P_{(2,3)}C_2\right| < 0.15 \times LEN_2$ and $\left|P_{(2,3)}C_3\right| < 0.15 \times LEN_3$, then line l_2 is a possible neighbor of line l_3.



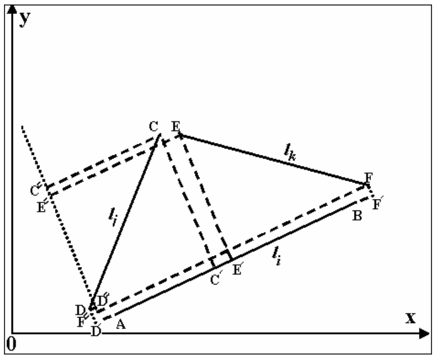


Fig. 1 Illustration of spatial relations among line segments.

Fig. 2 Illustration of line segment projection.

With the neighborhood system defined above, the field F can be assumed to be an MRF with its local characteristics.

$$\text{For } i \in d,\ P(F_i|\{F_j, j \in d, j \neq i\}) = P(F_i|\{F_j, j \in N(l_i)\}).$$

4.2 Labeling triangle sides

Let $f=\{f_1, f_2 \ldots f_n\}$ be a configuration of F, i.e. $\{F_1 = f_1 \ldots F_n = f_n\}$ and $N=\{N(l_i)|\ \forall i \in d\}$ be the collections of line segments neighboring to one line segment. Then we can calculate the posterior probability following Gibbs distribution as follows:

$$P(F=f\,|L) = Z^{-1}\exp[-E(f)/T] \qquad (3)$$

where T is the temperature which is assumed to be one unless, otherwise stated, Z is the normalization factor and E(f) is the posterior energy function.

Now, we can define the triangle side extraction as the following optimization problem,

$$\text{For a given } L,\ \arg\max P(f_i \mid l_i) \text{ for each } l_i,$$

which is also equivalently found by

$$\arg\min E(f).$$

Minimizing the energy function will maximize the probability defined by (3), which will give the maximum posterior estimate of potential triangles' sides in image edge map.

Some notations used in our energy function are illustrated in Fig. 1 and Fig. 2. LD(i,j) is the distance between two ends of l_i and l_j, which are not the nearest end pairs. In Fig. 1, the dashed arrow line length equals to LD(1,3). V(i,j,k) is the overlapping projection of two line segments l_j and l_k along the perpendicular direction of line segment l_i. H(i,j) denotes the projection of line segment l_j in the line segment l_i. In Fig. 2, $C'D'$ and $E'F'$ are the projection of l_j and l_k along the direction of l_i, $C''D''$ and $E''F''$ are the projection of l_j and l_k along the perpendicular direction of l_i. V(i,j,k) equals to length of $D''E''$, H(i,j) equals to length of AC', and H(i,k) equals to length of BE'.

The energy function E of our model is minimized while each f_i is converged to either 0 or 1. E consists of the following four terms:

$$E_1=\alpha_1\sum_{i=1}^{n}\sum_{j\in N(l_i)} f_i f_j D(i,j), \qquad E_2=\alpha_2\sum_{i=1}^{n}\left(\frac{f_i}{LEN_i}\times AVGLEN\right),$$

$$E_3=-\alpha_3\sum_{i=1}^{n}(f_i \ln f_i+(1-f_i)\ln(1-f_i)),$$

$$E_4=$$

$$\alpha_4\sum_{i=1}^{n}\frac{1}{LEN_i}\sum_{j\in N(l_i)} f_i f_j \times \min_{k\in\{N(l_j)\cap N(l_i)\}}\left(\frac{D(k,j)+D(k,i)}{f_k V(i,j,k)(H(i,j)+H(i,k))}\times LD(j,k)\right)$$

where α_1, α_2, α_3 and α_4 are positive constants, which control the contributions of the individual energy term to the value of the energy function. AVGLEN is the average length of line segment set *L*.

E_1 favors grouping line segments that are close to each other. E_2 favors long line segments. E_3 is the entropy of the configuration $\{f_i\}$ hence pushes the $\{f_i\}$ to either 0 or 1. E_4 favors three neighboring segments, which are most likely to compose a triangle.

The {fi} are all initialized to 0.5 and the energy is gradually reduced by using a gradient descent algorithm with a learning rate scheduling scheme.

$$\{f_i\}_{t+1}=\{f_i\}_t-\eta_t\nabla E,$$

$$\eta_{t+1}=\eta_t-\beta,$$

where η is a positive step-size parameter, which influences convergence rate and final labeling result, β is a small constant, which makes the step size linearly decreased at each iteration, and ∇E is the gradient of E.

Upon minimizing the energy, we can select certain line segments, which are boundary segments of unknown triangular shape objects in image, with its label parameter fi $\approx$ 1 from L. Then three sides of the same triangle will be grouped together by the spatial information and geometrical relations among line segments, which can be obtained from our neighborhood system.

5 Experimental results

In order to evaluate the actual performance of our proposed algorithm, we implemented the algorithm in C++ language under Windows-XP on an EPSON Endeavor MT7000 PC equipped with Intel PIV 2.4GHz processor and 1G RAM. 130 real and synthetic color images are tested, which include variant triangular shape objects. The resolution of test images varies from 128*128 to 3072*2048 pixels.

During the experimentation, considering tradeoff between cost and performance, we set different values for parameter η_0(initial learning rate). If η_0 is too small, our algorithm will take a long time to converge. If η_0 is too large, our algorithm may oscillate and the detection result becomes unstable. So η_0 is set to different value according to variant image complexity such that

$$\eta_0 = \frac{\text{number of line segments in image}}{(\text{image width}) \times (\text{image height})}.$$

For measuring accuracy, recall rate and precision are calculated to evaluate our algorithm performance. Recall rate is the ratio between the number of detected triangular objects and number of triangular objects in images. Precision equals to the ratio between the number of detected triangular objects and the summation of number of detected triangular objects and number of false alarms (detected non-triangular objects).

Due to limited space, only one test image with 512*512 pixels is illustrated in Fig. 3. Figure 4 shows the image edge map obtained by using our edge extraction algorithm discussed in section 2, which has 27062 edge pixels. The precise gradient direction of each edge pixel computed from equation (1) is represented with short green lines in Figure 5. In Figure 6, we depict all line segments detected from the image edge map with red line segments, whose number is 8289. Then triangular shape objects are detected via an MRF model built on line segments and the final detection result is shown in Fig. 7, where green line segments represent the triangular shape object boundary(the parameters used was $\eta_0 = 0.0316$). The total processing time of this image is 1.625 seconds. The edge detection process takes only 0.188 seconds and the line segment detection and combination process consumes 0.250 seconds.

The performance achieved by our algorithm is summarized in table 1, from which we can see that most of the CPU processing time is spent on triangle detection. Compared with triangle detection algorithm in [8], which directly estimate the 6 parameters of an arbitrarily oriented triangle with high time complexity and space requirement, our algorithm explores the merit of MRF to allow a global optimization problem to be simplified and solved locally, whereby the computation cost is minimized.

Table 1 shows evidence that our algorithm has high recall rate and precision. But we also noticed that one problem associated with our algorithm is that when triangular shape object boundary in color images is blurred with cast shadowed by other objects, it is difficult to get precise edge information and locate such object accurately. So we hope to integrate color constancy technique to our algorithm to make the triangular shape object detection approach more robust in future.

6 Conclusions

In this paper, we regard the triangle detection task as an optimization problem and propose an MRF model based algorithm to extract multiple triangular shape objects simultaneously from color images. First we utilize an elaborate edge extraction algorithm to obtain image edge map. Then all line segments in image edge map are extracted efficiently with precise edge pixel gradient information. Eventually an MRF model built on line segments with a careful selection of neighborhood system is employed to classify certain line segments, each of which is one side of triangular shape object.

References

1. J. lllingworth, J. Kittler: A survey of the Hough transform. Computer Vision, Graphics and Image Processing, Vol. 44. (1998) 87-116
2. H. Kalviainen, P. Hirvonen: An extension to the randomized Hough transform exploiting connectivity. Pattern Recognition Letters. (1997) 77-85
3. G. Roth, M.D. Levine: Extracting Geometric Primitives. Computer Vision, Graphics and Image Processing, Vol. 58. (1993) 1-22
4. Roth G., Levine M.D: Geometric primitive extraction using a genetic algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 16, Issue 9. (1994) 901-905
5. Jianping Fan, Yau, D.K.Y., Elmagarmid, A.K., Aref, W.G.: Automatic image segmentation by integrating color-edge extraction and seeded region growing. IEEE Transactions on Image Processing, Vol. 10, Issue 10. (2001) 1454-1466
6. Lee, H.-C., Cok, D.R.: Detecting Boundaries in a Vector Field. IEEE Transactions on Signal Processing, Vol. 39, Issue 5. (1991) 1181-1194
7. Xiao Wang, Han Wang: Evolutionary optimization with Markov random field prior. IEEE Transactions on Evolutionary Computation, Vol. 8, Issue 6. (2004) 567-579
8. Hu, Z.Y., Tang, M., Tsui, H.T.: Direct triangle extraction by a randomized Hough technique. In: the 14th International Conference on Pattern Recognition, Vol. 1. (1998) 717-719

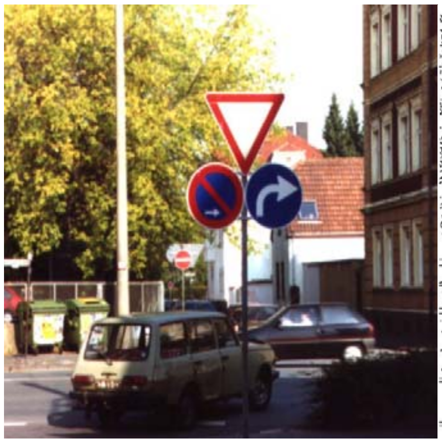


Fig. 3 Original color image.

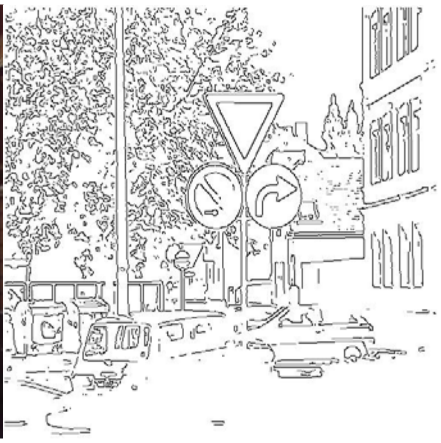


Fig. 4 Edge map of test image.

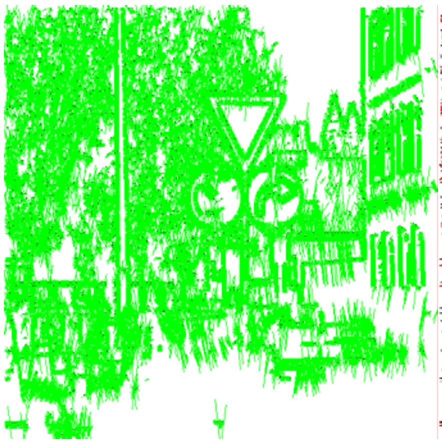


Fig. 5 Illustration of eigenvector direction.

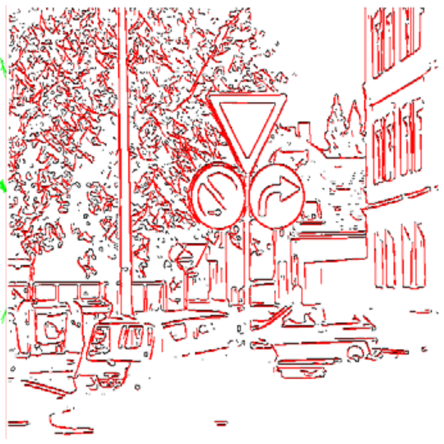


Fig. 6 Line segment detection result.

Table 1. Triangular shape object detection result.

N_S	N_{IM}	N_{TR}	Precision	Recall rate	T_E(s)	T_L(s)	T_{TR}(s)	T_T(s)
<650*780	75	68	92.8%	95.6%	0.187	0.241	4.478	4.906
<960*1280	28	19	90.4%	95.0%	0.459	0.940	17.445	18.844
<1786*1680	12	17	88.9%	94.1%	1.516	1.937	41.610	45.063
<3072*2048	15	13	85.7%	92.3%	3.683	5.286	70.257	79.226

N_S: Image size (pixels); N_{IM}: Number of images; N_{TR}: Number of triangular objects in images; T_E: Average execution time of image edge extraction process; T_L: Average execution time of line segment detection procedure; T_{TR}: Average execution time of triangle detection process; T_T: Average total execution time.

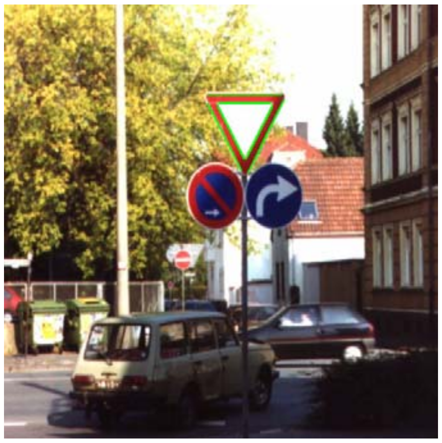


Fig. 7 Detected triangular object of test image ($\eta_0 = 0.0316$).

Yangxing LIU is a Ph.D. student of the Graduate School of Information, Production and Systems at Waseda University, Japan. He received the M.E. degree from the Department of Automation, Tsinghua University, China, in 2002. His research interests include pattern recognition, computer vision and VLSI design.

Satoshi GOTO born in Hiroshima, Japan, 1945. He received the B.E. and the M.E. degrees in Electronics and Communication Engineering from Waseda University in 1968 and 1970, respectively. He also received the Dr. of Engineering from the same university in 1981. He is IEEE Fellow, IEICE Fellow, Member of Academy Engineering Society of Japan, Professor of Waseda University. His research interests include LSI system and multimedia system.

Takeshi IKENAGA received the B.E. and M.E. degrees in electrical engineering and the Ph.D degree in information & computer science from Waseda University, Japan, in 1988, 1990, and 2002, respectively. He joined LSI Laboratories, NTT Corporation in 1990, where he has been undertaking research on the design and test methodologies for high-performance ASICs, a real-time MPEG2 encoder chipset, and a highly parallel LSI & system design for image processing. He is presently an associate professor in the system LSI field of the Graduate School of Information, Production and Systems, Waseda University. His current interests are application SoCs for image, security and network processing. Dr. Ikenaga is a member of the IPSJ and the IEEE. He received the IEICE Research Encouragement Award in 1992.