The Detection of Rectangular Shape Objects Using Matching Schema

Soo-Young Ye  
Department of Radiological Science, Catholic University of Pusan, Busan 46252, Korea

Joon-Young Choi and Ki-Gon Nam†  
Department of Electronics Engineering, Pusan National University, Busan 46241, Korea

Received August 19, 2016; Revised September 2, 2016; Accepted September 12, 2016

Rectangular shape detection plays an important role in many image recognition systems. However, it requires continued research for its improved performance. In this study, we propose a strong rectangular shape detection algorithm, which combines the canny edge and line detection algorithms based on the perpendicularity and parallelism of a rectangle. First, we use the canny edge detection algorithm in order to obtain an image edge map. We then find the edge of the contour by using the connected component and find each edge contour from the edge map by using a DP (douglas-peucker) algorithm, and convert the contour into a polyline segment by using a DP algorithm. Each of the segments is compared with each other to calculate parallelism, whether or not the segment intersects the perpendicularity intersecting corner necessary to detect the rectangular shape. Using the perpendicularity and the parallelism, the four best line segments are selected and whether a determined the rectangular shape about the combination. According to the result of the experiment, the proposed rectangular shape detection algorithm strongly showed the size, location, direction, and color of the various objects. In addition, the proposed algorithm is applied to the license plate detecting and it wants to show the strength of the results.

Keywords: Rectangular shape detection, Canny edge and line detection algorithm, Perpendicularity and parallelism of the rectangle

1. INTRODUCTION

Shape detection plays an important role in many recognition systems. Rectangular shape detection is one of the fundamental challenges in the field of computer vision. In particular, it is very useful in applications that require artificial object detection, such as traffic signs, building a roof with many rectangular shapes, etc. [1].

A study was carried out on rectangular shape detection in binary images or grey scale images [2-4]. However, due to the increase in the calculation speed of today’s computers, the hardware cost has decreased and color images have replaced the monochrome or grey scale images.

Color information can provide much information in video representation, showing a high reliability during image segmentation. Therefore, detecting rectangular shape information from the color image plays an important role in the application of computer vision. Although study in this field has been carried out for 10 years, a system for high detection performance has yet to be developed, which is a challenge needing to be addressed.

In addition, the proposed algorithm is applied to the license plate detecting and it wants to show the strength of the results.

The most common method was performed based on the HT (Hough transform) method [5]. Conventional HT requires significant time and memory space. Especially, it requires even more time and memory space when estimating the five unknown pa-
2. RECTANGULAR SHAPE DETECTION ALGORITHM USING THE MATCHING SCHEMA

2.1 Line segment detection using the canny edge detection algorithm

The methods of detecting straight segments are classified as the gradient-orientation-based method and gradient-magnitude-based method.

The method of detecting a straight line in a gradient-orientation-based method was proposed by Burns and his co-worker [12] and is used to define a region composed of image dots with a similar gradient in the direction of a straight line segment.

The recently proposed LSD (line segment detector) [13] detects exact straight line segments, and the error is controlled by using the Helmholtz principle, where straight line direction is effectively combined with the validity of the gradient. However, the threshold of the size of the gradient also removes useful information. When the threshold becomes lower, the detector detects effectively but the straight line accuracy is reduced. That is, when the magnitude of the gradient is small, the gradient direction shows an unstable characteristic.

The gradient-magnitude-based method first uses an edge detector to find the edge map and to detect a line segment from the edge map.

HT [5] is a useful method for detecting lines from more than threshold pixels in the edge map, but requires time complexity and space complexity.

Other post processing is also necessary in order to determine the correct length of the line element.

In order to solve this problem, Akinlar and his co-workers [14] proposed an efficient method for extracting a line segment from the edge line.

Akinlar [14] et al. proposed as follows. 1) Using the ED (edge drawing) algorithm [15], the edge line segment is extracted. 2) The least square method is then used to convert the edge segments in the line segment 3) Finally, the Helmholtz principle is used to remove the invalid line segments.

Most edge extraction masks are very sensitive to noise, even a small amount of noise that it is often regarded as an edge extraction. In order to compensate for these drawbacks, Canny proposed edge detector which is satisfied with these three standards ; good detection, good localization and clear response.

Actually, it showed strong edge detecting ability and it isn’t sensitive to noise.

In this study, the Canny edge detection algorithm was modified to detect the canny line. The canny line detection algorithm consists of three main parts as shown in Fig. 1.

1) Detecting an edge in the canny edge detection algorithm. 2) Forming the boundary list according to the connection element from the detected edge. 3) The detection DP algorithm is used to approximate the detected boundary list to the number of straight line segments. The detected straight line is represented by \( L = \{(x_1, y_1), (x_2, y_2)\} \). At this point, \((x_1, y_1)\) and \((x_2, y_2)\) represent the start and end points of an approximated straight line segment.

A line simplification algorithm removes the unnecessary information from the original boundary line to simplify the polygon [16]. In the line simplification algorithm, corners are expressed as boundary points.

In line simplification technique, it is need to have the standard about how does it represent the original boundary.

As shown in Fig. 2, when we attempt to simplify from the boundary list, \( E = \{(x_i, y_i) | i = 1 \ldots N\} \) consists of N to the line segment which is connected to both ends.
We will set the standard of error is mean square error from corner \((x_c, y_c)\) to line segment or we set \((\text{maximal error}) d_{\text{max}}\) the range of all distance. To determine a straight line when the \(d_{\text{max}}\) are within tolerance, \(T\), or without \(T\), the two boundary lists \(E_1 = \{ (x_i, y_i) | i=1…c \}\) and \(E_2 = \{ (x_i, y_i) | i=c…N \}\) are separated from the \((x_c, y_c)\) corner, after simplifying each boundary list that combines the subset list of two straight lines; a final Line Simplification can then be made. The line simplification algorithm is shown in Fig. 3. In this study, tolerance \(T\) equals 3 pixels for the distance.

### 2.2 Probability matching schema of rectangular shape using perpendicularity and parallel lines.

Each line segment detected by the polygonal approximation is expressed as \(L = (x_1, y_1, x_2, y_2)\). Using the end of the points \((x_1, y_1)\) and \((x_2, y_2)\) to calculate the direction \(\Theta\) and length \(l\) of the line segment, we can represent each line segment as \(L= (p_1, p_2, \Theta, l)\). At this point, \(p_1=(x_1, y_1)\) and \(p_2=(x_2, y_2)\). A rectangle is composed of four corners or four segments. The basic properties to be used for rectangle detection can be defined as the relation between the two line segments placed. Three characteristics of the rectangle to be used for rectangular detection are defined as, adjacency, verticality, and parallelism. Figure 4 represents the relationship between the two straight segments.

**A. Adjacency**

Adjacency is to show the adjacent that distance between the two line segments \(L_i\) and \(L_j\), when we called the intersection to \(v_p\), comparing \(v_p\) with the end point of two line segments then we can classify the close a pair of end point \((p_1, p_1)\) and far a pair of end point \((p_2, p_2)\).

\[
f_1(x, T) = \frac{1}{1+e^{-k(x-T)}}
\]

\[
f_2(x, T) = 1 - f_1(x, T)
\]

Two sigmoid functions are represented as shown in Fig. 5. At this point, \(k\) represents the slope; in this study, we use \(0.1 - 0.05\), assuming \(a_{\text{max}}=90\).

**B. Perpendicularity**

Perpendicularity shows the degree to which the two line segments intersect vertically. Therefore perpendicularity is dependent on each \(\Theta_i\) between the two segments, and the perpendicularity of the two segments is quantified with \(f_1(\Theta_i, T)\) when they meet in the corner and \(f_i(\Theta_i, T)\) when they meet on the opposite side.

**C. Parallelism**

Parallelism shows the degree to which the two line segments intersect.
have the same direction, and the degree to which the two line segments depend on each \( \theta_i \) between the two lines.

The parallelism of the two segments is quantified to \( f_2(\theta, T) \) and \( f_2(d', T) \) when they meet in the corner, \( f_2(d^{11}, T) \) and \( f_2(d^{21}, T) \) when they meet on the opposite side.

### D. Matching Schema

Schema is used to represent the overall scheme for the object showing the form of a crest which plays a role in the recognition scheme to describe the overall objective target existing outside. That is, a rectangular object that appears in various forms will be recognized through a combination of adjacency, perpendicularity, and the parallelism of the lines outside those rectangular objects. Four segments are given, that determine the adjacency, perpendicularity, and parallelism between the respective components to distinguish whether they are rectangular.

Four straight line segments constituting the rectangle cross each other to present a corner or an opposite side.

When the two line segments intersect with the corner, it should have the adjacency of \( f_1(d', T) \) perpendicularity while having \( f_1(d, T), d = |d^1 - d^2|/2 \) occurs if the opposite side has less adjacency with the parallelism of \( f_1(\theta, T) \). In addition, it must have \( f_1(d', T) \).

Using the adjacency, perpendicularity, and parallelism between the respective line segments forming the two rectangular schemes are defined that the matching schema which is meet in the corner and meet on the opposite side. In other word, the matching schema which is cross at right angle two straight line segments \( F \) defined

\[
F(i, j) = f_1(\theta, T_i) \times f_2(d', T_o) \quad i = 1...N, j = 1...N \tag{3}
\]

and the matching schema which is composed with opposite side \( F_g \) defined

\[
F_g(i, j) = f_2(\theta, T_i) \times f_1(d, T_o) \times f_2(d^{11}, T_o) \times f_2(d^{22}, T_o) \quad i = 1...N, j = 1...N \tag{4}
\]

In this study, the threshold is placed as \( T_0 = 60, T_{10} = 20 \) in the case of perpendicularity matching schema and is placed as \( T_{10} = 30, T_{101} = 30 \) in the case of opposite side matching schema.

This means that if an intersection of two straight lines \( L_i \) and \( L_j \) cross at the corner of a rectangular then the include angle is over than 60°, corner and straight lines \( L_i \) are in the less than the 30 pixels distance, when they exist opposite side, they have parallelism which of include angle degree is less than the 30°, the range deviation of the end of two straight line less than 30 pixels, the range of two straight line is far from the 50 pixels.

Figure 6 shows an example of the matching scheme. Figures 6(a) and 6(b) show the result of extracting the straight line segment, while 6(c) and 6(d) show formula (3) with the perpendicularity matched opposite side schema and the schema matching defined in (4).

As you see in Fig. 6(c), straight line 1, 3, 4 and 5 have probability to cross at right angles to straight line 7 and 8, it showed the high probability of being placed to opposite side straight line 1, 5 to line 3, 4, line 3, 4 to line 1, 5 and the line 7 to 8 in Fig. 6(d).

### 2.3 Detecting rectangular shape by hypothesis verification system

Four line segments \( L_0, L_1, L_4, \) and \( L_i \) are selected by using the previously determined matching schema, using adjacency, perpendicularity, and parallelism of the straight line segment when as previously mentioned. As seen from Fig. 7 the four selected linear segments are used to calculate the corners \( v_0, v_1, v_2, \) and \( v_3 \) which comprise the rectangle.

1) The least length of the 4 selected line segments must be given.
2) The sum of the 4 line segments must 50% more than the rectangle circumference consisting of the corners \( v_0, v_1, v_2, v_3 \).
3) The two end points of the line segment should stay the rectangular area.

Using the matching Schema and the condition of determination in order to detect the rectangular shape, the algorithm based on a hypothesis verification system is represented follows.

1) Select line segment \( L_0 \) randomly.
2) Select the line \( L_i' \) which has the highest probability to orthogonalize the selected line segment \( L_0 \) and corner \( p_i' \), select the line \( L_i' \) which has the highest probability to orthogonalize corner \( p_i' \).
3) If the 2 line segments selected, \( L_0, L_i \) do not satisfy the characteristics of the opposite side, return to the process of 1).
4) Select the line segment \( L_i \) and the highest probability line segment \( L_j \) which is made up of the opposite side.
5) If the selected line segment \( L_i \) and the two line segments \( L_j \),

![Fig. 6. Example of rectangular shape matching schema. (a) image of extracted linear segment, (b) information of extracted linear segment, (c) perpendicularity matching schema, and (d) opposite side matching schema.](image)

![Fig. 7. Rectangle matching condition.](image)


// Rectangle Detection
Algorithm RectangleDetection (L)
Input List of polylines, L = \{ (x_0,y_0), x_0) \}
Output List of rectangles, R = \{ (x_0,y_0) \}
1. Calculate the probability schema of vertices and parallel segments
   \[ F_i = \left\{ \begin{array}{l}
   \{ j \mid i = 1, j \geq 1, N \} \\
   \{ j \mid i = 1, N = j \}
   \end{array} \right. \]
2. for i = 1 to N do
   j = arg max (f(x,y) \mid x = 1, N)
   k = arg max (f(x,y) \mid x = 1, N, x \neq j)
   if F_i(j) \neq F_i(k) and F_j(j) \neq F_k(j) then
     l = arg max (f(x,y) \mid x = 1, N)
   if F_i(j) \neq l and F_k(l) then
     calculate 4-corner points as the parameter of rectangle
     add 4-corner parameter to rectangular shape list R
   3. return rectangular shape list, R

Fig. 8. Rectangular shape detection algorithm.

L do not have characteristics of perpendicularity to the corner on time, return to the process of 1).
6) Calculating the probability of the 4 selected line segments, L_i, L_j, L_k, and L_l match the rectangle; according to the probability, determine whether or not the line segments L_i, L_j, L_k, and L_l form a rectangle shape.

The hypothesis-rectangular shape detection algorithm based on the verification is explained in the supplementary explanation in Fig. 6. If a straight line 1 selected in Fig. 6(c), the first step is selected and the two straight lines, 8(66) and 7(62), are perpendicular, and straight lines 7 and 8 have the characteristics of (69) of the opposite side in Fig. 6(d). Furthermore, we can find straight line 4(54) which have opposite side character with straight line 1 in Fig. 6(c), we can confirm to the line 7(60) and 8(48 which is at right angle to straight line 4 from Fig. 6(c). Thus, a straight line 1-7-8-4 can be determined as a rectangular shape.

The algorithm is shown in Fig. 8.

3. RESULTS

In order to evaluate the performance of the proposed algorithm Intel® Core™ 2.93 GHz CPU with Windows-7 is used under the Environment and is attached to the 4 GB RAM and implemented in the algorithm in Visual C++. For the experiment, we used 276 sheets of images, which include a variety of rectangular shapes as a test image. The test image has rectangle sizes of 500×375, 1,624×1,288 pixels. The rectangular-shaped object contained in the test image has various sizes, directions, and the size of the rectangular shape is in a range of 85×60 ~ 360×120 pixels.

In order to measure the accuracy of the rectangle detection, the performance of the algorithm is evaluated by calculating the error rates, the FAR (false alarm rate), and the DR (detection rate). The DR and FAR error rates are defined as follows.

\[ DR = \frac{\text{no. of detected true rectangular object}}{\text{total no. of rectangular objects in image}} \]
\[ FAR = \frac{\text{no. of false alarms}}{\text{no. of detected true rectangular object + no. of false alarms}} \]

Figure 9 shows the test image to explain the algorithm and it has 607×701 pixels. Figure 9(a) shows the edge map, in which 128 segments were detected, and 9(b) shows that the segment is more than 20 pixel. The length of the line segment (LS=line segment) is detected from the edge map of 9(b). Figure 9(d) shows the three rectangles detected as indicated by the contour segment of the proposed algorithm which detects the rectangular shaped object.

Compared with the conventional rectangle detection algorithm shown in the literature [6,10] it was exhibited excellent performance.

Table 1 shows the execution time of the proposed algorithm for the 276 real images. The average overall processing time, as shown in Table 1 is 0.524 sec from the test images of size 500×375 and 1.939 sec in the test image of size 1,624×1,288.

Table 2 and Table 3 show the processing time for the proposed algorithm with respect to the actual image 276, MRF and RHT . The proposed algorithm, as shown in Table 2, shows a faster processing time than the RHT and MRF algorithm. Table 3 and Table 4 show the detection performance.

As shown in Table 3, for the algorithm, the detection error rate is lower than RHT, but is higher than MRF. This seems to occur because the multiple rectangles are detected in a rectangular shape by detecting a combination of the detected line segment,
Fig. 10. Example of false rectangle detection by line segments combination.

Table 3. Comparison of false alarm rate [%].

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>500 × 375</td>
<td>150</td>
<td>40.6±1.5</td>
<td>5.3 ±0.23</td>
<td>9.4±0.3</td>
</tr>
<tr>
<td>1,624 × 1,288</td>
<td>126</td>
<td>34.9±1.44</td>
<td>5.6 ±0.251</td>
<td>10.6±0.40</td>
</tr>
</tbody>
</table>

Table 4. Comparison of detection rate [%].

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>500 × 375</td>
<td>150</td>
<td>65.8±2.014</td>
<td>96.8±3.012</td>
<td>95.3±2.317</td>
</tr>
<tr>
<td>1,624 × 1,288</td>
<td>126</td>
<td>70.4±2.233</td>
<td>95.9±2.937</td>
<td>94.7±2.259</td>
</tr>
</tbody>
</table>

ACKNOWLEDGMENT

This work was supported by 2-Year Research Grant of Pusan National University.

REFERENCES