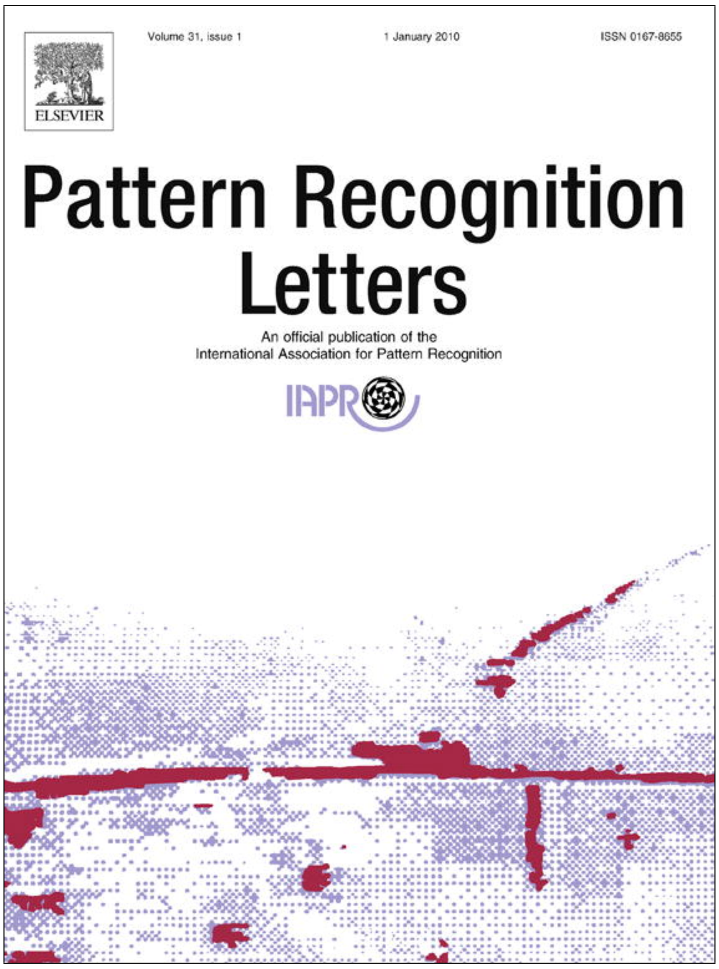
Volume 31, issue 1
1 January 2010
ISSN 0167-8655
ELSEVIER
Pattern Recognition Letters
An official publication of the International Association for Pattern Recognition
IAPR

Pattern Recognition Letters 31 (2010) 11–19

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

New orientation-based elimination approach for accurate line-detection

Kuo-Liang Chung [a,*], Zeng-Wei Lin [a], Shih-Ting Huang [a], Yong-Huai Huang [b], Hong-Yuan Mark Liao [c,1]

[a] *Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taipei 10672, Taiwan, ROC*
[b] *Institute of Computer and Communication Engineering, Jinwen University of Science and Technology, No. 99, An-Chung Road, Hsin-Tien, Taipei 23154, Taiwan, ROC*
[c] *Institute of Information Science, Academia Sinica, No. 128, Academia Road, Section 2, Taipei 115, Taiwan, ROC*

A R T I C L E I N F O

Article history:
Received 26 October 2008
Received in revised form 7 September 2009
Available online 13 September 2009

Communicated by G. Borgefors

Keywords:
Elimination strategy
Hough transform
Line-detection
Orientation-histogram
Randomized voting
Speed up

A B S T R A C T

Detecting lines correctly from a digital image is an important crucial step in many real-word applications. In this paper, we present an orientation-based strategy to filter out those inappropriate edge pixels before performing the line-detection task. Due to the effective strategy, both the memory size and the computation time are significantly reduced during a Hough transform-based detection process. Further, the proposed elimination strategy can also speed up the randomized-based detection process. Taking four previously developed line-detection techniques as comparison targets, experimental results have shown that our proposed orientation-based elimination strategy is superior to the previous line-detection methods in terms of memory requirement and computation time.

1. Introduction

Detecting lines in a digital image (Illingworth and Kittler, 1988; Leaver, 1993; Kälviäinen et al., 1995) is a very important and fundamental operation in image processing (Gonzalez and Wood, 2002), pattern recognition, machine vision (Hough, 1992; Jain et al., 1995; Davies, 2004), and computer vision (Forsyth and Ponce, 2002). The detected lines are very useful in many applications, such as document analysis (Jiang et al., 1997), autonomous vehicle navigation (Tsai et al., 2007; Lin et al., 2008), content-based image and video retrieval (Su et al., 2007), stereo matching (Long and Kanade, 1997), and so on.

Previously, many efficient deterministic algorithms have been presented (Hough, 1962; Duda and Hart, 1972; Illingworth and Kittler, 1987; Ben-Tzvi and Sandler, 1990; Aghajan and Kailath, 1994; Murakami and Naruse, 2000; Climer and Bhatia, 2003; Cha et al., 2006; Schindler, 2006; Chung et al., 2009). Among these deterministic algorithms, the Hough transform (HT) (Duda and Hart, 1972), called the DHT, is the most popular line-detection algorithm. However, it is memory- and time-consuming since it needs a 2D accumulator array to perform a voting process. In (Illingworth and Kittler, 1987) presented an adaptive HT scheme and arranged it in a coarse-to-fine manner to detect lines by using a smaller accumulator array. Murakami and Naruse (2000) proposed an efficient line-detection algorithm which could choose the region-of-interest to perform HT. Further, based on the line segments detected from an interested region, an extension process is applied to find more line segments. Ben-Tzvi and Sandler (1990) presented a combinatorial HT to speed up the voting process in the HT. Instead of considering all possible combinations of normal angles and distances, the combinatorial HT only checks all possible combinations of two-point line segments to reduce the computation time. Climer and Bhatia (2003) presented a local line-based HT which applies the slope mask with a small 3D accumulator array to each edge pixel for detecting lines passing through it. Cha et al. (2006) presented an extended HT (EHT) to detect line segments and the most distinctive advantage of the EHT is the ability to detect any line segment with desired length. Later, Chung et al. (2009) presented an advanced EHT to further improve the performance of the original EHT.

Besides the above deterministic HT-based algorithm, the randomized algorithm is another efficient line-detection paradigm (Fischler and Firschein, 1978; Fischler and Bolles, 1981; Xu et al., 1990; Xu and Oja, 1993; Kälviäinen and Hirvonen, 1997; Chutatape and Guo, 1999; Kyrki and Kälviäinen, 2000; Chen and Chung, 2001; Chung and Huang, 2007). (Fischler and Bolles, 1981; Fischler and Firschein, 1978) presented a random sample consensus technique for line-detection. (Xu et al., 1990; Xu and Oja, 1993) proposed a

* Corresponding author. Tel.: +886 227301081; fax: +886 227301080.
E-mail address: k.l.chung@mail.ntust.edu.tw (K.-L. Chung).
1 This work was completed when visiting Dept. of Computer Science and Information Engineering at National Taiwan University of Science and Technology from March 2009 to August 2009.

0167-8655/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2009.09.013

randomized HT (RHT). Kälviäinen and Hirvonen (1997) proposed a connective RHT (CRHT) which improves the RHT by exploiting the connectivity of local edge pixels. Based on the concept of seed point, Chutatape and Guo (1999) proposed a modified RHT (MRHT). Later, Kyrki and Kälviäinen (2000) proposed an extension version of CRHT (ECRHT). Based on the evidence-based sampling technique, Chen and Chung (2001) later proposed a faster randomized algorithm for detecting lines, called the RLD. Recently, Chung and Huang (2007) presented a look-up table-based (LUT-based) approach to speed up the RLD. The above randomized line-detection approach can save memory as well as computation time.

This paper presents an orientation-based approach that can improve simultaneously the HT-based algorithm and the randomized algorithm. The proposed orientation-based approach can classify edge pixels into several sets and each set is composted of edge pixels with similar gradient orientations. Thus, for an HT-based algorithm, the voting process only requires a smaller 2D accumulator array to record the gradient orientations of related edge pixels. Besides memory-saving, a smaller accumulator array indicates that the computation time is also significantly reduced since much less amount of quantized normal angles are considered in the voting process. For a randomized approach, since the edge pixels contained in an edge-pixel set have similar gradient orientations, it is very likely that the chosen edge pixels are colinear. When dealing with colinear edge pixels, the time required in the voting process can be reduced and it would save the computation time significantly. Thus, a randomized algorithm would have a good speed up when executing the proposed orientation-based strategy. To conduct a fair performance evaluation process, the computation time consumed in the orientation-histogram construction process and the inappropriate edge pixels elimination process is included in the total execution-time. For comparison purpose, we chose the DHT (Duda and Hart, 1972), the EHT (Cha et al., 2006), the RLD with LUT (Chung and Huang, 2007), called RLDL, and the MRHT (Chutatape and Guo, 1999) as the representatives in our experiments. Experimental results have justified the above advantages of our proposed orientation-based elimination strategy for the DHT, the EHT, the RLDL, and the MRHT.

The remainder of this paper is organized as follows. In Section 2, our proposed orientation–elimination strategy is presented. In Section 3, following the orientation–elimination strategy, the line-detection algorithms of four existing ones are presented. In Section 4, experiments are conducted to show the advantages of our algorithms. Concluding remarks are drawn in Section 5.

2. The orientation–elimination strategy

Given an image f, the edge map can be obtained by using the Canny edge detector (Canny, 1986). For the edge pixel at position (x, y), the gradient orientation $AG(x, y)$ can be computed by

$$AG(x,y) = \tan^{-1}\frac{\nabla_y f(x,y)}{\nabla_x f(x,y)}, \tag{1}$$

where $f(x, y)$ denotes the image pixel at position (x, y); $\nabla_x f(x, y)$ and $\nabla_y f(x, y)$ are the gradients obtained by running the two masks shown in Fig. 1a and b on $f(x, y)$, respectively.

Based on the obtained edge pixels, we can construct a K-level orientation-histogram as follows:

-1	1
-1	1

-1	-1
1	1

Fig. 1. Two kinds of masks for computing gradients. (a) X-directional mask. (b) Y-directional mask.

$$s(x,y) = \begin{cases} \text{round}\left(\frac{K\times AG(x,y)}{\pi}\right) \bmod K, & \text{if } 0 \leqslant AG(x,y) < \frac{\pi}{2}, \\ \text{round}\left(\frac{K\times (AG(x,y)+\pi)}{\pi}\right) \bmod K, & \text{if } -\frac{\pi}{2} \leqslant AG(x,y) < 0, \end{cases} \tag{2}$$

where for each edge pixel $f(x, y)$ with orientation $AG(x, y)$, $s(x, y)$ denotes the index of the assigned angle slot in the K-level orientation-histogram. For example, let $K = 18$, then the 18-level orientation-histogram has 18 quantized angle slots, say $\overline{\theta_0}, \overline{\theta_1}, \ldots$ and $\overline{\theta_{17}}$. Here, $\overline{\theta_i}$ can be treated as a set of edge pixels whose AG values satisfy $(10 \times i - 5) < \text{AG} \leqslant (10 \times i + 5)$. Fig. 2a illustrates the input gray image and Fig. 2b depicts the resultant edge map using the Canny edge detector. When running the Canny edge detector on an input image, the above two rules indicate that the orientations of all edge pixels can be obtained simultaneously. Fig. 3 depicts that the 18-level orientation-histogram of all the determined edge pixels for Fig. 2a. For fairness, the experiments in Section 4 include the extra time requirement for computing all the orientations and for constructing the orientation-histogram.

At the end of this section, based on the K-level constructed orientation-histogram, we present an orientation–elimination strat-

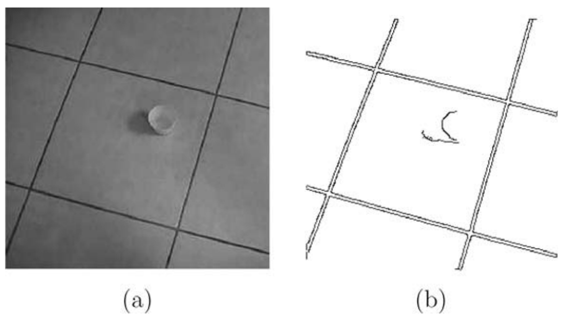

Fig. 2. The input gray image and its corresponding edge map. (a) The input gray image. (b) The obtained edge map by using the Canny edge detector.

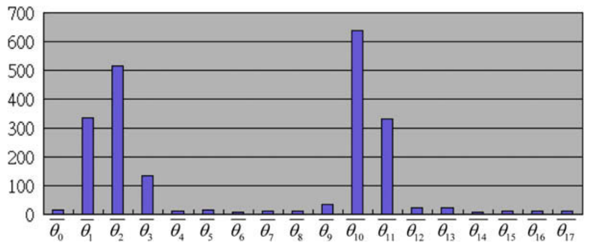

Fig. 3. The orientation-histogram of all the determined edges for Fig. 2a.

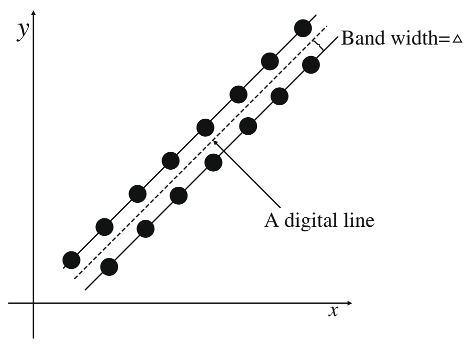

Fig. 4. A digital true line with bandwidth Δ.

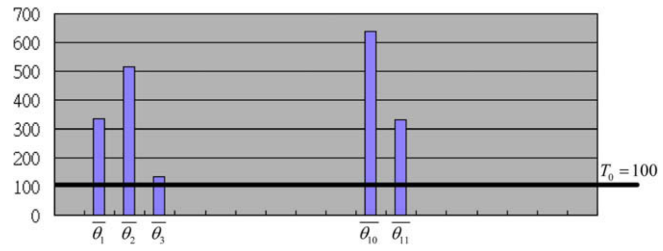

Fig. 5. The simplified orientation-histogram of Fig. 3 after discarding inappropriate edge pixels for $T_o = 100$.

egy to help us to discard those inappropriate edge pixels in order to speed up the line-detection process.

Since the number of edge pixels along a digital true line with bandwidth Δ (see Fig. 4) should be larger than a pre-determined threshold T_o, we can utilize the threshold T_o as a filter to discard those inappropriate edge pixels which are in the same quantized slot. The frequency of those discarded inappropriate edge pixels should be less than T_o. According to this orientation–elimination strategy, the orientation-histogram is transformed into a simplified version of orientation-histogram after filtering out those inappropriate edge pixels. For example, Fig. 5 is the simplified orientation-histogram of Fig. 3. Comparing the orientation-histogram in Fig. 3 and the simplified orientation-histogram in Fig. 5, as a filter, the threshold T_o (=100) does discard those inappropriate edge pixels since they make no contribution to true lines. Instead of considering all the eighteen quantized angle slots in Fig. 3, $\overline{\theta_0}, \overline{\theta_1}, \ldots$, and $\overline{\theta_{17}}$, we only consider five quantized angle slots in Fig. 5, $\overline{\theta_1}$, $\overline{\theta_2}, \overline{\theta_3}, \overline{\theta_{10}}$, and $\overline{\theta_{11}}$. This selection leads to a significant computation-saving effect. Since those valid edges may sit along several different lines, a voting process is still needed to examine which lines are true lines.

3. Four improved line-detection algorithms using the orientation–elimination strategy

In this section, our proposed orientation–elimination strategy will be used to speed up several existing line-detection algorithms, including the DHT, the EHT, the RLDL, and the MRHT. For convenience, the four improved line-detection algorithms are called the IDHT, the IEHT, the IRLDL, and the IMRHT, respectively.

3.1. Improvement of HT-based line-detection methods

3.1.1. The improved DHT: IDHT

In this subsection, the proposed orientation-based elimination strategy cooperates with the DHT, called the IDHT, is introduced. This improvement can speed up the DHT and reduce its memory requirement. In the DHT, a straight line in an image is parameterized in normal form by $\rho = x\cos\theta + y\sin\theta$ where (x, y) denotes the coordinates of the edge pixels on a straight line; ρ and θ denote the normal distance from the origin to the line and the angle between the normal line and positive x-axis. Essentially, the DHT consists of two processes: (1) the voting process in the accumulator array and (2) the searching process for finding the desired line in the accumulator array. After finishing the voting process for all edge pixels, the searching process selects the cells whose scores are larger than a specified threshold in the accumulator array, and then their corresponding parameters are recognized as the parameters of the desired lines in an image. Fig. 6 depicts the used accumulator array in the DHT for $-\sqrt{2}N \leqslant \rho \leqslant \sqrt{2}N$ and $0 \leqslant \theta < 180$; N denotes the longer length of the height and the width of an input image.

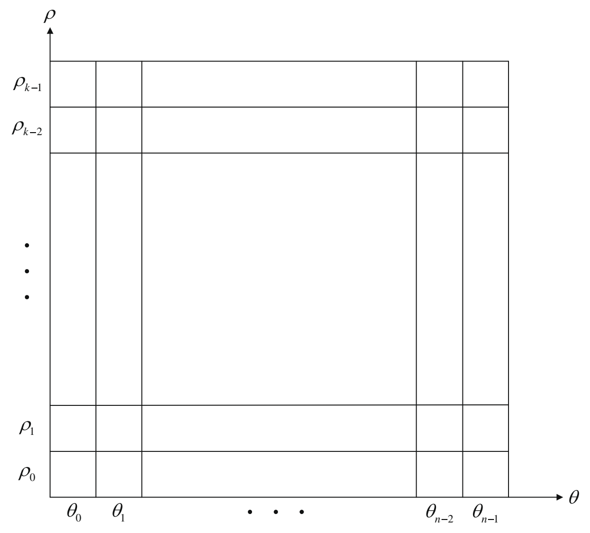

Fig. 6. The accumulation array used in the DHT.

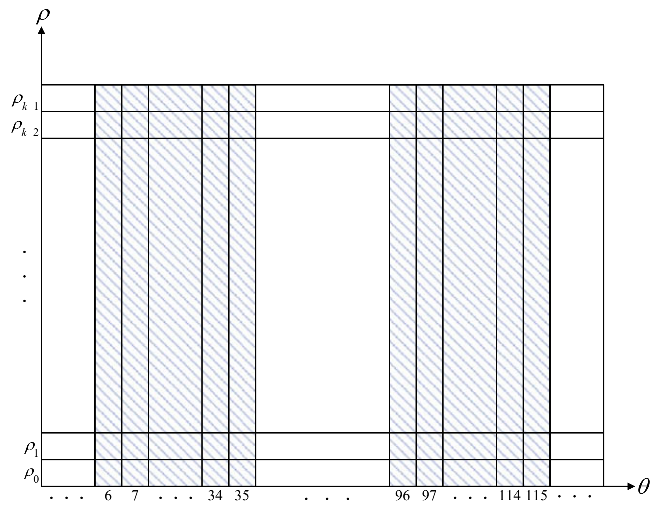

Fig. 7. The hatched area denotes the considered normal angles.

However, the memory space required in the DHT is dependent on the number of quantized angles and quantized normal distances. Based on our proposed orientation-based elimination strategy, we now present the IDHT to reduce the memory space and the time required in the DHT.

Taking Fig. 2a as an example, we can obtain the corresponding orientation-histogram as shown in Fig. 5. In Fig. 5, there are five angle slots. Each of them contains more than $T_o(= 100)$ edge pixels. Consequently, our proposed IDHT only focuses on these five angle slots. On the other hand, the normal angles considered in the accumulation array in the proposed IDHT are limited within 6°—15°, 16°—25°, 26°—35°, 96°—105° and 106°—115°. The hatched area in Fig. 7 denotes the considered normal angles which we should consider in the voting process. Here the basis unit of quantized angle and that of quantized normal distance are set to one degree and 1, respectively. Each time only those edge pixels in the same angle slot are considered in the line-detection process. In the proposed IDHT, it only needs a small accumulator array to handle the voting process for all angle slots. For the ith angle slot, the horizontal coordinates of the used small accumulator array are ranged from $(10 \times i - 5)$ to $(10 \times i + 5)$; the vertical coordinates are ranged from $-\sqrt{2}N$ to $\sqrt{2}N$. Thus, the ratio of the voting space in the proposed IDHT over the DHT is $5.6\% \left(= \frac{(10\times i+5)-(10\times i-5))}{180}\right)$. From the small accumulator array, the IDHT has great improvement on theoretical execution-time of the IDHT over the DHT. The improvement is $94.4\% (= \frac{1-5.6\%}{1})$. In Section 4, the experimental results show that the average execution-time improvement ratio of the proposed IDHT over the DHT is 94.2% and it is close to the theoretical execution-time improvement ratio. Our proposed IDHT is listed below.

Step 1. Let the angle slot $\overline{\theta_i}$, $0 \leqslant i \leqslant 17$, contain the set of edge pixels, $V_i = \{v_k = ((x_k, y_k), O_k) | \text{ for } 1 \leqslant k \leqslant |V_i| \text{ where } (x_k, y_k)$ and O_k denote the location and the orientation of the edge pixel $v_k\}$. According to Eqs. (1) and (2), we put the coordinates of all edge pixels into the corresponding angle slots. Let T_l be a given threshold which denotes the number of a true line should have at least T_l edge pixels. Initialize $i = 0$, and go to Step 2.

Step 2. If $|V_i| \geqslant T_o$, where T_o is the least number of edge pixels that the angle slot θ_i should have, go to Step 3; otherwise, go to Step 5.

Step 3. We set up the small accumulation array, A, such that the horizontal coordinates are ranged from $(10 \times i - 5)$ to $(10 \times i + 5)$ and the vertical coordinates are ranged from $-\sqrt{2}N$ to $\sqrt{2}N$. For each edge pixel $v_k = (x_k, y_k)$ whose orientation is within the angle interval $(10 \times i - 5, 10 \times i + 5)$, we compute the normal distance by $\rho_k = x_k \cos\theta + y_k \sin\theta$ for all quantized angles in the angle interval and then we perform the voting operation $A[\theta][\rho_k] = A[\theta][\rho_k] + 1$. Go to Step 4.

Step 4. Check all entries of the array A, if $A[\theta][\rho] \geqslant T_l$, it infers that there is a true line L_c which satisfies $\rho = x\cos\theta + y\sin\theta$, where (x, y) is on L_c. Go to Step 5.

Step 5. Perform $i = i + 1$. Check i whether it is larger than 17 or not. If yes, we stop the algorithm; otherwise, go to Step 2.

3.1.2. The improved EHT: IEHT

In this subsection, we put the proposed orientation–elimination strategy into the previous EHT, called the IEHT, to improve the performance of previous EHT. The most distinctive advantage of the EHT method is the ability to detect any line segment with designated length. Further, it also can detect the starting point and ending point of each detected line segment. For each edge pixel, the voting process is performed in two 3D Hough spaces. A 3D Hough space consists of a set of 2D Hough planes. The voting process is able to detect line segments. In the EHT method, each 2D Hough plane is used to collect the evidence of the line segments which is passing through a specific column of the input image. Considering the first column of the edge map, the edge pixel is selected one by one in top–down manner. For each selected edge pixel with location (x, y), $0 \leqslant x \leqslant w - 1$ and $0 \leqslant y \leqslant h - 1$ where w and h denote the width and height of the input image, a slope–intercept equation $\beta = -\alpha x + y$ is created, and then the voting process is performed in the first Hough plane of the first Hough space for $-1 < \alpha \leqslant 1$ corresponding to the angle range $(-45°, 45°]$; after rotating the x and y axes by 90°, i.e. mapping the edge pixels with location (x, y) into the one with location $(-y, x)$, the voting process is performed in the first Hough plane of the second Hough space. Note that each edge pixel in the first column of the edge map must perform two voting processes for the first and second Hough spaces. Based on the same concept, for the second column of the edge map, the above two voting processes are applied to the second Hough plane of the first Hough space and the second Hough space, and so on. After finishing the voting processes for the last column of the edge map, we sum up the votes of each pair of α and β through all Hough planes of each Hough space. For each pair of α and β, if the number of total votes is larger than a pre-defined threshold, it can be claimed that the input image has a line segment with the parameters α and β and its starting point and ending point can be determined by checking the number of votes of each Hough planes.

Based on the constructed orientation-histogram mentioned above, the proposed orientation-based elimination strategy can still be used to discard those inappropriate edge pixels originally considered in the EHT. For the ith angle slot, if the value of $10 \times i - 45$ is within the range $(-45°, 45°]$, the first Hough space is used to perform the voting process; otherwise, the voting process is performed in the second Hough space. Since for each angle slot, only the first Hough space or the second Hough space is used to performed the voting process, we can use only one accumulator array to implement the proposed IEHT. Thus, the proposed IEHT only requires 50% memory space when compared to the EHT. Further, the IEHT also reduces about 50% voting time of the EHT, thus the theoretical voting time improvement ratio of the IEHT over the EHT is 50%. From our experiments, we find that the ratio of the average voting time over the total execution-time is about 50%, so the theoretical execution-time improvement ratio of the IEHT over the EHT is about $50\% \times 50\% = 25\%$. In Section 4, the experimental results show that the average execution-time improvement ratio of the proposed IEHT over the EHT is 27.6% and it is close to the theoretical execution-time improvement ratio.

3.2. Improve the previous randomized line-detection methods

3.2.1. The improved RLDL: IRLDL

In this subsection, based on the proposed orientation-based elimination strategy, we present an improved randomized line-detection algorithm, the IRLDL.

In the RLDL, it randomly selects three edge pixels each time to check whether they can construct a possible line or not. Let $v_1 = (x_1, y_1)$, $v_2 = (x_2, y_2)$, and $v_3 = (x_3, y_3)$ denote the three selected edge pixels. For the edge pixels v_1 and v_2, a line L_{12} can be determined and then we calculate the distance $d_{3\to i12}$ between v_3 and L_{12}. If the distance $d_{3\to i12}$ is less than the specified threshold T_2 where T_2 is set to 1 empirically, L_{12} is a possible line and then a voting process based on the LUT-based platform is performed to determine whether L_{12} is a true line or not. According to v_1 and v_2, Bresenham's line drawing procedure (Hearn and Baker, 1997) is used to draw L_{12} on the 2D binary array where bit '1' in the entry denotes the corresponding pixel in the original edge map passing through the banded possible line for $\Delta = 1$. The 2D binary array A_{lut} associated with the drawn banded possible line is called the LUT-based platform. From A_{lut}, each input edge pixel $v_k = (x_k, y_k)$ is used as the key to perform the query on the LUT-based platform A_{lut}. If $\mathrm{A}_{lut}(x_k, y_k) = 1$, it means that the edge pixel v_k is on L_{12} and it contributes a vote to the possible line. The total number of edge pixels contributing to L_{12} is used to determine whether the possible line is a true line or not. If L_{12} is a true line, all edge pixels lying on it are taken out from the edge pixel set and the RLDL selects next three edge pixels to detect the remaining lines in the image. In the RLDL, if the selected three edge pixels cannot help us to detect a true line, it is called a failed selection. If the number of successive failed selections is larger than the threshold T_f, it indicates that all true lines in the input image have been detected, and then the RLDL can be stopped.

In the RLDL, it's time-consuming to handle the redundant voting processes for those failed selections. Thus, the total voting time of the RLDL is dominated by the threshold T_f. We now present an improved RLDL, called the IRLDL, to solve the above problem. In our proposed IRLDL, the edge pixels belonging to the same angle slot are stored in the same edge-pixel set. A possible line is determined by three edge pixels selected from the same angle slot since they should have similar gradient angles. Under these circumstances, we are able to discard the considerations of inappropriate possible lines. On the other hand, for each feasible possible line, the voting process is performed only on the corresponded angle slot and its two neighboring angle slots. Since the inappropriate possible lines are all discarded, we can set a smaller T_f for each angle slot.

Let the ratio P_i denote the number of edge pixels in the ith angle slot over that of total edge pixels and naturally it is defined by

$$P_i = \begin{cases} \frac{|V_i|}{\sum_{j=0}^{17}|V_j|} & |V_i| > T_o, \\ 0 & \text{otherwise}, \end{cases} \tag{3}$$

where $|V_i|$ denotes the number of edge pixels in the ith angle slot and T_o has been defined in Section 2. According to the definition of P_i, naturally the threshold of successive failed selections for the ith angle slot can be determined by $T_f^i = T_f \times P_i$. From the determined threshold T_f^i, the ith angle slot needs to perform the voting process T_f^i times at most; in each voting process, at most $|V_i|$ edge pixels are required to query on the LUT-based platform A_{LUT}. For simplifying the analysis, the voting time for the ith angle slot is linearly proportional to $T_f^i \times |V_i|$ and the theoretical voting time improvement ratio of the proposed IRLDL over that of the RLDL is $\frac{T_f \times |V| - \sum_{i=0}^{17}(T_f^i \times |V_i|)}{T_f \times |V|} = \frac{T_f - \sum_{i=0}^{17} T_f^i \times P_i}{T_f}$. From our experiments, we find that the ratio of the average voting time over the total execution-time is 60%, and the theoretical execution-time improvement ratio of the proposed IRLDL over that of the RLDL is $\frac{T_f - \sum_{i=0}^{17} T_f^i \times P_i}{T_f} \times 60\%$. For example, from Figs. 2, 3, and 5, the number of edge pixels in Fig. 2b is 2187, and $|V_1|$, $|V_2|$, $|V_3|$, $|V_{10}|$, and $|V_{11}|$ are 331, 513, 132, 628, and 315, respectively. The values of T_f^1, T_f^2, T_f^3, T_f^{10}, and T_f^{11} are $0.15T_f$, $0.23T_f$, $0.06T_f$, $0.29T_f$, and $0.14T_f$, respectively. Therefore, for Fig. 2a, the theoretical execution-time improvement ratio of the proposed IRLDL over that of the RLDL is $49\%\left(= \frac{T_f - (0.15 \times T_f^1 + 0.23 \times T_f^2 + 0.06 \times T_f^3 + 0.29 \times T_f^{10} + 0.14 \times T_f^{11})}{T_f} \times 60\%\right)$. In Section 4, experimental results show that the execution-time improvement ratio of the proposed IRLDL over that of the RLDL is 52.0% for Fig. 2a (see Table 3) and it is close to the theoretical execution-time improvement ratio. Note that the voting space required in the IRLDL is the same as that in the RLDL since the accumulator array is unnecessary for the RLDL. Our proposed IRLDL is shown below.

Step 1. Let the angle slot $\overline{\theta_i}$, $0 \leqslant i \leqslant 17$, contain the set of the edge pixels $V_i = \{v_k = ((x_k, y_k), O_k) |$ for $1 \leqslant k \leqslant |V_i|$ where (x_k, y_k) and O_k denote the location and the orientation of the edge pixel $v_k\}$. According to Eqs. (1) and (2), we put the coordinates of all edge pixels into the corresponding angle slots. Let T_f be a given threshold to denote the number of failures that we can tolerate. The two thresholds, T_o and T_l, have been defined in Steps 1 and 2 of the IDHT. Initialize $i = 0$, and go to Step 2.

Step 2. If $|V_i| \geqslant T_o$, set $T_f^i = T_f \times \frac{|V_i|}{\sum_{j=0}^{17}|V_j|}$ and let the failure counter be $f = 0$; otherwise, go to Step 8.

Step 3. If $f = T_f^i$ or $|V_i| < T_l$, go to Step 8; otherwise, we randomly select three edge pixels v_k, $k = 1, 2, 3$, from V_i. When v_k has been taken away from the set V_i, $1 \leqslant k \leqslant 3$, set $V_i = V_i - \{v_k\}$.

Step 4. Based on the three selected points v_k for $k = 1, 2, 3$, determine the line L_{12} and check the following two conditions.

Condition 1: The values of $|AG(x_1, y_1) - AG(x_2, y_2)|$, $|\Theta + \frac{\pi}{2} - AG(x_1, y_1)|$, and $|\Theta + \frac{\pi}{2} - AG(x_2, y_2)|$ are all less than the specified threshold T_1 where $\Theta = \tan^{-1}\frac{(y_1 - y_2)}{(x_1 - x_2)}$ is used to represent the angle between L_{12} and x-axis and T_1 is set to $\frac{2\pi}{180}$ empirically.

Condition 2: The distance between L_{12} and v_3 is less than the specified threshold T_2 where T_2 is set to 1 empirically.If the above two conditions are satisfied, L_{12} is a possible line and go to Step 5; otherwise, put v_k for $k = 1, 2, 3$, back to V_i, and perform $f = f + 1$; go to Step 3.

Step 5. We initialize each entry of the voting platform A_{lut}. Set the counter $C = 0$. We then set up the possible line on the LUT-based platform A_{lut}. For each element v_k in $V_i \bigcup V_{i-1} \bigcup V_{i+1}$, we use the location of v_k, (x_k, y_k), as the key to vote on the array A_{lut}. If $A_{lut}(x_k, y_k) = 1$, perform $C = C + 1$ and take v_k out of V_i.

Step 6. Assume there are lp edge pixels on the possible line, i.e. $C = lp$. If $C > T_l$, go to Step 7; otherwise, we regard the possible line be a false line and return these lp edge pixels to V_i; $f = f + 1$, and go to Step 3.

Step 7. The possible line has been determined as a true line. Go to Step 3.

Step 8. Perform $i = i + 1$. Check i whether it is larger than 17 or not. If yes, we stop the algorithm; otherwise, go to Step 2.

3.2.2. The improved MRHT: IMRHT

In this subsection, we put the proposed orientation-based elimination strategy into the previous MRHT, called IMRHT, to speed up the MRHT. In the previous MRHT, a straight line in an image is parameterized in normal form by $\rho = x\cos\theta + y\sin\theta$ that has been described in Section 3.1.1. From the set of n edge pixels $\{(x_i, y_i) | i = 1, 2, \ldots, n\}$ in the input image, the MRHT randomly picks one edge pixel (x_m, y_m) as the seed and then the voting process is performed in the accumulator array $A(\theta)$, $0 \leqslant \theta < 180$, to detect the straight lines passing through the seed point. By pairing each remaining edge pixel (x_j, y_j) with the seed point (x_m, y_m), a straight line can be determined and the normal angle θ can be solved by

$$\theta = \tan^{-1}\left(\frac{x_j - x_m}{y_m - y_j}\right). \tag{4}$$

From the obtained θ, we increase the value of $A(\theta)$ by one. The voting process for the specified seed point (x_m, y_m) continues until each pair of points $\{(x_m, y_m), (x_j, y_j)|$ for $j = 1, 2, \ldots n$ and $j \neq m\}$ is processed. After all procedures for the specified seed point (x_m, y_m) have been finished, the votes in the $A(\theta)$ can be further detected. If the votes of one cell $A(\theta_m)$ is larger than the predefined global threshold, the detected straight line is determined by (x_m, y_m) and θ_m. By the same argument, the new seed point is selected randomly to perform the above voting process.

Based on the constructed orientation-histogram mentioned above, the pair of a specified seed point (x_m, y_m) and one of the remaining edge pixel (x_j, y_j) are selected from the same angle slot since they have similar gradient angles. Thus, the MRHT can be improved by the proposed orientation-based strategy. In what follows, we use Fig. 2a as an example to analyze the voting time improvement advantage of the IMRHT. Fig. 2a contains four lines, so each edge pixel may be considered at most four times in the four voting processes for detecting four lines. In the proposed IMRHT, the four lines can be classified into two groups and each group includes two lines with similar orientations. Thus, each angle slot contains at most two lines and each pixel is involved in the voting processes of two lines at most in the IMRHT. Therefore, the theoretical voting time improvement ratio of IMRHT over the MRHT is $50\%(= \frac{4-2}{4})$ for Fig. 2a. Since the time required in the voting process of the MRHT is almost equal to the total execution-time, the theoretical execution-time improvement is also 50% for Fig. 2a. In Section 4, experimental results show that the execution-time improvement ratio of the proposed IMRHT over that of the MRHT is 55% for Fig. 2a (see Table 3) and it is close to the theoretical execution-time improvement ratio. Note that the voting space required in the IMRHT is the same as the that in the MRHT since the MRHT only uses a small accumulator array to deal with the voting process of each seed point.

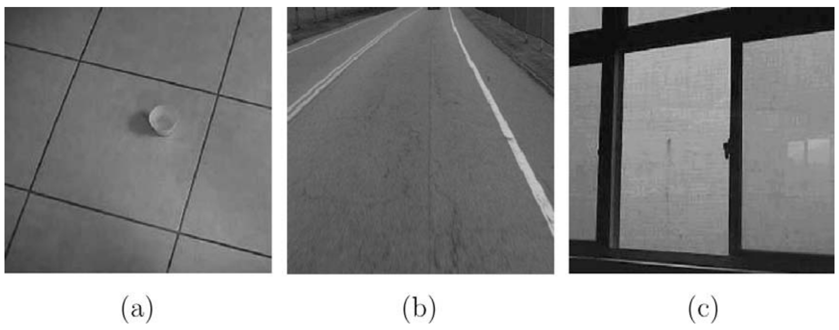
Fig. 8. Three testing images. (a) Floor image (b) Road image. (c) Window image.

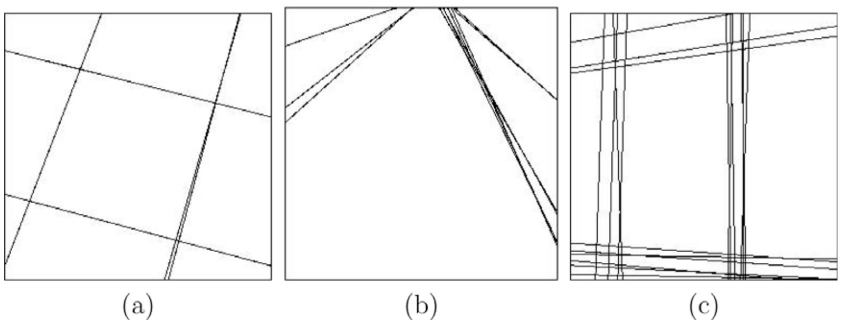
Fig. 9. The resultant detected lines by using the DHT. (a) Detected lines for floor image. (b) Detected lines for road image. (c) Detected lines for window image.

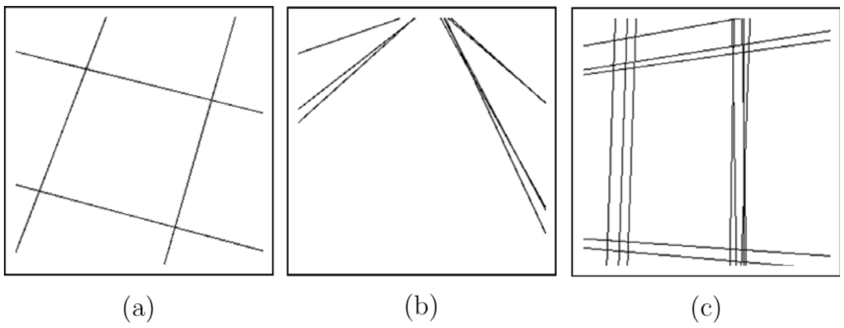
Fig. 10. The resultant detected lines by using the IDHT. (a) Detected lines for floor image. (b) Detected lines for road image. (c) Detected lines for window image.

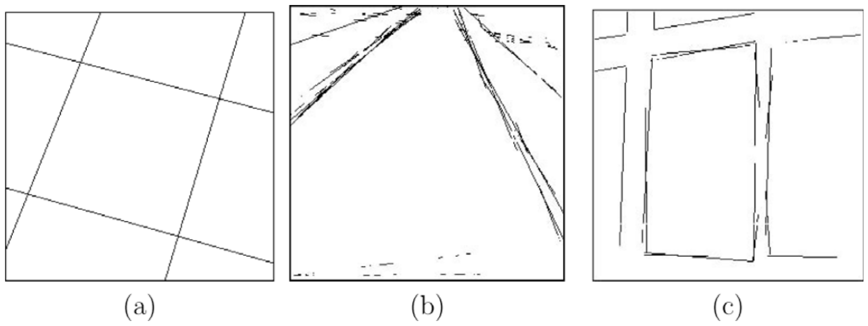
Fig. 11. The resultant detected lines by using the EHT. (a) Detected lines for floor image. (b) Detected lines for road image. (c) Detected lines for window image.

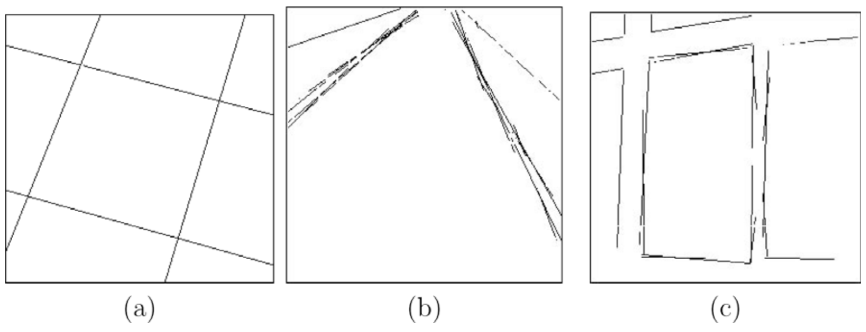

Fig. 12. The resultant detected lines by using the IEHT. (a) Detected lines for floor image. (b) Detected lines for road image. (c) Detected lines for window image.

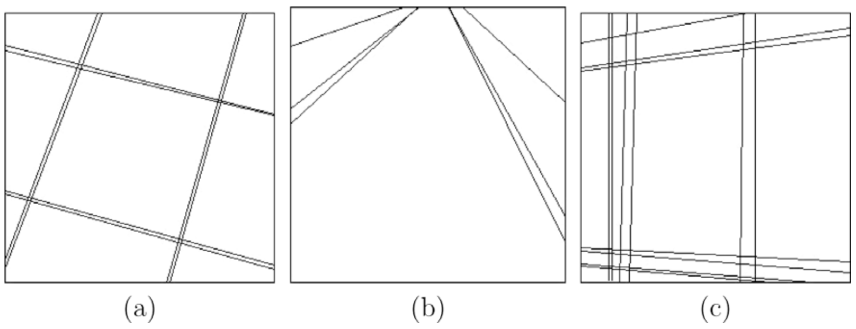

Fig. 13. The resultant detected lines by using the RLDL. (a) Detected lines for floor image. (b) Detected lines for road image. (c) Detected lines for window image.

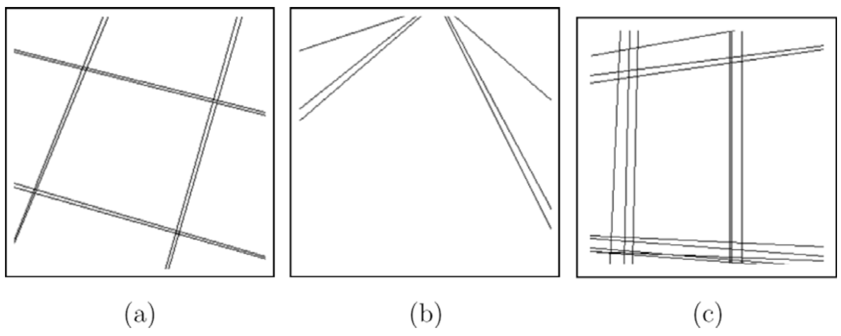

Fig. 14. The resultant images by using the IRLDL. (a) Detected lines for floor image. (b) Detected lines for road image. (c) Detected lines for window image.

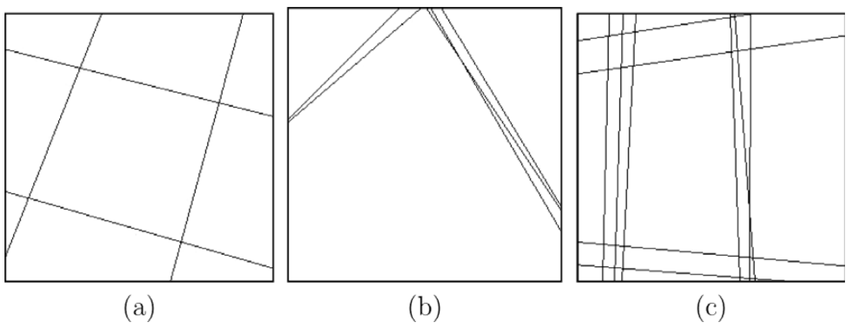

Fig. 15. The resultant detected lines by using the MRHT. (a) Detected lines for floor image. (b) Detected lines for road image. (c) Detected lines for window image.

4. Experimental results

In this section, we show the performance improvement after employing our proposed orientation-based elimination strategy into the previous four line-detection methods. For fairness, the experiments include the extra time requirement for computing all the orientations and for constructing the orientation-histogram. Experimental results have justified the advantages of our proposed orientation-based elimination strategy. The concerned algorithms are implemented on the IBM compatible computer with Pentium

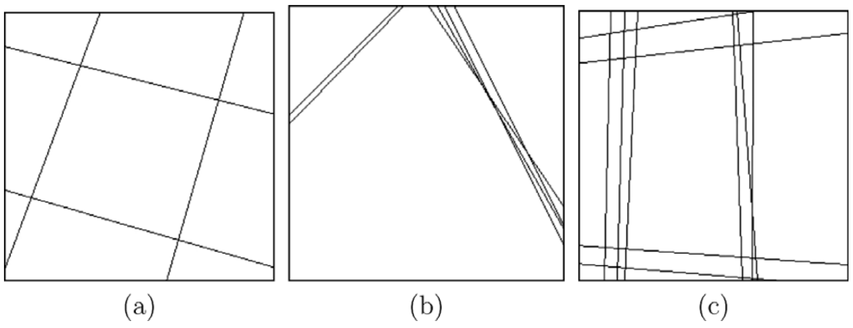

Fig. 16. The resultant detected lines by using the IMRHT. (a) Detected lines for floor image. (b) Detected road lines for image. (c) Detected lines for window image.

IV CPU 3.2GHz and 1GB RAM. The operating system used is MS-Windows XP and the program developing environment is Borland C++ Builder 6.0. Fig. 8 illustrates the used three testing images where each image is of size 256×256.

After running the four algorithms on the images shown in Fig. 8, the resultant detected lines of the DHT, the IDHT, the EHT, the IEHT, the RLDL, the IRLDL, the MRHT, and the IMRHT are shown in Figs. 9–16, respectively. In our experiments, the number of angle slots K is set to 18 and the thresholds T_o, T_l, and T_f are set to 100, 100, and 10000, respectively. The ratio of the memory required in the IDHT (IEHT) over the DHT (EHT) is 5.6% (50%). The memory required in the IRLDL (IMRHT) is the same as that in the RLDL (MRHT). The average execution-time improvement ratios of the proposed improved line-detection methods over the previous methods are listed in Tables 1–4 where the symbol 'ms' denotes milliseconds. From the four tables, it is observed that the execution-time improvement ratios of our proposed four improved methods, the IDHT, the IEHT, the IRLDL, and the IMRHT, over the previous methods are 94.2%, 27.6%, 58%, and 60.8%, respectively.

Table 1
Execution-time improvement ratio of the proposed IDHT over the DHT.

	Fig. 8a	Fig. 8b	Fig. 8c
DHT (ms)	84.53	63.44	79.22
Proposed IDHT (ms)	4.85	3.59	4.84
$\frac{DHT-IDHT}{DHT}$	94.3%	94.3%	93.9%
Average improvement ratio		94.2%	

Table 2
Execution-time improvement ratio of the proposed IEHT over the EHT.

	Fig. 8a	Fig. 8b	Fig. 8c
EHT (ms)	29.7	42.2	90.7
Proposed IEHT (ms)	16.45	31.63	78.63
$\frac{EHT-IEHT}{EHT}$	44.6%	25%	13.3%
Average improvement ratio		27.6%	

Table 3
Execution-time improvement ratio of and the proposed IRLDL over the RLDL.

	Fig. 8a	Fig. 8b	Fig. 8c
RLDL (ms)	7.782	8.703	9.922
Proposed IRLDL (ms)	3.735	2.078	5.375
$\frac{RLDL-IRLDL}{RLDL}$	52.0%	76.1%	45.8%
Average improvement ratio		58%	

Table 4
Execution-time improvement ratio of the proposed IMRHT over the IMRHT.

	Fig. 8a	Fig. 8b	Fig. 8c
MRHT (ms)	9.3	106.2	36
Proposed IMRHT (ms)	4.17	50.62	9.06
$\frac{MRHT-IMRHT}{MRHT}$	55.2%	52.3%	74.8%
Average improvement ratio		60.8%	

5. Conclusions

We have proposed a new orientation-based elimination strategy for line-detection. Based on the constructed orientation-histogram, the proposed orientation-based elimination strategy can be used to discard those inappropriate edge pixels to improve the performance of four existing methods. Based on three test images, experimental results demonstrate that the execution-time improvement ratios of our methods, the IDHT, the IEHT, the IRLDL, and the IMRHT, over the four existing methods are 94.2%, 27.6%, 58%, and 60.8%, respectively. Further, the ratio of the memory required in the IDHT (IEHT) over the DHT (EHT) is 5.6% (50%) and the memory required in the IRLDL (IMRHT) is the same as that in the RLDL (MRHT). Our orientation elimination-based strategy has been applied to speed up the computation of line-detection successfully.

Acknowledgement

Supported in part by the National Science Council of ROC under Contract NSC98-2923-E-011-001-MY3.

References

Aghajan, H.K., Kailath, T., 1994. Slide: Subspace-based line detection. IEEE Trans. Pattern Anal. Machine Intell. 16, 1057–1073.
Ben-Tzvi, D., Sandler, M.B., 1990. A combinatorial hough transform. Pattern Recognition Lett. 11, 167–174.
Canny, J.F., 1986. A computational approach to edge detection. IEEE Trans. Pattern Anal. Machine Intell. 8, 679–698.
Cha, J., Cofer, R.H., Kozaitis, S.P., 2006. Extended hough transform for linear feature detection. Pattern Recognit. 39, 1034–1043.
Chen, T.C., Chung, K.L., 2001. A new randomized algorithm for detecting lines. Real-Time Imaging 7, 473–482.
Chung, K.L., Huang, Y.H., 2007. Speed up the computation of randomized algorithms for detecting lines, circles, and ellipses using novel tuning- and lut-based voting platform. Appl. Math. Comput. 190, 132–149.
Chung, K.L., Chang, T.C., Huang, Y.H., 2009. Comment on: Extended hough transform for linear feature detection. Pattern Recognit. 42, 1612–1614.
Chutatape, O., Guo, L., 1999. A modified hough transform for line detection and its performance. Pattern Recognit. 32, 181–192.
Climer, S., Bhatia, S.K., 2003. Local lines: A linear time line detector. Pattern Recognition Lett. 24, 2291–2300.
Davies, E.R., 2004. Maching Vision: Theory, Algorithms, and Practicalities, third ed. Morgan Kaufmann.
Duda, R.O., Hart, P.E., 1972. Use of the hough transformation to detect lines and curves in pictures. Commun. ACM 15, 11–15.
Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24, 381–395.

Fischler, M.A., Firschein, O., 1978. Intelligence: The Eye, the Brain, and the Computer. Addison Wesley.

Forsyth, D.A., Ponce, J., 2002. Computer Vision: A Modern Approach. Prentice Hall.

Gonzalez, R.C., Wood, R.E., 2002. Digital Image Processing, second ed. Prentice Hall, New Jersey.

Hearn, D., Baker, M.P., 1997. Computer Graphics, third ed. Prentice Hall, New York.

Hough, P.V.C., 1992. Method and Means for Recognizing Complex Patterns. Addison-Wesley.

Hough, P.V.C., 1962. Method and means for recognizing complex patterns. US Patent 3,069,654, 18 (December).

Illingworth, J., Kittler, J., 1987. The adaptive hough transform. IEEE Trans. Pattern Anal. Machine Intell. 9, 90–698.

Illingworth, J., Kittler, J., 1988. Surver: Survey of the hough transforms. Computer Vision Graphics Image Process. 44, 87–116.

Jain, R., Kasturi, R., Schunck, B.G., 1995. Maching Vision. MacGraw Hill.

Jiang, H.F., Han, C.C., Fan, K.C., 1997. A fast approach to the detection and correction of skew documents. Pattern Recognition Lett. 18, 675–686.

Kälviäinen, H., Hirvonen, P., 1997. An extension to the randomized hough transform exploiting connectivity. Recognit. Lett. 18, 77–85.

Kälviäinen, H., Hirvonen, P., Xu, L., Oja, E., 1995. Probabilistic and nonprobabilistic hough transforms: Overview and comparison. Image Vision Comput. 13, 239–252.

Kyrki, V., Kälviäinen, H., 2000. Combination of local and global line extraction. Real-Time Imaging 6, 79–91.

Leaver, V.F., 1993. Survey: Which hough transform. CVGIP: Image Understanding 58, 250–264.

Lin, C.J., Tseng, D.C., Lin, C.W., Hu, T., Luo, R., 2008. Forward and sideward collision warning for advanced safety vehicles. in: Proc. 21st IPPR Conf. on Computer Vision, Graphics and Image Processing (CD).

Long, Q., Kanade, T., 1997. Affine structure from line correspondences with uncalibrated affine cameras. IEEE Trans. Pattern Anal. Mach. Intell. 19, 834–845.

Murakami, K., Naruse, T., 2000. High speed line detection by hough transform in local area. Proc. Int. Conf. Pattern Recognit. 3, 467–470.

Schindler, K., 2006. Geometry and construction of straight lines in log-polar images. Computer Vision Image Understanding 103, 196–207.

Su, C.W., Liao, H.Y.M., Tyan, H.R., Lin, C.W., Chen, D.Y., Fan, K.C., 2007. Motion flow-based video retrieval. IEEE Trans. Multimedia 9, 1193–1201.

Tsai, L.W., Hsieh, J.W., Fan, K.C., 2007. Vehicle detection using normalized color and edge map. IEEE Trans. Image Process. 16, 850–864.

Xu, L., Oja, E., 1993. Randomized hough transform: Basic mechanisms, algorithms, and computational complexities. CVGIP: Image Understanding 57, 131–154.

Xu, L., Oja, E., Kultanan, P., 1990. A new curve detection method: Randomized hough transforms. Pattern Recognition Lett. 11, 331–338.