Real-Time Imaging **1**, 419–426 (1995)

A Robust Hough Transform Technique for Complete Line Segment Description

When the Hough transform is applied to the detection of straight lines in images, it provides the parameters but not the *length* or the *end points* of the line. We propose a new algorithm for the determination of the length and the end points of a line in an image. It is very efficient in terms of computing time and does not depend on the sharpness of the peak in the accumulator array.

© 1995 Academic Press Limited

M. Atiquzzaman* and M. W. Akhtar†

**Dept of Electrical & Computer Systems Engineering, Monash University, Clayton 3168, Melbourne, Australia.*
E-mail:atiq@eng.monash.edu.au
†Dept of Electrical Engineering
King Fahd University of Petroleum & Minerals
Dhahran 31261, Saudi Arabia.

Introduction

Detection of patterns in images is an important operation in machine vision tasks. The Hough transform (1) is a powerful technique for the determination of parameterizable patterns in binary images. It is essentially a voting process where each feature point votes for all the possible patterns passing through that point. The votes are accumulated in an accumulator array, and the pattern receiving the maximum vote is considered to be the pattern in the image. The advantages of the transform are its robustness to noise and discontinuities in the patterns. On the other hand, high computational and storage requirements are the drawbacks of the transform. Examples of parameterizable patterns are straight lines, circles, ellipses, etc.

Several variants of the standard Hough transform (SHT) have been proposed in the literature to reduce the time and space complexity (2, 3). Multiprocessor implementations of the transform have also been suggested in order to speed-up the execution of the transform (4,5,6). Investigations regarding the performance of the transform have been reported by VanVeen & Groen and Brown (7, 8).

When the Hough transform is used to detect straight lines represented by the $\rho - \theta$ parameterization, it provides only the ρ and θ parameters of the straight lines. It fails to provide any information regarding the *lengths* or the *end points* of the lines. Since many machine vision tasks require the lengths and end points of lines to determine the location of objects described by the lines accurately, it is extremely important to determine the length and end points of a line. Previous studies to detect the lengths or the end points of the lines are either highly computation bound or are not robust in detecting the end points. Six different algorithms (9–14) have been proposed in the literature for finding the length and end points of a line from the Hough accumulator array. The algorithms described by Yomato *et al.* and Costa

1077-2014/95/060419 + 08 $12.00

© 1995 Academic Press Limited

et al. (9, 10) are based on the projections of the line on the x- or y-axis. The algorithm described by Niblack and Truong (11) is based on the concept of surface fitting and is prohibitively expensive in terms of computing power. The method proposed by Akhtar & Atiquzzaman (12) can detect only the length of a line; it cannot detect the end points.

The algorithm described by Atiquzzaman & Akhtar (13) can detect the length and the end points of a line, but the accuracy of the detection is dependent on the accurate detection of the line parameters. The algorithm is not effective in determining the end points of the line if the ρ and θ parameters of the line cannot be detected accurately. Hence, the method described by Atiquzzaman & Akhtar (13) is not suitable in those cases where the peak cannot be detected accurately. The algorithm described by Richards & Casasent (14) is also based on the accurate determination of the peak and hence suffers from the same drawback experienced by Atiquzzaman & Akhtar (13). The aim of this paper is to develop a robust algorithm to detect the length and end points of lines. The novelty of the proposed algorithm is its independence of the accuracy with which the peak can be detected. For example, it has been shown by VanVeen & Groen (7) that improper choice of discretization values of the accumulator array may lead to unsharp or multiple peaks. A second example would be an image where, after discretization, pixels corresponding to a straight line are not collinear. It is in those images and cases where the proposed algorithm is superior to previously published algorithms.

In this paper, we propose a new algorithm to detect the lengths and the end points of straight lines. The proposed algorithm is based on an analysis of the distribution of votes around the peak in the accumulator array. The phenomenon of the spreading of votes in an accumulator array has been described in detail (7, 13). The algorithm proposed in this paper does not use the values of the ρ and θ parameters of the line in order to detect the end points, and hence is independent of the accuracy with which the line parameters can be detected. Moreover, it has a time complexity of $O(1)$ which is the same as the complexity of the best available algorithm (13). Note that the time complexity of the algorithms previously described (9–12), are $O(n_f n_i + N)$, $O(n_f + N)$, $O(n_f n_i N + N^2)$ and $O(1)$ respectively. n_f, n_i, and N^2 are the number of feature points, the number of iterations, and the size of the accumulator array respectively (12). The proposed algorithm differs from the previously published algorithms in the following:

(i) reduced computational complexity, and
(ii) robustness in detecting the end points.

The non-iterative nature of the algorithm and the use of a small accumulator array account for its reduced computational complexity. The robustness of the algorithm is due to it being independent of the peak detection accuracy. Because of its low computational complexity and high robustness, the algorithm is very suitable for implementation in real-time machine vision tasks. Results show that the proposed algorithm can detect the end points and the line length very accurately.

Notations

A straight line, represented by the normal parameterization, is expressed as

$$\rho = x \cos \theta + y \sin \theta \qquad [1]$$

where ρ is the length of the normal to the line and θ is the angle of the normal with the positive x-direction. The ρ and θ axes of the accumulator array are divided into a number of equal divisions of resolution $\Delta\rho$ and $\Delta\theta$ respectively. The accumulator array is constructed by sampling the values of θ at Δq intervals, and quantizing the ρ values which are computed using Equation [1]. A cell, $a_{\rho,\theta}$, in the accumulator array corresponds in the image plane to a bar-shaped window of infinite length, of width $\Delta\rho$, making an angle θ with the positive x-axis, and at a distance ρ from the origin (7). The cells $a_{\cdot,\theta_k}$ in the accumulator array, therefore, correspond to a set of parallel bars of width $\Delta\rho$ and each making an angle θ_k with the positive x-axis in the image plane as shown in Figure 2. The sets of cells in the different columns of the accumulator array correspond to different sets of parallel bars in the image plane.

A sharp and distinct peak in the accumulator array (Figure 1) is necessary for accurately determining the parameters of the line to be detected. The accuracy with which the line parameters can be detected also depends on the values of $\Delta\rho$ and $\Delta\theta$. In general, smaller values of $\Delta\rho$ and $\Delta\theta$ result in a higher accuracy with which the line parameters can be detected. However, if the $\Delta\rho$ and $\Delta\theta$ are made too small, the detection of the peak becomes difficult due to the spread of the votes in the peak as described by VanVeen and Groen (7). The sharpness and the shape of the peak depend on the discretization of the image, width of the digitized line, and the quantization resolution (denoted by $\Delta\rho$ and $\Delta\theta$) of the accumulator array.

Van Veen *et al.* (7) have demonstrated that the peak in the accumulator array spreads in the ρ direction as $\Delta\rho$ is decreased. On the contrary, a decrease in $\Delta\theta$ results in a

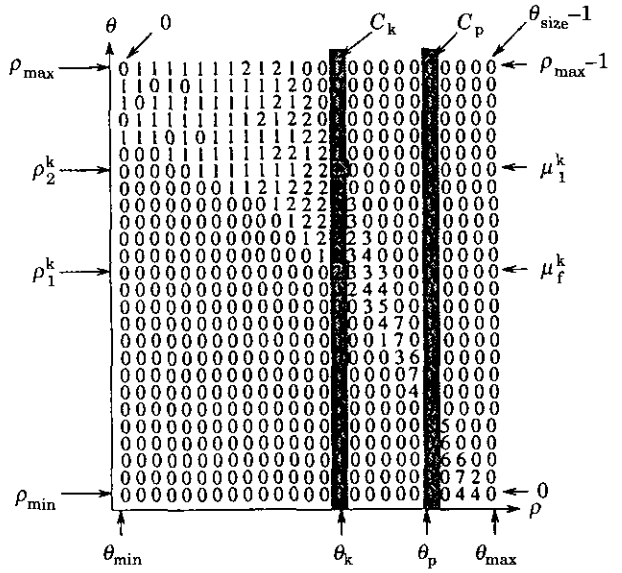

Figure 1. Accumulator array A showing the cells around the peak.

spread of the peak in the θ direction. It is well known that the pixels belonging to a line (expressed in the $\rho - \theta$ parameterization) in an image produce a butterfly shape in the parameter space. The angle subtended by the wings of the butterfly depends on the length and the position of the line (15).

The parameters of a line along with its length and coordinates of the end points are sometimes referred to as the complete line segment descriptions. The following notations will be used in describing the proposed algorithm for finding the complete line segment descriptions. To maintain consistency, we use the same notations as Atiquzzaman & Akhtar (13).

Where

ρ_{size}, θ_{size} = Size of the accumulator array

A = $\{a_{i,j}, 0 \le i \le \rho_{size} - 1, 0 \le j \le \theta_{size} - 1\}$ = Hough accumulator array.

ρ_{min}, ρ_{max}, θ_{min}, θ_{max} = Minimum and maximum values of ρ and θ axes of the accumulator array.

$\Delta\rho$, $\Delta\theta$ = Resolution of the ρ and θ axes of the accumulator.

$b_{i,k}$ = The bar in the image plane corresponding to the cell $a_{i,k}$ in the accumulator array.

C_k = k-th column in the accumulator array. The column consists of the cells $a_{.,k}$.

C_p = The column in the accumulator array containing the peak.

ρ_a, θ_a = Actual parameters of the line.

ρ_p, θ_p = Line parameters obtained from the coordinates of the peak in the accumulator array.

ρ_c, θ_c = Line parameters calculated from the end points of the line by using the method proposed in this paper.

μ_f^k = Row index corresponding to the first non-zero cell in C_k. The first non-zero cell in C_k (see Figure 1) is defined to be the first cell containing a non-zero vote when scanning the cells in C_k from $\rho = 0$ to $\rho_{size} - 1$.

μ_l^k = Row index corresponding to the last non-zero cell in C_k. The last non-zero cell in C_k is the first cell containing a non-zero vote when scanning the cells in C_k from $\rho = \rho_{size} - 1$ down to 0. The number of cells between the first and the last non-zero cells in C_k will be called the spread of votes in C_k.

$n_\rho^k = \mu_l^k - \mu_f^k$ =The spread of votes in C_k.

$d_{q,r} = q - r$ = Number of columns between C_q and C_r.

l_c = Length of the line computed from the coordinates of the end points obtained from the proposed algorithm.

ϵ_x = Error in the x-coordinates of the end points obtained from the proposed algorithm.

ϵ_y = Error in the y-coordinates of the end points obtained from the proposed algorithm.

ρ_1^k = Length of the normal to the bar (in the image plane) corresponding to the first non-zero cell in C_k. Note that the first non-zero cell in C_k corresponds to the bar $b_{i,k}$ in the image plane containing the end point (x_2, y_2) in Figure 2.

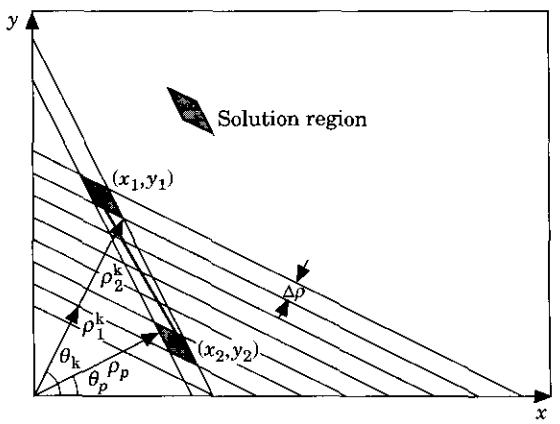

Figure 2. Illustration of parallel bars corresponding to the cells in Ck.

ρ_2^k = Length of the normal to the bar (in the image plane) corresponding to the last non-zero cell in C_k. Note that the last non-zero cell in C_k corresponds to the bar $b_{i,k}$ in the image plane containing the end point (x_1, y_1) in Figure 2.

The Proposed Method

As mentioned in the previous section, the cells $a_{.,\theta_k}$ correspond to a set of parallel bars of width $\Delta\rho$ in the image plane. The number of non-zero cells in $a_{.,\theta_k}$ (called the spread in $a_{.,\theta_k}$) is equal to the number of parallel bars (in the set corresponding to $a_{.,\theta_k}$) intersected by the line. The proposed algorithm is based on a microanalysis of the distribution of the votes around the peak in the accumulator array.

A flow chart describing the proposed algorithm is given in Figure 3 followed by a detailed explanation of the algorithm.

Atiquzzaman & Akhtar (13) obtained the end points of a line by determining the intersecting points between the bar corresponding to the peak (in the accumulator array) and the two bars corresponding to the cells μ_f^k and μ_l^k in any column C_k. The accuracy of the detection therefore, depends on:

(i) the choice of C_k, and
(ii) the accuracy with which the peak can be detected.

An algorithm for the choice of C_k has been described by Atiquzzaman & Akhtar (13). However, an accurate detection of the peak in the accumulator array is a non-trivial task (8). The accuracy with which the peak can be detected depends on the sharpness and uniqueness of the peak. It has been shown (7) that the peak in the accumulator array may spread due to discretization of the image space and the accumulator array. Due to the dependence of the algorithm, described in Atiquzzaman & Akhtar (13), on the accuracy with which the peak can be detected, the end points obtained from the above algorithm are not always reliable. In this paper, we propose an algorithm which is independent of the accuracy with which the peak is determined. Hence the proposed algorithm is more reliable and robust than the one described previously (13). In the proposed algorithm, the θ value of the peak (θ_p) is only used to determine two columns, C_q and C_r, as described below.

Consider two columns C_q and C_r whose cells correspond to the two sets of parallel bars having their normals inclined at angles θ_q and θ_r respectively with the positive x-axis. The lengths of the normals ρ_1^q, ρ_2^q, ρ_1^r, and ρ_2^r (see Figure 4) to the bars corresponding to the cells μ_f^q, μ_l^q, μ_f^r, and μ_l^r respectively are determined from the accumulator array. The above lengths are the lengths of the normals to the bars

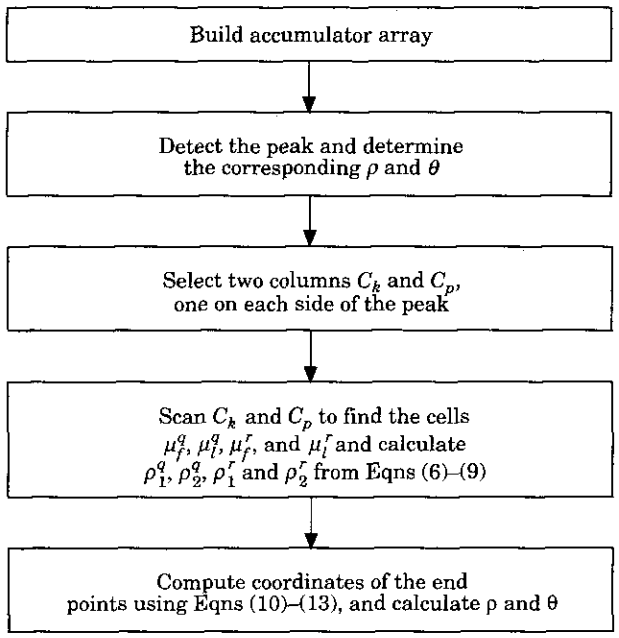

Figure 3. Flowchart of the proposed algorithm.

corresponding to the first and last non-zero elements in C_q and C_r. These normals can be expressed as

$$\rho^q_1 = x_1 \cos\theta_q + y_1 \sin\theta_q \quad [2]$$

$$\rho^r_1 = x_1 \cos\theta_r + y_1 \sin\theta_r \quad [3]$$

$$\rho^q_2 = x_2 \cos\theta_q + y_2 \sin\theta_q \quad [4]$$

$$\rho^r_2 = x_2 \cos\theta_r + y_2 \sin\theta_r \quad [5]$$

ρ^q_1, ρ^q_2, ρ^r_1, and ρ^r_2 can be expressed by the following equations (see Figures 1 and 4).

$$\rho^q_1 = \rho_{\min} + \mu^q_f \Delta\rho \quad [6]$$

$$\rho^q_2 = \rho_{\min} + \mu^q_l \Delta\rho \quad [7]$$

$$\rho^r_1 = \rho_{\min} + \mu^r_f \Delta\rho \quad [8]$$

$$\rho^r_2 = \rho_{\min} + \mu^r_l \Delta\rho \quad [9]$$

Since $\rho_{\min}$ and $\Delta\rho$ are known, ρ^q_1, ρ^q_2, ρ^r_1, and ρ^r_2 in Equations (6)–(9) can be calculated after μ^q_f, μ^q_l, μ^r_f, and μ^r_l are obtained by scanning the columns C_q and C_r for the first and last non-zero elements. The calculated values of ρ^q_1, ρ^q_2, ρ^r_1, and ρ^r_2 are then substituted in Equations (2)–(5). Since θ_q and θ_r are known from the choice of C_q and C_r, solution of Equations (2) and (3) will give the coordinates (x_1, y_1) of one of the end points. The other end point (x_2, y_2) can be obtained similarly by solving Equations (4) and (5). The coordinates obtained by solving the above two sets of equations are as follows.

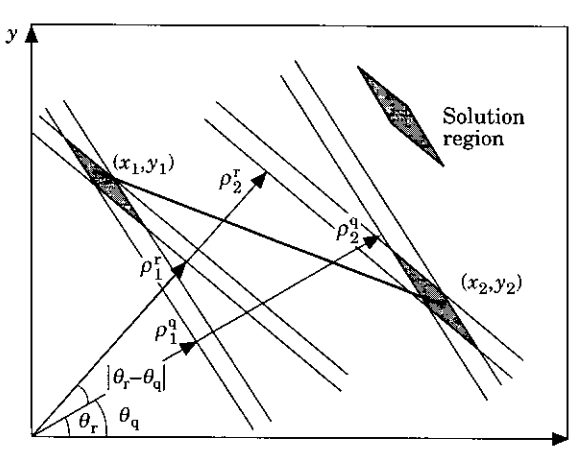

Figure 4. Computation of the end points independent of θ_p for $\theta_p > 45°$.

$$x_1 = \frac{\rho^q_1 \sin\theta_r - \rho^r_1 \sin\theta_q}{\sin(\theta_r - \theta_q)} \quad [10]$$

$$y_1 = \frac{\rho^r_1 \cos\theta_q - \rho^q_1 \cos\theta_r}{\sin(\theta_r - \theta_q)} \quad [11]$$

$$x_2 = \frac{\rho^q_2 \sin\theta_r - \rho^r_2 \sin\theta_q}{\sin(\theta_r - \theta_q)} \quad [12]$$

$$y_2 = \frac{\rho^r_2 \cos\theta_q - \rho^q_2 \cos\theta_r}{\sin(\theta_r - \theta_q)} \quad [13]$$

We define a thick line as one which has $\theta_a \neq 45°$. Thick lines exhibit the phenomenon of a line being split up into a number of smaller horizontal or vertical line segments. Due to the splitting of a thick line into several smaller line segments, the following rules regarding the choice of C_q and C_r have to be applied for the accurate detection of the end points of a thick line. The reasons behind the above rules are explained elsewhere (13).

- *Rule 1*: Select C_q (or C_r), q (or r) $> p$, if $-45° < \theta_a < 45°$, $135° < \theta_a < 225°$.
- *Rule 2*: Select C_q (or C_r), q (or r) $< p$, if $45° < \theta_a < 135°$, $225° < \theta_a < 315°$.

The above rules determine whether C_q and C_r should be chosen to the left or the right of C_p. To apply the above rules, θ_a can be approximated by θ_p. For a thin line, defined as a line having $\theta_a = 45°$, it does not matter whether C_q and C_r are chosen to the left or the right of C_p.

Once the end points of a line are determined from the above algorithm, the line length (l_c) is obtained from the end points by

$$l_c = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad [14]$$

and the normal parameters are obtained as

$$\rho_c = \frac{x_2 y_1 - x_1 y_2}{\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}} \quad [15]$$

$$\theta_c = \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right) - 90° \quad [16]$$

The maximum errors in the coordinates of the end points depend on the area enclosed by the intersection of the two

bars (C_q and C_r) as shown by the shaded parallelograms in Figure 4. The maximum errors in the x and y-coordinates can be easily shown to be

$$\epsilon_x = \frac{(\Delta\rho/2)\sin((\theta_q+\theta_r)/2)}{\sin((\theta_r-\theta_q)/2)} \quad [17]$$

$$\epsilon_y = \frac{(\Delta\rho/2)\cos((\theta_q+\theta_r)/2)}{\sin((\theta_r-\theta_q)/2)} \quad [18]$$

From Equations [17] and [18] it is seen that the errors can be reduced by:

(i) increasing $\Delta\theta$, thereby making $|\theta_r - \theta_q|$ large (Increasing $\Delta\theta$ has the additional advantage of requiring less computational time to construct the accumualtor array.) and
(ii) choosing two columns C_q and C_r which are far apart, i.e., $|q - r|$ is large.

Results

In this section, we present results regarding the accuracy of the end points and line length from the proposed algorithm. The main focus of this paper is to illustrate the effectiveness of the proposed algorithm in accurately determining the end points of a straight line. The effectiveness is illustrated by comparing the difference between the coordinates obtained from the proposed algorithm and the actual coordinates of the line in the image. The parameters and end points of a straight line generated in a synthetic image are known much more precisely than a line in a real-world image. Therefore, to determine the accuracy of the results obtained from the proposed algorithm and calibrating the results, we have used synthetic images in our experiments.

Using the algorithm proposed earlier, the errors in the line length are shown in Figure 5 for different values of $\Delta\rho$. A line with $\rho_a = 90$ and $\theta_a = 25°$ was used with $\Delta\theta = 0.7087$. The errors are lower than those obtained from the previous algorithm (13). The increased accuracy is due to the proposed algorithm being independent of the accuracy of the line parameters.

Note that the errors decrease with an increase in $d_{q,r}$. This is expected since the area of the solution region (shaded parallelogram in Figure 4) decreases with an increase in $d_{q,r}$ due to a reduction in the angle between the bars corresponding to the cells in C_q and C_r.

The variation of the average errors in the length and end

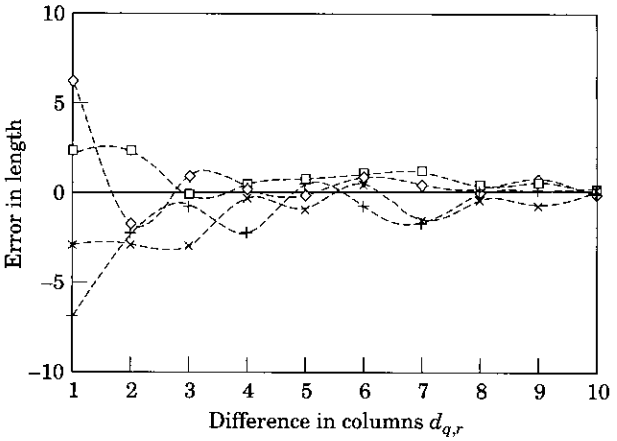

Figure 5. Errors in the line length *vs.* $d_{q,r}$ for a line having ρ_a = 90 and $\theta_a = 25°$. □, $\Delta\rho = 0.0898$; ◊, $\Delta\rho = 0.0998$; +, $\Delta\rho = 0.1123$; ×, $\Delta\rho = 0.1283$.

points *vs.* $d_{q,r}$ are shown in Figures 6 and 7 for different combinations of $\Delta\rho$ and $\Delta\theta$. The average errors were obtained by applying the proposed algorithm to a large number of lines with different values of ρ and θ and taking the mean of the errors from the different lines. Different combinations of $\Delta\rho$ and $\Delta\theta$ have been used in the above figures to show the effectiveness of the proposed algorithm for different values of $\Delta\rho$ and $\Delta\theta$. The errors decrease with a decrease in $\Delta\rho$, as can be seen from the lower errors in Figure 7 compared to the errors in Figure 6.

The effects of varying $\Delta\rho$ and $\Delta\theta$ (keeping $d_{q,r}$ constant) are shown in Figures 8 and 9. Figure 8 shows the error in the determination of the line length as a function of $\Delta\rho$ for $d_{q,r} = 10$ and three different values of $\Delta\theta$ (0.31, 0.47 and 1.0). It is seen that the errors increase with an increase in $\Delta\rho$ and/or a decrease in $\Delta\theta$. This is because the area of the solution region increases with an increase in $\Delta\rho$ and/or a decrease in $\Delta\theta$, and thus validates Equations [17] and [18]. Figure 9 shows the effect of $\Delta\rho$ and $d_{q,r}$ (keeping $\Delta\theta$ constant) on the errors in the computed length. It is observed that the errors increase with a decrease in $d_{q,r}$ and an increase in $\Delta\rho$. The above observation further confirms the validity of Equations [17] and [18]. Figure 10 shows a real image and the corresponding image showing straight lines which have been detected. The proposed algorithm requires about 5 s of execution time for a 256 × 256 binary image on a SUN Sparcstation.

Conclusions

The Hough transform, when applied to the detection of straight lines in images, provides only the parameters of the line. It fails to provide the length or the end points of the

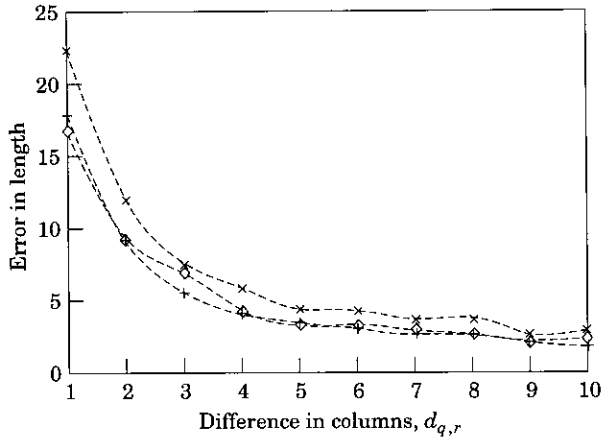

Figure 6. Average errors in the line length and end points *vs.* $d_{q,r}$ for $\Delta\rho = 0.9$ and $\Delta\theta = 1.0$. ◊, error in x_2, y_2; +, error in x_1, y_1; ×, error in length.

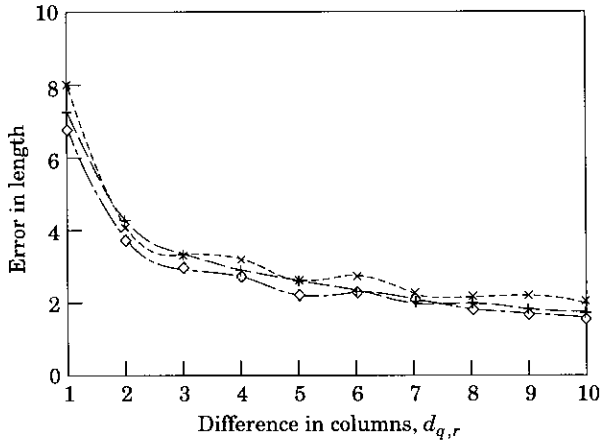

Figure 7. Average errors in the line length and end points *vs.* $d_{q,r}$ for $\Delta\rho = 0.2$ and $\Delta\theta = 0.7$. ◊, error in x_2, y_2; +, error in x_1, y_1; ×, error in length.

line. Previous extensions of the Hough transform for obtaining the length or the end points are either highly computation intensive or rely on the ability of the standard Hough transform to detect the parameters of the line accurately. In this paper, we have proposed an improved algorithm to detect the end points and determine the line length. The proposed algorithm is very robust against the failure of the standard Hough transform to detect the parameters of the line due to spreading of the peak in the accumulator array. It is not computation intensive and has a time complexity of $O(1)$ which is the lowest among the algorithms reported in the literature.

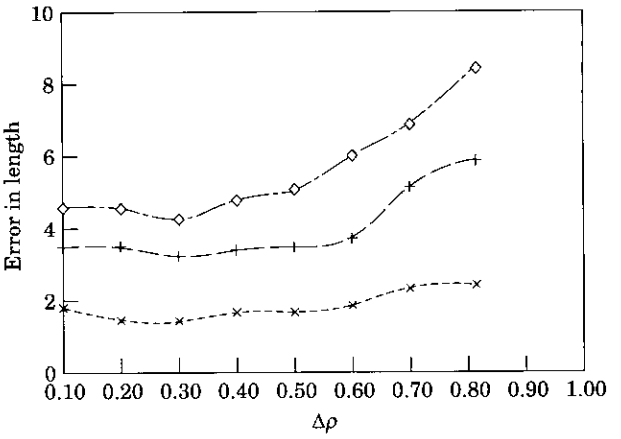

Figure 8. Variation in average error in the line length *vs.* $\Delta\rho$ for different values of $d_{q,r}$ and $\Delta\theta = 10$. ◊, $\Delta\theta = 0.31$; +, $\Delta\theta = 0.47$; ×, $\Delta\theta = 1.0$.

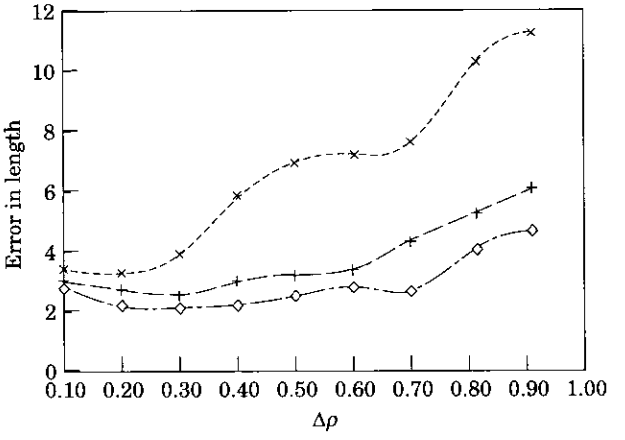

Figure 9. Variation in average error in the line length *vs.* $\Delta\rho$ for different values of $d_{q,r}$ and $\Delta\theta = 0.7$. ◊, $d_{q,r} = 9$; +, $d_{q,r} = 6$; ×, $d_{q,r} = 3$.

When the Hough transform fails to determine the line parameters due to the peak in the accumulator array not being very sharp, the proposed algorithm can serve as an alternative method to compute the parameters using the coordinates of the end points. Thus the algorithm can be used to obtain the parameters or to verify the parameters obtained from the peak in the accumulator array by the standard Hough transform.

Results show that the proposed algorithm can detect the length and the end points very accurately. The proposed algorithm is non-iterative in nature and is based on a micro-analysis of the spreading of the votes in the accumulator array. This approach makes the algorithm very robust compared to previous algorithms, most of which analyse the image space to obtain the line length and the end points.

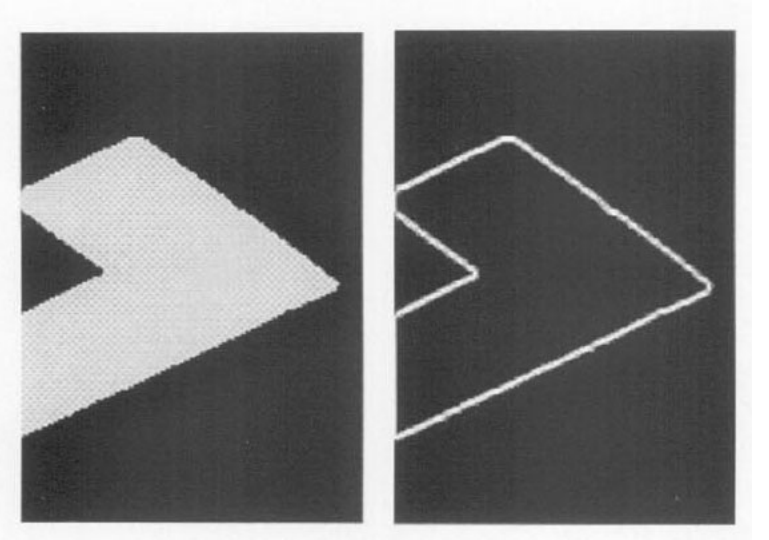

Figure 10. Real-world image and the corresponding image showing detected lines.

References

1. Hough, P.V.C. (1962) Methods and means for recognizing complex patterns. U.S. Patent 3069654.
2. Atiquzzaman, M. (1992) Multiresolution Hough transform – an efficient method of detecting pattern in images, *IEEE Trans. Patt. Anal. Mach. Intell.*, **14**(11): 1090–1095.
3. Illingworth, J. & Kittler, J. (1987) Adaptive Hough transform, *IEEE Trans. Patt. Anal. Mach. Intell.*, **PAMI-9**(5): 690–698.
4. Atiquzzaman, M. (1994) Pipelined implementation of the multiresolution Hough transform in a pyramid multiprocessor, *Patt. Recog. Lett.*, **15**(9): 841–851.
5. Fisher, A.L. & Highnam, P.T. (1989) Computing the Hough transform on a scan line array processor, *IEEE Trans. Patt. Anal. Mach. Intell.*, **11**(3): 262–265.
6. Ben-Tzvi, D., Naqvi, A. & Sandler, M. (1990) Synchronous multiprocessor implementation of the Hough transform, *Comp. Vision, Graph. and Image Process.*, **52**: 437–446.
7. VanVeen, T.M. & Groen, F.C.A. (1981) Discretization errors in the Hough transform, *Patt. Recog.*, **14**: 137–145.
8. Brown, C.M. (1983) Inherent bias and noise in the Hough transform, *IEEE Trans. Patt. Anal. Mach. Intell.*, **PAMI-5**(5): 493–505.
9. Yamato, J., Ishii, I. & Makino, H. (1990) Highly accurate segment detection using Hough transformation, *Sys. Comp. Japan*, **21**(1): 68–77.
10. Costa, L.F., Ben-Tzvi, B. & Sandler, M. (1990) Performance improvements to the Hough transform, UK IT 1990 Conference, London, Mar 1990, pp. 98–103.
11. Niblack, W. & Truong, T. (1990) Finding line segments by surface fitting to the Hough transform, *IAPR International Workshop on Machine Vision and Applications*, Tokyo, Japan, Nov. 28–30, 1990.
12. Akhtar, M.W. & Atiquzzaman, M. (1992) Determination of line length using Hough transform, *Elec. Lett*, **28**(1): 94–96.
13. Atiquzzaman, M. & Akhtar, M.W. (1994) Complete line segment description using the Hough transform, *Image Vision Comp.*, **12**(5): 267–273.
14. Richards, J. & Casasent, D.P. (1991) Extracting input-line position from Hough data, *Appl. Opt.*, **30**(20): 2899–2905.
15. Leavers, V.F. & Boyce, J.F. (1987) The Radon transform and its application to shape parametrization in machine vision, *Image Vision Comp.*, **5**(2): 161–166.