PERGAMON

Pattern Recognition 33 (2000) 1105–1117

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

A window-based inverse Hough transform

A.L. Kesidis, N. Papamarkos*

Electric Circuits Analysis Laboratory, Department of Electrical Computer Engineering, Democritus University of Thrace, 67100 Xanthi, Greece

Abstract

In this paper a new Hough transform inversion technique is proposed. It is a window-based inverse Hough transform algorithm, which reconstructs the original image using only the data of the Hough space and the dimensions of the image. In order to minimize memory and computing requirements, the original image is split into windows. Thus, the algorithm can be used to large-size images as a general purpose tool. In this paper, the proposed technique is applied for edge extraction and filtering. The edges are detected not just as continuous straight lines but as they really appear in the original image, i.e. pixel by pixel. Experimental results indicate that the method is robust, accurate and fast. © 2000 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: Inverse Hough transform; Edge extraction; Line detection; Non-linear filtering

1. Introduction

The Hough Transform (HT) is one of the most often used tools for curve detection. The original HT is commonly used to detect straight lines in a binary image, and was first proposed by Hough [1]. It is a voting process where each point (pixel) of the original binary image votes for all possible patterns (straight lines) passing through that point [2]. The votes are accumulated in an accumulator array, in which its peaks correspond to line segments. However, the information given by the peaks of the accumulator array are only the polar parameters of the straight line and the total number of pixels that belong to it. Unfortunately, the HT does not determine the exact position of each pixel in the straight lines.

The main advantages of the HT are its robustness to image noise and that it can determine the slope and the distance from the origin (polar parameters) of discontinuous straight lines. The disadvantages of the HT are associated with its large storage and computational requirements. For this reason many approaches have been proposed in the literature, regarding the reduction of computation time and memory requirements [3–8]. Additional techniques have been proposed to improve the accuracy [9–12] and to analyze the quantization effects of the Hough space [13,14]. Duda and Hart [15] improved the HT algorithm and extend it to the detection of other geometrical shapes. Ballard [16] introduced the generalized HT that could find arbitrary shapes of any orientation and scale. Additionally, Chatzis and Pitas introduced the fuzzy cell HT, which using fuzzy split of the Hough space detects shapes with better accuracy, especially in noisy images [17].

As it was mentioned above, HT cannot determine the exact position of the pixels of a straight line. This is a serious disadvantage in many applications such as edge detection via the HT. In this case, it is required to know the pixels of the edges and not only the polar coordinates of the edges. The solution of this problem can be achieved by the development of an inverse Hough transform (IHT) technique. The IHT can be defined as the technique that permits the detection of the original binary image knowing only its size and the data of the Hough space. No further information about the image is needed. Recently, a general IHT algorithm has been proposed by Kesidis and Papamarkos [18]. This IHT algorithm can be considered as a decomposition procedure which by checking the existence of the sinusoidal curve peaks in the

* Corresponding author. Tel.: +30-541-79585; fax: +30-541-79569.

E-mail address: papamark@voreas.ee.duth.gr (N. Papamarkos)

PII: S0031-3203(99)00167-3

Hough space, identifies the curves and reconstructs the original image pixel by pixel. In order to have correct inversion, the size of the accumulator array must satisfy some conditions. These conditions are analytically stated and are associated with the scale coefficients that control the size of the accumulator array. However, the necessary size of the accumulator array, the memory requirements and the computation time increase significantly with the size of the original image. Therefore, for large-size images the application of the proposed IHT is unpractical. To solve this problem we propose a Window-based inverse Hough transform (WIHT) algorithm. The method considers the original image, as a sum of not overlapped rectangular windows. In other words, the original image is split in $n^2, n = 1, 2, \ldots,$ windows and HT and IHT are applied independently to each of them. The proposed algorithm is suitable for large size images. As an application, we describe the use of WIHT for edge detection. In the last stage of the edge extraction algorithm via WIHT, a filtering merging procedure is applied. This produces the final filtered edges taking into account the filtering edge results of each window and the global filtering requirements. It can be noticed that the extracted edges include all pixels in the correct positions as these appear in the original image. The proposed algorithm is robust and always applicable to any size of binary images. In this paper, we provide representative examples that cover different types of edge extraction and filtering. The experimental results shown confirm the effectiveness of the proposed method.

The rest of this paper is arranged as follows. Section 2 gives definitions of the HT and discusses the quantization problems of the discrete HT implementation. In Section 3 the inversion conditions are formulated and the proper values of the scale coefficients are defined. Section 4 summarizes the IHT algorithm and its implementation. Section 5 analyzes the new WIHT algorithm and describes its application. Section 6 gives some experimental and comparative results of the application of the WIHT algorithm and demonstrates its suitability for edge extraction and filtering. Finally, Section 7 presents the conclusions.

2. Definitions of the Hough transform

In order to analyze our method it is necessary to provide some definitions and discuss the quantization problems associated with the discrete form implementation of the HT. The HT maps a line (not necessarily a straight line) of the image space (x, y) into a point in the Hough space. A definition of the HT is based on the polar representation of lines

$$\rho = x_i \cos\theta + y_i \sin\theta. \tag{1}$$

All points (x_i, y_i) of a line in the binarized image space correspond to a point (θ, ρ) in the Hough space. Additionally, any point (x_i, y_i) in the image space is mapped to a sinusoidal curve in the HT space. For this reason, the HT can be considered as a point-to-curve transformation. In the discrete case the Hough space is an accumulator array. In the accumulator array C, if $1/sf_\theta$ is the step interval for variable θ, then $\theta \in [-90^\circ, -90^\circ + 1/sf_\theta, \ldots, 180^\circ]$. Let also

$$\theta_C = \theta\, sf_\theta$$

and

$$\tilde{\theta}_C = \text{Round}\,(\theta_C), \tag{2}$$

where the Round(.) function gives the nearest integer of (.). Similarly, $1/sf_\rho$ is the step interval for variable ρ and $\rho \in [\rho_1, \rho_1 + 1/sf_\rho, \ldots, \rho_2]$ where ρ_1 and ρ_2 denote the minimum and maximum values of ρ. Also it is defined that

$$\rho_C = \rho \cdot sf_\rho \tag{3}$$

and

$$\tilde{\rho}_C = \text{Round}\,(\rho_C). \tag{4}$$

For each point (x_i, y_i) the peak coordinates (θ_M, ρ_M) of the sinusoidal curve in the HT space are given by

$$\frac{\mathrm{d}\rho}{\mathrm{d}\theta} = 0 \Rightarrow \theta_M = \tan^{-1}\left(\frac{y_i}{x_i}\right) \tag{5}$$

and

$$\rho_M = x_i \cos\theta_M + y_i \sin\theta_M. \tag{6}$$

Generally, for any value of sf_θ and sf_ρ the coordinates of each peak are given by the equations

$$\theta_{CM} = \theta_M sf_\theta \tag{7}$$

and

$$\rho_{CM} = \rho_M sf_\rho. \tag{8}$$

At the peak of each curve in the HT space, there is a region around θ_{CM} defined by $\pm\,\delta\theta_C$ where the $\tilde{\rho}_C$ values are constant due to the effect of the round function. That is, if ρ_C belongs to the interval

$$\tilde{\rho}_{CM} - 0.5 \leqslant \rho_C < \tilde{\rho}_{CM} + 0.5, \tag{9}$$

then

$$\tilde{\rho}_C = \text{Round}\,(\rho_C) = \tilde{\rho}_{CM}. \tag{10}$$

Also

$$\begin{aligned}\rho_M - \rho &= x\cos\theta_M + y\sin\theta_M - x\cos(\theta_M + \delta\theta) \\ &\quad - y\sin(\theta_M + \delta\theta) \\ &= \rho_M(1 - \cos\delta\theta) \Rightarrow \rho = \rho_M \cos\delta\theta.\end{aligned} \tag{11}$$

In the general case, and for any value of sf_θ it is assumed that

$$\rho_C = \rho_{CM} \cos\left(\frac{\delta\theta_C}{sf_\theta}\right). \tag{12}$$

Since $\delta\theta_C$ is symmetrically distributed around θ_{CM}, Eqs. (9) and (12) give

$$\delta\theta_C = sf_\theta \cos^{-1}\left(\frac{\tilde{\rho}_{CM} - 0.5}{\rho_{CM}}\right). \tag{13}$$

The range of the angle values where $\tilde{\rho}_C = \tilde{\rho}_{CM}$(depicted in Fig. 1) is given by the following equations

$$\delta\tilde{\theta}_{CL} = \text{Trunc}\,(\tilde{\theta}_{CM} - (\theta_{CM} - \delta\theta_C)), \tag{14}$$

$$\delta\tilde{\theta}_{CR} = \text{Trunc}\,((\theta_{CM} + \delta\theta_C) - \tilde{\theta}_{CM}), \tag{15}$$

where

$$\tilde{\theta}_{CM} = \text{Round}\,(\theta_{CM}), \tag{16}$$

$$\tilde{\rho}_{CM} = \text{Round}\,((x\cos\tilde{\theta}_{CM} + y\sin\tilde{\theta}_{CM})\,sf_\rho) \tag{17}$$

and Trunc is the truncation function.

In Fig. 1 can be observed the peak region of the curve of pixel (18,19) for $sf_\theta = 1$ and $sf_\rho = 5$. It is $\theta_{CM} = 46.55$, $\rho_{CM} = 130.86$, $\tilde{\theta}_{CM} = 47$ and $\tilde{\rho}_{CM} = 131$. The $\delta\theta_C$ value equals to 4.24 while the angle width on the left side of $\tilde{\theta}_{CM}$ is $\delta\tilde{\theta}_{CL} = 4$ and on the right $\delta\tilde{\theta}_{CR} = 3$.

3. Determination of the scale coefficients

The inversion of the HT is possible only if the dimensions of the accumulator array C satisfy some lower bounds. These dimensions are defined by the scale coefficients sf_θ and sf_ρ. In this section, a method is described which determines the minimum values of the scale coefficients that permit the inversion of the HT.

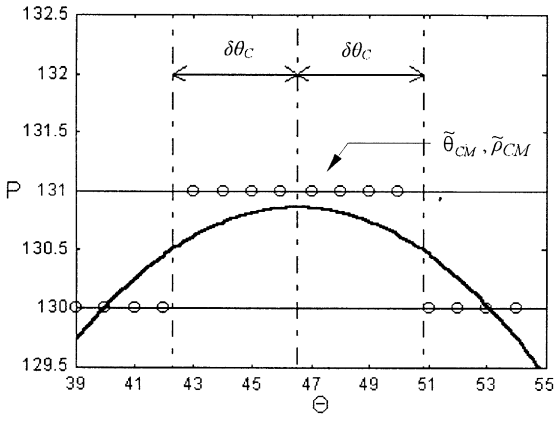


Fig. 1. Description of a peak region. The solid line indicates the real values while the circles depict the discrete elements of the accumulator array C.

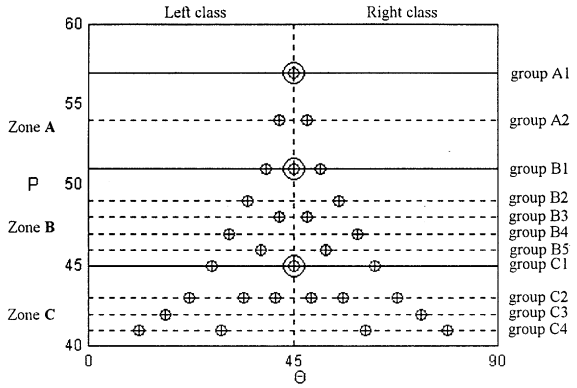


Fig. 2. The curve peaks in the upper three zones of the HT space of an 10×10 image array A having all pixels on.

Let us consider the general case of a binary image A of $N \times N$ pixel size, which has all the pixel values equal to one. If the image dimensions are $N_1 \times N_2$ where $N_1 \neq N_2$, then, without the loss of generality, it can be considered that $N = \text{maximum}\{N_1, N_2\}$. According to the previous analysis, in accumulator array C, the peaks of the curves of the diagonal pixels of A are located at $\theta_D = 45sf_\theta$. The coordinates of those peaks are given by Eqs. (16) and (17).

To determine the minimum values for the scale coefficients it is necessary the N^2 curve peaks of the image A to be sorted according to their $\tilde{\rho}_{CM}$ value. Instead of sorting all of them, they are divided in horizontal zones, groups and classes as depicted in Fig. 2.

Each zone is defined by the $\tilde{\rho}_{CM}$ value of two consecutive pixels in the diagonal of the matrix A (marked with a circle in Fig. 2). The peaks in each zone are sorted in descending order and then are separated into groups so that the elements of each group have the same $\tilde{\rho}_{CM}$ value. Next, the elements of each group are divided in two classes according to their $\tilde{\theta}_{CM}$ value. In the left class belong the elements of the group that have $\tilde{\theta}_{CM} < \theta_D$, while the right class contains the elements that have $\tilde{\theta}_{CM} \geqslant \theta_D$. Equivalently, the separation can also be done as follows: the elements with $x_i > y_i$ and $x_i \leqslant y_i$ belong to left and right classes, respectively.

As it has been analyzed above, for each sinusoidal curve in the HT space there is a region $\delta\theta_C$ around the peak θ_{CM}, where, due the quantization, the values $\tilde{\rho}_C$ of the curve points are equal to $\tilde{\rho}_{CM}$. Therefore, each curve i has $\tilde{\rho}_C$ values equal to its maximum $\tilde{\rho}_{CM}$ within an angle range $[\tilde{\theta}^{(i)}_{CM} - \delta\tilde{\theta}^{(i)}_{CL}, \tilde{\theta}^{(i)}_{CM} + \delta\tilde{\theta}^{(i)}_{CR}]$, where $\delta\tilde{\theta}^{(i)}_{CL}$, $\delta\tilde{\theta}^{(i)}_{CR}$ and $\tilde{\theta}^{(i)}_{CM}$ are given by Eqs. (14)–(16), respectively.

3.1. Class overlapping

Let i, j denote two curves of a right class with $\tilde{\theta}_{CM}^{(i)} < \tilde{\theta}_{CM}^{(j)}$. If $\tilde{\theta}_{CM}^{(i)} + \delta\tilde{\theta}_{CR}^{(i)} < \tilde{\theta}_{CM}^{(j)} + \delta\tilde{\theta}_{CR}^{(j)}$, then there is no overlapping (Fig. 3). This means that there is a number of points (at least one) on the right side of row $\tilde{\rho}_{CM}$, which are contributed only by the right curve j. The furthest right of these points is the *characteristic point* of the curve that allows the detection of the curve during the inversion process. Similarly, for two curves i, j of a left class with $\tilde{\theta}_{CM}^{(i)} < \tilde{\theta}_{CM}^{(j)}$ if $\tilde{\theta}_{CM}^{(i)} - \delta\tilde{\theta}_{CL}^{(i)} < \tilde{\theta}_{CM}^{(j)} - \delta\tilde{\theta}_{CL}^{(j)}$, then there is a number of points (at least one) on the left side of row $\tilde{\rho}_{CM}$ which are contributed only by the left curve i. The furthest left is the *characteristic point* of the curve that allows the detection of the curve during the inversion process.

In general, starting from a small value of sf_ρ and by gradually increasing it we can achieve separation of all curve peaks into distinguished classes so that *each* left or right class satisfies the following condition:

- For a left class

$$\tilde{\theta}_{CM}^{(S)} - \delta\tilde{\theta}_{CL}^{(S)} < \tilde{\theta}_{CM}^{(S+1)} - \delta\tilde{\theta}_{CL}^{(S+1)} \tag{18}$$

with $s = 1, \ldots, k_L - 1$, where k_L is the number of class members sorted from left to right according to the distances

$$|\tilde{\theta}_{CM}^{(i)} - \theta_D|, i = 1, \ldots, k_L \tag{19}$$

- For a right class

$$\tilde{\theta}_{CM}^{(S-1)} + \delta\tilde{\theta}_{CR}^{(S-1)} < \tilde{\theta}_{CM}^{(S)} + \delta\tilde{\theta}_{CR}^{(S)} \tag{20}$$

with $s = 2, \ldots, k_R$, where k_R is the number of class members sorted from right to left according to the distances

$$|\tilde{\theta}_{CM}^{(i)} - \theta_D|, i = 1, \ldots, k_R. \tag{21}$$

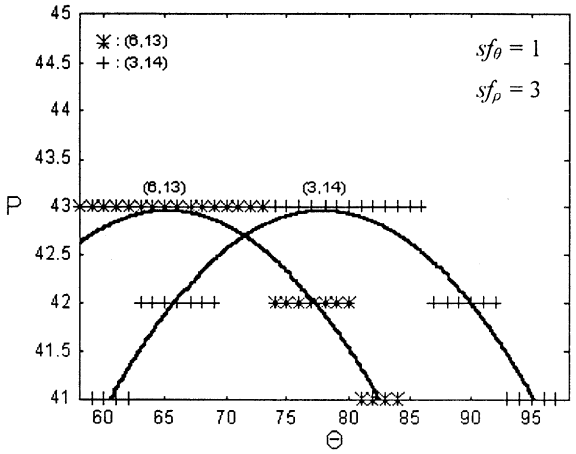


Fig. 3. The curves (6, 13) and (3, 14) in the right class of group in row $\tilde{\rho}_{CM} = 43$.

3.2. Group overlapping

Let also i, j denote two curves one from the left and one from the right class of a given group. The upper parts of all possible pairs i, j must differ at one point at least. Starting from a small value of sf_θ and by gradually increasing it the following process is iteratively applied to determine the scale coefficient sf_θ:

- In every group, for each element i of the left class and each element j of the right class, one of the next inequalities must be satisfied

$$\tilde{\theta}_{CM}^{(i)} - \delta\tilde{\theta}_{CL}^{(i)} < \tilde{\theta}_{CM}^{(j)} - \delta\tilde{\theta}_{CL}^{(j)}$$

or

$$\tilde{\theta}_{CM}^{(i)} + \delta\tilde{\theta}_{CR}^{(i)} < \tilde{\theta}_{CM}^{(j)} + \delta\tilde{\theta}_{CR}^{(j)}. \tag{22}$$

Summarizing, for any square image matrix A, of size $N \times N$, the original image can be reconstructed correctly by using only the array C, if the scale coefficients sf_ρ and sf_θ have values that satisfy the conditions (18), (20) and (22), respectively. These conditions are referred to as the *Inversion Conditions*.

In general, the scale coefficients do not depend on the form of the image but only on its dimensions. Therefore, it is not necessary to apply the above procedure for scale coefficients determination in every image under study. Alternatively, the minimum (optimal) values of the scale coefficients can be directly obtained from a table such as Table 1, which gives the values of sf_θ and sf_ρ for several image dimensions.

4. The inversion procedure

Using the above analysis and definitions we developed an IHT procedure which permits the exact reconstruction of the original image by using only the HT space.

Let us consider an accumulator array C of a HT space corresponding to an $N \times N$ pixel image array A. Let also

Table 1
Minimum scale coefficients sf_θ and sf_ρ for several values of image dimension N

Image dimension N	sf_θ	sf_ρ
10	1	4
25	1	9
50	1	17
100	2	34
150	3	53
200	4	68
250	5	89
300	6	102

A_{inv} be an $N \times N$ image array, where the reconstructed image is stored. We suppose that the pixels of the original image that are equal to one have been transformed to the HT space. The corresponding sinusoidal curves are separated into groups and classes as mentioned above. The decomposition process of the IHT algorithm is a top-down procedure that runs from the "upper" groups (higher $\tilde{\rho}_{CM}$ value) to the "lower" ones and from the "outer" member of each class (greater $|\tilde{\theta}_{CM} - \theta_D|$ value) to the "inner". Analytically, the procedure is as follows:

Step 1: Examine the groups from up to down, according to the above-mentioned separation of the curves into zones, groups and classes.

Step 2: Examine first the "outer" members of the group and then the "inner" ones, according to their $|\tilde{\theta}_{CM} - \theta_D|$ value. That is, examine successively the furthest left member of the left class, the furthest right member of the right class, the second furthest left member of the left class and so on. For each examined curve go to Step 3 if it belongs to the left class or to Step 5 if it belongs to the right class. If all the members of the group are examined then go to Step 1 and continue with the next lower group.

Step 3: Let us suppose that the examined member corresponds to pixel (x_i, y_i) of the original image A. The values $\delta\tilde{\theta}_{CL}^{(x_i,y_i)}$, $\tilde{\theta}_{CM}^{(x_i,y_i)}$ and $\tilde{\rho}_{CM}^{(x_i,y_i)}$ are given by Eqs. (14), (16) and (17), respectively, and describe the peak position of the curve. The extreme left peak element $[\tilde{\theta}_{CM}^{(x_i,\,y_i)} - \delta\tilde{\theta}_{CL}^{(x_i,\,y_i)}, \tilde{\rho}_{CM}^{(x_i,y_i)}]$ of row $\tilde{\rho}_{CM}^{(x_i,y_i)}$of the accumulator C is examined. If this element has non-zero value then go to Step 4 else execute Step 2 with the next member.

Step 4: Since the value at $[\tilde{\theta}_{CM}^{(x_i,\,y_i)} - \delta\tilde{\theta}_{CL}^{(x_i,y_i)}, \tilde{\rho}_{CM}^{(x_i,y_i)}]$ is non-zero, the curve of pixel (x_i, y_i) had a contribution in array C during the direct HT, which means that point (x_i, y_i) in array A was equal to 1. In that case, the array A_{inv} is updated (i.e. its (x_i, y_i) point is set to 1) and the curve obtained from point (x_i, y_i) is removed from array C, i.e. all the points of C corresponding to this curve decrease their value by 1. Go to Step 2 to proceed with the next member.

Step 5: Let suppose that the examined member corresponds to pixel (x_i, y_i) of the original image A. The values $\delta\tilde{\theta}_{CR}^{(x_i,y_i)}$, $\tilde{\theta}_{CM}^{(x_i,y_i)}$ and $\tilde{\rho}_{CM}^{(x_i,y_i)}$ are given by Eqs. (15)–(17) respectively, and describe the peak of the curve. The furthest right peak element $[\tilde{\theta}_{CM}^{(x_i,y_i)} + \delta\tilde{\theta}_{CR}^{(x_i,y_i)}, \tilde{\rho}_{CM}^{(x_i,y_i)}]$ of row $\tilde{\rho}_{CM}^{(x_i,y_i)}$ of the accumulator C is checked. If this element has non-zero value then go to Step 6 else execute Step 2 with the next member.

Step 6: Since the value at $[\tilde{\theta}_{CM}^{(x_i,y_i)} + \delta\tilde{\theta}_{CR}^{(x_i,y_i)}, \tilde{\rho}_{CM}^{(x_i,y_i)}]$ is non-zero, the curve of pixel (x_i, y_i) had a contribution in array C during the direct HT, which means that point (x_i, y_i) in array A was equal to 1. In that case, the array A_{inv} is updated (i.e. its (x_i, y_i) point is set to 1) and the curve obtained from point (x_i, y_i) is removed from array C, i.e. all the points of C corresponding to this curve decrease their value by 1. Go to Step 2 to check the next member.

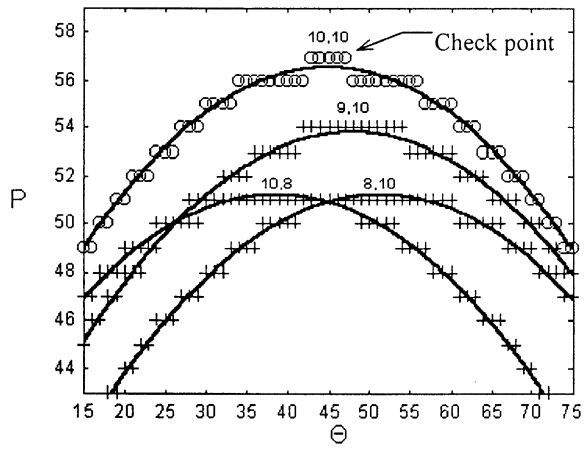


Fig. 4. Decomposition of the curve of the (10, 10) pixel.

At the end of the above procedure, the array C is empty and the restored image A_{inv} is the same with the original A. Fig. 4 depicts an example of the decomposition of the accumulator array C corresponding to a 10×10 pixel image and having the pixels (10,8), (8,10), (9,10) and (10,10) on. Specifically, the curve (10,10) is examined. This curve is the only member of the group at row $\tilde{\rho}_{CM} = 57$ and belongs to the right class. The check point value is non-zero, so the curve is removed and the element (10,10) of array A_{inv} is set to one. The procedure continues by checking the left class of the group at row $\tilde{\rho}_{CM} = 54$ which has no members, then the right class of that group which has one member i.e. the curve (9,10), etc.

Summarizing, in an accumulator array C, which has scale coefficients sf_θ and sf_ρ that satisfy the inversion conditions, we can fully reconstruct the original image A from C following the described decomposition procedure.

5. The window-based inverse Hough transform

Let us suppose that we have a $N \times N$ pixel image A and we want to apply a filtering procedure to find the edges that satisfy some specific conditions. According to IHT, we can calculate the direct HT of the entire image then apply the filter to the accumulator array values and finally use the IHT to extract the pixels of image A that satisfy the filter conditions. As it was already mentioned, if the coefficients sf_θ and sf_ρ satisfy the inversion conditions (18), (20) and (22), then we can reconstruct the original image, by applying a decomposition process in accumulator array C. Unfortunately, the larger the original image is, the higher the values sf_θ and sf_ρ are. This increases both the dimensions of the accumulator array and the required processing time. These requirements make the IHT prohitive if the image size is large enough.

It will be shown that by using our proposed method, which separates the image into windows, leads to significantly less computation time and reduced memory requirements. Thus, the inversion procedure can be applied to large size images.

5.1. Determination of the line parameters according to a point (x_k, y_k)

Before we describe the WIHT procedure, it is necessary to extract the relations that describe any straight line of the image from the origin of each image window. Let us suppose the image of Fig. 5, which contains a line with polar coordinates (ρ_o, θ_o). These coordinates are referred to the image origin, which is the bottom left corner of the image. The image is separated into k windows W_k. Let $k = 4$, as shown in Fig. 5. The bottom left corner of each window is denoted as (x_k, y_k), where $k = 1, 2, 3, 4$. Especially, for window W_1 the parameters values of the line are the same, that is $(\theta_o, \rho_o) = (\theta_1, \rho_1)$.

The general line equation gives

$$Ax + By + C = 0. \tag{23}$$

Also

$$\rho_o = x \cos\theta_o + y \sin\theta_o. \tag{24}$$

Thus

$$A = \cos\theta_o \tag{25}$$

$$B = \sin\theta_o \tag{26}$$

$$C = -\rho_o \tag{27}$$

The distance of a point (x_k, y_k) from the line (θ_o, ρ_o) is given by the relation

$$\rho_k = \frac{|Ax_k + By_k + C|}{\sqrt{A^2 + B^2}}$$

$$\Rightarrow \rho_k = |x_k \cos\theta_o + y_k \sin\theta_o - \rho_o|. \tag{28}$$

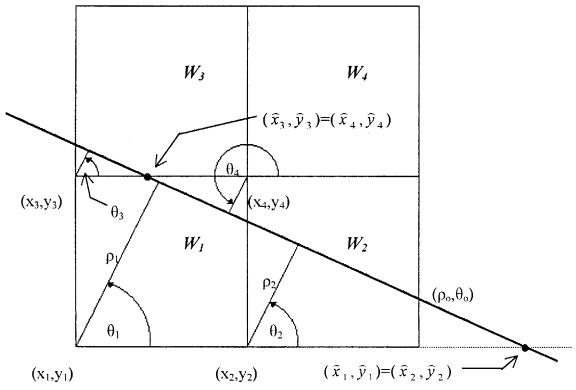


Fig. 5. The line parameters (ρ, θ) according to the origin of the windows.

The intersection point $(\hat{x}_k, \hat{y}_k)$ of line (θ_o, ρ_o) and axis $y = y_k$ is given by the relations

$$\left.\begin{aligned} A\hat{x}_k + B\hat{y}_k + C = 0 \\ \hat{y}_k = y_k \end{aligned}\right\} \Rightarrow \begin{cases} \hat{x}_k = -\dfrac{B\hat{y}_k + C}{A}, \\ \hat{y}_k = y_k. \end{cases} \tag{29}$$

For the definition of θ_k there are the following cases:

- If $A = 0$ then $\theta_o = \pm 90^\circ$ and the line is parallel to axis x, so

$$\theta_k = \begin{cases} \theta_o, & \rho_k \leqslant y_k, \\ \theta_o + 180, & \rho_k > y_k. \end{cases} \tag{30}$$

- If $A \neq 0$ and $\theta_o \in (-90, 90)$ then

$$\theta_k = \begin{cases} \theta_o, & \hat{x}_k \geqslant x_k, \\ \theta_o + 180, & \hat{x}_k < x_k. \end{cases} \tag{31}$$

- If $A \neq 0$ and $\theta_o < -90$ or $\theta_o > 90$ then

$$\theta_k = \begin{cases} \theta_o, & \hat{x}_k < x_k, \\ \theta_o + 180, & \hat{x}_k \geqslant x_k. \end{cases} \tag{32}$$

Finally, the value of θ_k is eliminated between $(-180$ and $180)$:

$$\theta_k = \begin{cases} \theta_k, & \theta_k \leqslant 180, \\ \theta_k - 360, & \theta_k > 180. \end{cases} \tag{33}$$

From Fig. 5 we can notice that by checking the values of θ_k and ρ_k we can find out if line (θ_o, ρ_o) passes through the window W_k. The values of θ_k must be in the range $(-90, 180)$, while ρ_k must be in the range $(0, \sqrt{2}\, S_w)$, where S_w denotes the dimension of window W_k. So, in case of window W_4, $\theta_4 \notin (-90, 180)$ which means that there are no points in window W_4 that belong to line (θ_o, ρ_o).

Concluding, using Eqs. (28)–(33) we can determine the parameters of line (θ_o, ρ_o) according to the origin of each window W_k.

5.2. The WIHT algorithm

To apply the WIHT algorithm, the $N \times N$ pixel image A must first split into windows of size $S_w \times S_w$ where $S_w \in G$ and G is the set of the integer dividers of N. The total number of these windows is $W_{sum} = (N/S_w)^2$.

For a window of size $S_w \times S_w$ the coefficients sf_θ, και, sf_ρ are determined from the inversion conditions. Thus, we can calculate the HT of each window W_k with $k = 1, \ldots, W_{sum}$, filter the values of accumulator array and using the inversion process extract only the pixels of window W_k that satisfy the filter conditions. Unfortunately, the filter conditions refer to the whole image A. In other words, the filter limits values $(\theta_{\min}, \theta_{\max})$ and

$(\rho_{\min}, \rho_{\max})$ as well as the threshold value T cannot be applied directly to the accumulator arrays that corresponds to each window W_k. To solve this problem the following two-phase procedure is introduced:

- *the control phase*: where using the direct HT for every window W_k we collect information about the pixel distribution in the whole image A, and
- *the decomposition phase*: where using the IHT in each window W_k we find the pixels of image A that satisfy the filter conditions.

These two phases are analyzed in the next:

Control phase

Step 1: Specify the filter parameters. These parameters concern the regions $[\theta_{\min}, \theta_{\max}]$ for θ_o the regions $[\rho_{\min}, \rho_{\max}]$ for ρ_o and the threshold value T.

Step 2: For every window W_k calculate the direct HT. The scale coefficients sf_θ and sf_ρ of the accumulator arrays C_k are defined using the inversion conditions or can be taken from Table 1.

Step 3: For every line (θ_o, ρ_o) with

$$\theta_o \in [\theta_{\min} \dots \theta_{\max}] \tag{34}$$

and

$$\rho_o \in [\rho_{\min} \dots \rho_{\max}], \tag{35}$$

define the values $(\theta_o^{(k)}, \theta_o^{(k)})$ from Eq. (28)–(33). These values specify the line (θ_o, ρ_o) according to the points (x_k, y_k), which is the origin of window W_k. To ensure that line $(\theta_o^{(k)}, \rho_o^{(k)})$ passes through window W_k, check the values $\theta_o^{(k)}$ and $\rho_o^{(k)}$ if they belong to the range $(-90, 180)$ and $(0, \sqrt{2}S_w)$, respectively. If any of them is out of range then go to Step 3 and proceed with the next line (θ_o, ρ_o) else go to Step 4.

If all lines have been checked, then go to Step 2 and repeat the procedure with the next window W_k.

Step 4: Calculate $\tilde{\theta}_C$ and $\tilde{\rho}_C$ using $\theta_o^{(k)}$ and $\rho_o^{(k)}$. Let V denote the value of element $(\tilde{\theta}_C, \tilde{\rho}_C)$ of the accumulator array C_k. If this value is non-zero, then there exists at least one pixel in window W_k that belongs to line (θ_o, ρ_o). So, if $V > 0$ then go to Step 5, else go to Step 3.

Step 5: Update array R which holds the pixels of the entire image that belong to line (θ_o, ρ_o):

$$R(\theta_o, \rho_o)_{new} = R(\theta_o, \rho_o)_{old} + V. \tag{36}$$

Go to Step 3 to continue with the next line (θ_o, ρ_o).

When all the lines (θ_o, ρ_o) are examined for every window W_k then array R holds the number of pixels of all windows that belong to the lines (θ_o, ρ_o). It must be noticed that until now the threshold value T has not been used. The aim of the above process is to find the number of pixels, which belong to the lines (θ_o, ρ_o) as they are given by Eqs. (34) and (35). Thus, each element (θ_o, ρ_o) of array R shows the total number of pixels (of any window W_k) that belong to line (θ_o, ρ_o).

Decomposition phase

Let us consider an $N \times N$ pixel image A_f where the filter results will be studied. More specifically, we want to find only the pixels of the original image A that belong to lines that are determined by the filter regions $(\theta_{\min}, \theta_{\max})$ and $(\rho_{\min}, \rho_{\max})$. The number of pixels in each of these lines must be equal or greater than T.

After the control phase, the distribution of the pixels along the lines is known and the threshold value T can be applied to the elements of array R. Every value (θ_o, ρ_o) of the array R is compared to the threshold T. If it is equal or greater than T then the pixels of the original image that belong to line (θ_o, ρ_o) satisfy the filter conditions. Thus, the accumulator array C_k, that corresponds to window W_k, is decomposed using the IHT. Also, in the window W_k of array A_f only the pixels that belong to the lines with $R(\theta_o, \rho_o) \geqslant T$ are activated. Analytically the decomposition phase is as follows:

Step 1: Let array S consist of the elements of R such as

$$R(\theta_o, \rho_o) \geqslant T. \tag{37}$$

Store the values in array S with the form $S_\theta^{(i)} = \theta_o$ and $S_\rho^{(i)} = \rho_o$, where i is a counter that shows the number of lines that satisfy Eq. (37). Let us suppose that the total number of these lines is S_{sum}. Thus, S contains only those elements of R that satisfy the threshold value T. Thereinafter, the decomposition process of the accumulator array C_k of each window W_k is applied.

Step 2: For every W_k:

Check the elements of array S. The pixels of the lines of array S are going to be activated.

Step 3: For every (θ_o, ρ_o) where

$$\theta_o = S_\theta^{(i)} \text{ and } \rho_o = S_\rho^{(i)} \text{ with } i = 1..S_{sum}, \tag{38}$$

define the values $(\theta_o^{(k)}, \rho_o^{(k)})$ from Eqs. (28)–(33). These values specify a line (θ_o, ρ_o) according to the points (x_k, y_k) which is the origin of window W_k. Then check the values $\theta_o^{(k)}$ and $\rho_o^{(k)}$ if they are in the range $(-90, 180)$ and $(0, \sqrt{2}S_w)$, respectively. If any of them is out of range continue with Step 3 with the next line (θ_o, ρ_o), else go to Step 4.

Step 4: Calculate $\tilde{\theta}_C$ and $\tilde{\rho}_C$ using $\theta_o^{(k)}$ and $\rho_o^{(k)}$. Let V denote the value of element $(\tilde{\theta}_C, \tilde{\rho}_C)$ of the accumulator array C_k. If this value is non-zero, then there exist pixels of window W_k that pass through that point. Therefore, these pixels belong to line $(\theta_o^{(k)}, \rho_o^{(k)})$ and consequently belong to line (θ_o, ρ_o). So, if $V \neq 0$ then go to Step 5, otherwise return to Step 3.

Step 5: Place the element $(\tilde{\theta}_C, \tilde{\rho}_C)$ in an array F. These elements will be used during the decomposition of array C_k of window W_k. If all S_{sum} lines (θ_o, ρ_o) have been

checked, then go to Step 6, else go to Step 3 and check the next line (θ_o, ρ_o).

Step 6: Apply the decomposition process to accumulator array C_k as it is described in Section 3. However, there is a significant difference using the IHT with a filter value T. During the inversion process, the window W_k of image A_f is not immediately updated. Alternatively, for every removed curve from the accumulator array C_k we check if it passes through a point of array F. If so, the pixel of image A that corresponds to that curve belongs to line (θ_o, ρ_o) and the corresponding pixel of image A_f is activated. After the full decomposition of array C_k the active pixels of window W_k of image A_f belong to lines that satisfy the filter conditions which were applied to the original image.

Go to Step 2 and continue with next window W_k.

The two phases of WIHT can be summarized in the following two pseudocodes:

Control phase:

Set input image A of size $N \times N$
Define filter regions $(\theta_{\min}, \theta_{\max})$ and $(\rho_{\min}, \rho_{\max})$
Define filter threshold T
Set the range of valid values for $\theta = (-90 \ldots 180)$ and for $\rho = (0 \ldots \sqrt{2}S_w)$
Define G as the set of integer dividers of N
Select a window size $S_w \in G$ (usually $S_w \geqslant 10$)
$W_{sum} = (N/S_w)^2$ is the number of windows
Clear array R
Determine sf_θ and sf_ρ for window size S_w
For every $W_k, k = 1 \ldots W_{sum}$ **do**
  Calculate the direct HT of window W_k
  For every (θ_o, ρ_o) that belongs to filter limits **do**
    Calculate $(\theta_o^{(k)}, \rho_o^{(k)})$
    If values of $\theta_o^{(k)}$ and $\rho_o^{(k)}$ are valid **Then**
      Calculate $\tilde{\theta}_C$ and $\tilde{\rho}_C$
      $V = C_k[\tilde{\theta}_C, \tilde{\rho}_C]$
      $R(\theta_o, \rho_o)_{new} = R(\theta_o, \rho_o)_{old} + V$
    Endif
  Endfor
Endfor

Decomposition phase:

Clear array S
Add to array S the elements of array R that satisfy the filter threshold value T.
Denote them as $S_\theta^{(i)}$ and $S_\rho^{(i)}$ with $i = 1 \ldots S_{sum}$, where S_{sum} the total number of the above elements.
For every $W_k, k = 1 \ldots W_{sum}$ **do**
  Clear array F
  For $i = 1$ to S_{sum} **do**
    $\theta_o = S_\theta^{(i)}$ and $\rho_o = S_\rho^{(i)}$
    Calculate $(\theta_o^{(k)}, \rho_o^{(k)})$
    If values of $\theta_o^{(k)}$ and $\rho_o^{(k)}$ are valid **Then**
      Calculate $\tilde{\theta}_C$ and $\tilde{\rho}_C$
      $V = C_k[\tilde{\theta}_C, \tilde{\rho}_C]$
      If $V > 0$ **Then**
        Add point $(\tilde{\theta}_C, \tilde{\rho}_C)$ to array F
      Endif
    Endif
    Update window W_k in final image A_f by decomposing accumulator array C_k using array F.
  Endfor
Endfor

6. Experimental results

As it has been already mentioned, in contrast with the previous version of the IHT, the main benefits of the proposed method is the significant reduction of the memory requirements and computation time. Thus, comparing this method to the basic non-window bared form of the IHT, we notice that the larger the original image A is the higher the gain is. Figs. 6 and 7 show the memory required for several window sizes for two images of 150×150 pixels and 300×300 pixels, respectively. Fig. 8 shows the memory requirements of the IHT and the WIHT for several image sizes. It must be noted that the IHT values correspond to the case of applying the inversion procedure to the entire image. We can see that as the image size increases the memory gain using WIHT becomes higher.

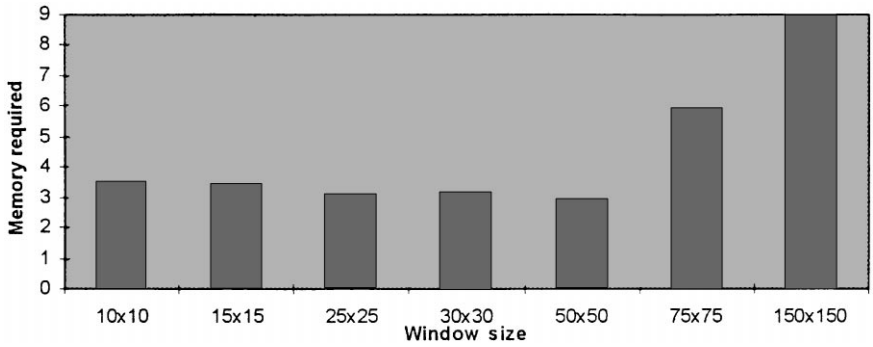


Fig. 6. Memory requirements for an 150×150 image.

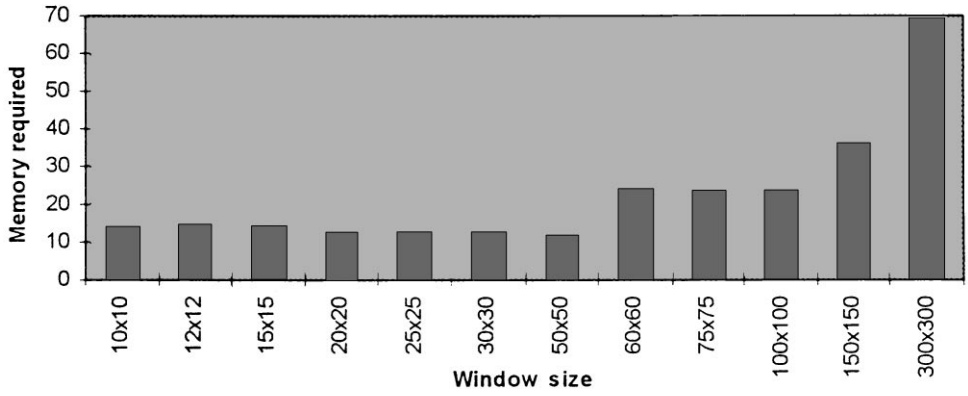


Fig. 7. Memory requirements for an 300×300 image.

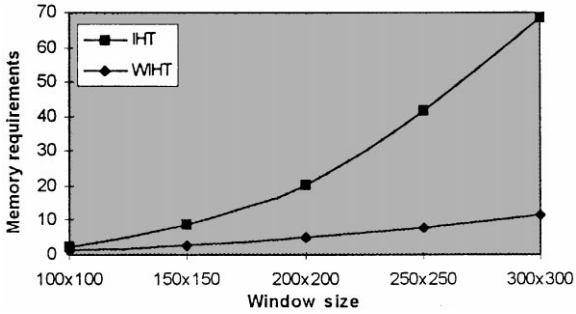


Fig. 8. Memory requirements of WIHT and IHT for several image sizes.

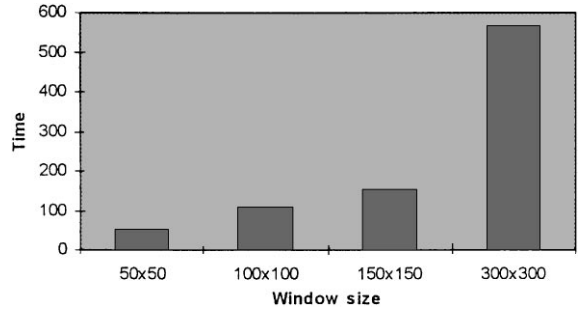


Fig. 10. Computation time for several window sizes of an 300×300 image.

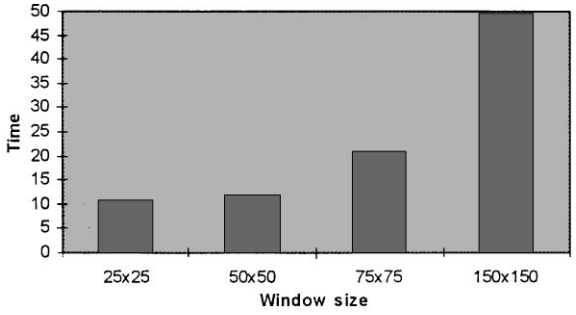


Fig. 9. Computation time for several window sizes of an 150×150 image.

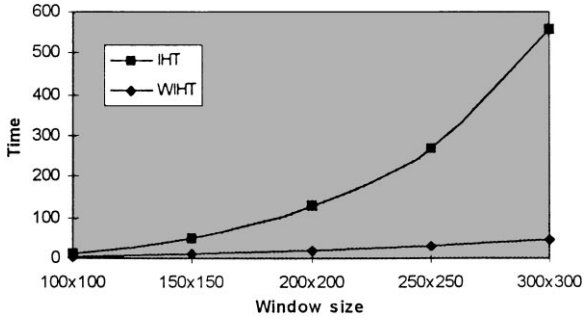


Fig. 11. Computation time of WIHT and IHT for several image sizes.

Figs. 9 and 10 show the computation time needed for the execution of the WIHT algorithm for several window sizes in two images of 150×150 pixels and 300×300 pixels, respectively. Fig. 11 shows the relation of the computation time of the IHT and the WIHT for several image sizes. Again, the IHT values correspond to the case of applying the inversion procedure to the entire image. As it can be noticed in Fig. 11, the required time for applying a filter in the HT space is significantly smaller using the WIHT than the IHT.

As a first example, the WIHT transform is applied to a grid image. The filter region values are for $\theta = [35, 50]$

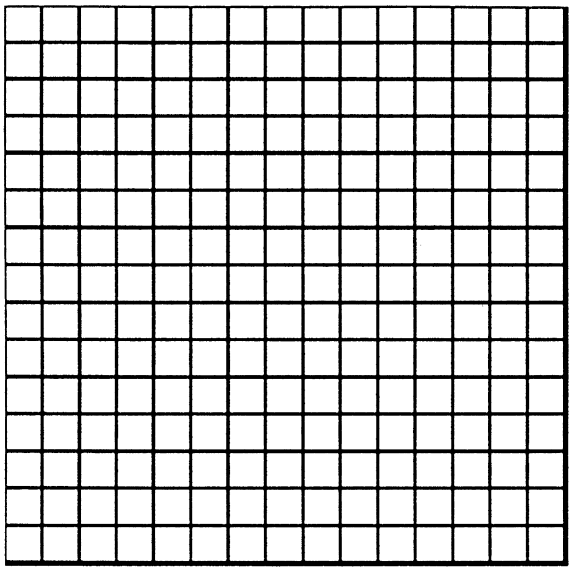


Fig. 12. The source image which depicts a bi-level grid.

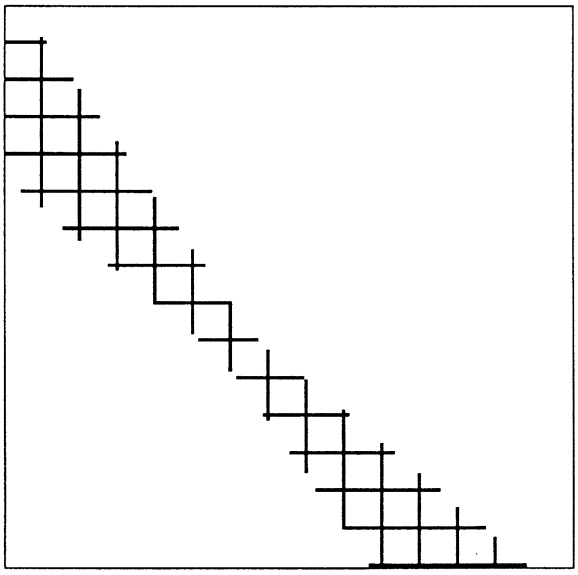


Fig. 13. Applying a filter with $35 \leqslant \theta \leqslant 50$ and $80 \leqslant \rho \leqslant 90$.

and for $\rho = [80, 90]$. No threshold value T is applied. The image is divided into nine windows. Fig. 12 depicts the original image while Fig. 13 shows the final results. It must be noticed that a significant advantage of the IHT and the WIHT is that in the resulting image, the edges that satisfy the filter conditions, have all their pixels precisely placed in the correct positions as these appear in the original image.

Fig. 14. The source bi-level image.

The second example demonstrates the effects of the application of only threshold conditions. Fig. 14 depicts the used bi-level image. Fig. 15 shows the results of the application of several threshold values. The filtering conditions are $\theta = 0$ and any value for ρ. Similarly, Fig. 16 shows the results of the application of several threshold values for $\theta = 90$ and any ρ. Fig. 17 shows the results of applying two threshold values for $\theta = 45$ and any ρ.

The original gray-level image for the last example is shown in Fig. 18. Both region values for θ, ρ and threshold values are applied. Fig. 19 depicts the binary form of this image after the application of the well-known Sobel edge extraction procedure. Using the WIHT a filter is applied to the HT space. Figs. 20–23 depict several edge extraction results for the filtering conditions given in Table 2.

It is noticed that in the above examples, in order to apply the WIHT, the original images was divided into nine (3×3) windows.

7. Conclusions

Hough transform is a widely used technique in the areas of image analysis and recognition. However, until recently no effective algorithm reported for inverting the Hough space. Clearly, an IHT technique permits the exact reconstruction of the original image pixel by pixel. In this paper, a new window-based method is proposed for the inversion of the HT space. The proposed technique divides the original image to rectangle windows and then applies the IHT algorithm to each of them. Thus, the inversion procedure can be applied to large-size images. The proposed technique is suitable for edge extraction and filtering. In this case, a merging process is applied to the final stage that merges the extracted edges according to the predefined filtering requirements. It is noted that the extracted edges have all of their pixels in the correct positions as they appear in the original image.

The WIHT algorithm is always applicable provided that some scaling conditions are fulfilled for each of the

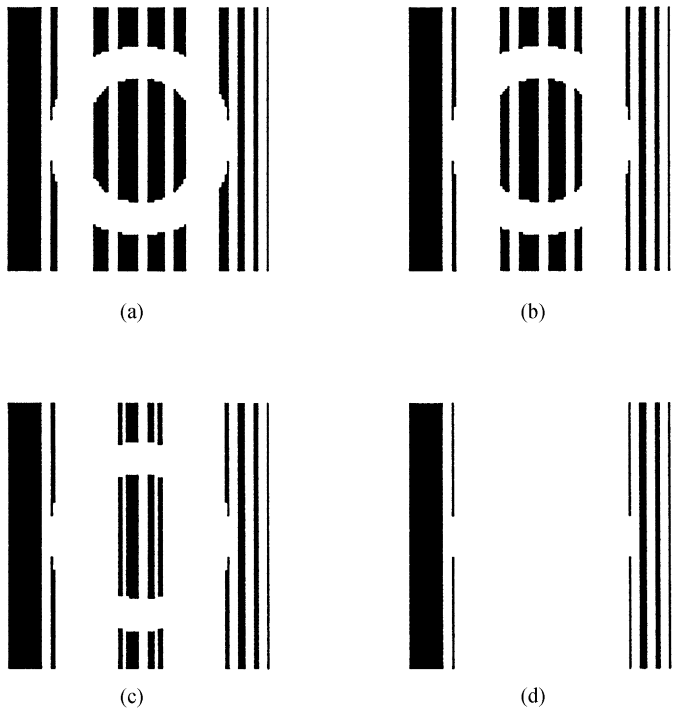


Fig. 15. Applying several threshold values for $\theta = 0$ and (a) $T = 65$, (b) $T = 70$, (c) $T = 75$, and (d) $T = 80$.

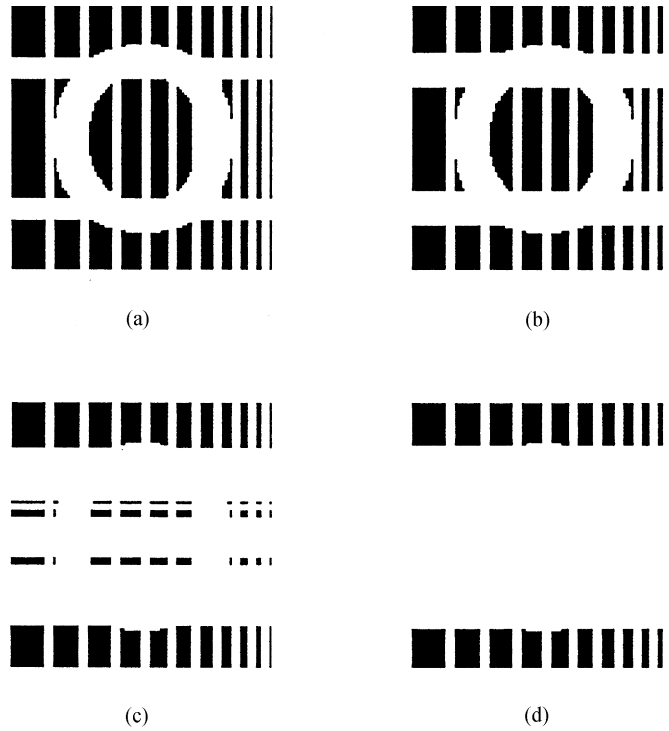
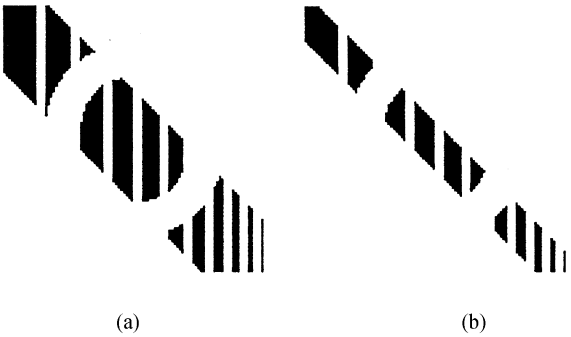


Fig. 16. Applying several threshold values for $\theta = 90$ and (a) $T = 40$, (b) $T = 45$, (c) $T = 50$, and (d) $T = 55$.



(a) (b)

Fig. 17. Applying several threshold values for $\theta = 45$ and (a) $T = 40$, and (b) $T = 50$.

Fig. 18. The original gray-level image.

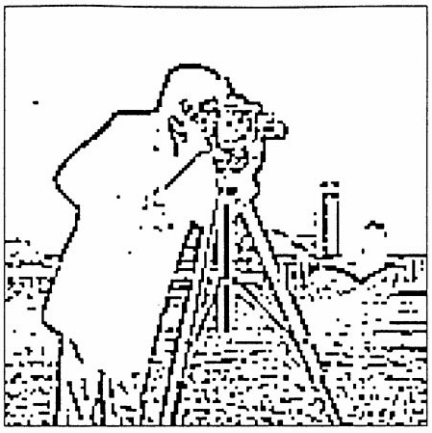


Fig. 19. The image after the application of the Sobel filter.

small size windows. These necessary inversion conditions are referred to the scale coefficients sf_θ and sf_ρ which define the size of the Hough space.

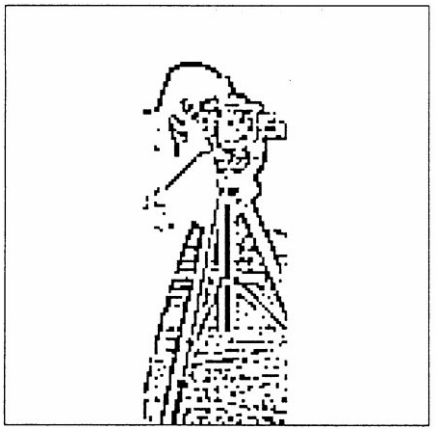


Fig. 20. Applying the filter conditions: $-10 \leqslant \theta \leqslant 10$, $50 \leqslant \rho \leqslant 100$ and $T = 10$.

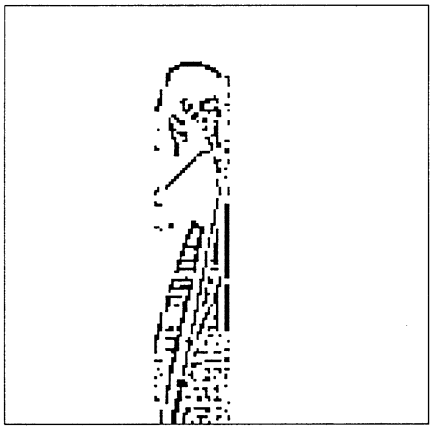


Fig. 21. Applying the filter conditions: $-20 \leqslant \theta \leqslant 10$, $50 \leqslant \rho \leqslant 80$ and $T = 20$.

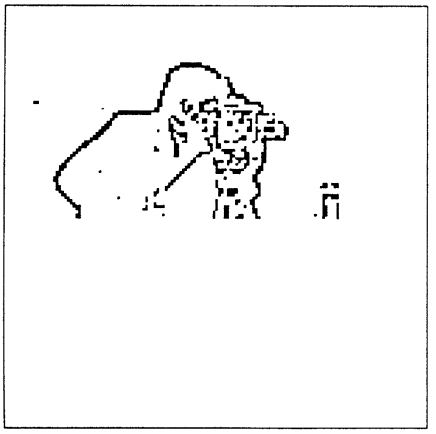


Fig. 22. Applying the filter conditions: $\theta = 90$, $75 \leqslant \rho \leqslant 140$ and $T = 3$.

Fig. 23. Applying the filter conditions: $10 \leqslant \theta \leqslant 30$, $100 \leqslant \rho \leqslant 120$ and $T = 5$.

Table 2
Filtering conditions

	Filtering parameter values	Fig.
1	$-10 \leqslant \theta \leqslant 10$, $50 \leqslant \rho \leqslant 100$, $T = 10$	Fig. 20
2	$-20 \leqslant \theta \leqslant 10$, $50 \leqslant \rho \leqslant 80$, $T = 20$	Fig. 21
3	$\theta = 90$, $75 \leqslant \rho \leqslant 140$, $T = 3$	Fig. 22
4	$10 \leqslant \theta \leqslant 30$, $100 \leqslant \rho \leqslant 120$, $T = 5$	Fig. 23

The WIHT algorithm was extensively tested with several images. The experimental results confirm its applicability.

References

[1] P.V.C. Hough, Methods and means for recognizing complex patterns, U.S. Patent 3069654, 1962.

[2] M. Atiquzzaman, Multiresolution Hough transform-An efficient method of Detecting Patterns in Images, IEEE Pattern Anal. Mach. Intell. 14 (1992) 1090–1095.

[3] N. Kiryati, Y. Eldar, A.M. Bruckstein, A probabilistic Hough transform, Pattern Recognition 24 (1991) 303–316.

[4] L. Xu, E. Oja, P. Kultanen, A new curve detection method: randomized Hough transform, Pattern Recognition Lett. 11 (1990) 331–338.

[5] B. Gatos, S.J. Perantonis, N. Papamarkos, Accelerated Hough transform using rectangular image decomposition, Electron. Lett. 32 (1996) 730–732.

[6] D. Ben-Tzvi, V.F. Leavers, M.B. Sandler, A dynamic combinatorial Hough transform, Proceedings of the fifth International Conference Image Analysis, 1990, pp. 152–159.

[7] Y. Zhang, R. Webber, A windowing approach to detecting line segments using Hough transform, Pattern Recognition 29 (1996) 255–265.

[8] S.J. Perantonis, B. Gatos, and N. Papamarkos, Block decomposition and segmentation for fast Hough transform evaluation, Pattern Recognition 32 (1999) 811–824.

[9] T. Van Veen, F. Groen, Discretisation errors in Hough transform, Pattern Recognition 14 (1981) 137–145.

[10] W. Niblack, D. Petrovic, On improving the accuracy of the Hough transform: theory, simulations and experiments, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1988, pp. 574–579.

[11] J. Illingworth, J. Kittler, The adaptive Hough transform, IEEE Trans. Pattern Anal. Mach. Intell. 9 (1987) 690–698.

[12] P. Palmer, J. Kittler, M. Petrou, Using focus of attention with the Hough transform for accurate line parameter estimation, Pattern Recognition 27 (1994) 1127–1134.

[13] D. Leung, L. Lam, W. Lam, Diagonal quantization of the Hough transform, Pattern Recogn. Lett. 14 (1993) 181–189.

[14] W. Lam, L. Lam, K. Yuen, D. Leung, An analysis on quantizing the Hough space, Pattern Recogn. Lett. 15 (1994) 1127–1135.

[15] R.O. Duda, P.E. Hart, Use of the Hough transform to detect lines and curves in pictures, Commun. ACM 15 (1972) 11–15.

[16] D.H. Ballard, Generalizing the Hough transform to detect arbitrary shapes, Pattern Recognition 13 (1981) 111–122.

[17] V. Chatzis, I. Pitas, Fuzzy cell Hough transform for curve detection, Pattern Recognition 30 (1997) 2031–2042.

[18] A.L. Kesidis, N. Papamarkos, On the inverse Hough transform and its applications for edge detection and filtering, Proceedings CIMCA'99, Vienna, Austria, (1999) 48–53.

About the Author—NIKOS PAPAMARKOS was born in Alexandroupoli, Greece, in 1956. He received his Diploma Degree in Electrical and Mechanical Engineering from the University of Thessaloniki, Thessaloniki, Greece, in 1979 and the Ph.D. Degree in Electrical Engineering in 1986, from the Democritus University of Thrace, Greece. From 1987 to 1990 Dr. Papamarkos was a Lecture, from 1990 to 1996 Assistant Professor in the Democritus University of Thrace where he is currently Associate Professor since 1996. During 1987 and 1992 he has also served as a Visiting Research Associate at the Georgia Institute of Technology, USA. His current research interests are in digital signal processing, filter design, optimization algorithm, image processing, pattern recognition and computer vision. Dr Nikos Papamarkos is a member of IEEE and a member of the Greek Technical Chamber.

About the Author—ANASTASIOS L. KESIDIS received the Diploma in Electrical and Computer Engineering from Democritus University of Thrace, Greece, in 1995. He is currently a research and teaching assistant and studying towards the Ph.D. degree at the Department of Electrical and Computer Engineering, Democritus University of Thrace. His research interests include nonlinear image processing and analysis, neural networks and pattern recognition. He is a member of the Institute of Electrical and Electronics Engineers (IEEE) and of the Technical Chamber of Greece.