Pattern Recognition Letters 11 (1990) 331-338
North-Holland

May 1990

A new curve detection method: Randomized Hough Transform (RHT)

Lei XU*, Erkki OJA and Pekka KULTANEN
Lappeenranta University of Technology, Department of Information Technology, Box 20, 53851 Lappeenranta, Finland

Received 9 October 1989
Revised 17 November 1989

Abstract: A new method is proposed for curve detection. For a curve with n parameters, instead of transforming one pixel into a hypersurface of the n-D parameter space as the HT and its variants do, we randomly pick n pixels and map them into one point in the parameter space. In comparison with the HT and its variants, our new method has the advantages of small storage, high speed, infinite parameter space and arbitrarily high resolution. The preliminary experiments have shown that the new method is quite effective.

Key words: Curve detection, Hough transform, random mapping, global feature extraction.

1. Introduction

Detecting curves (straight line, circle, ellipse, etc.) from an image is one of the basic tasks in computer vision. Presently, the commonly used curve detecting methods are the Hough Transform (HT) and its variants [1].

These methods consist of three basic steps: (1) one pixel of the image space is transformed into a parameterized curve (or a surface, depending on the number of parameters) of the parameter space; (2) an accumulator with a cell array is laid on the parameter space, and each image pixel gives one score to the cells lying on its transformed curve; (3) finally, a cell with the local maximum of scores is selected, and its parameter coordinates are used to represent a curve segment in the image space. These three steps make the method feasible, but also give rise to the two following difficulties:

(1) For one image pixel, not only the correct cell (the one that represents the curve on which the pixel lies), but also many other cells are accumulated. This brings difficulties in finding the local maxima in the accumulator array. Risse [2] proposed a method which can partly overcome such difficulties, by extracting the first maximum and then eliminating all the scores contributed on the accumulator by the pixels lying on the curve which is represented by the first maximum cell.

(2) More importantly, the accumulator array is practically predefined by windowing and sampling the parameter space in a heuristic way. Generally, to detect curves within a wide scope of images, we need a window of large size; and to detect curves with high accuracy we need a good parameter resolution. The two needs, in turn, lead to the need of a large array taking up much computing time and storage. Usually, without some prior knowledge about the image, it is not easy to appropriately predefine the accumulator array. However, an inappropriate array will lead to at least one of the following problems: (a) failure to detect some specific curves, (b) difficulties in finding local maxima, (c) low accuracy, (d) large storage and

This work was supported by Tekes Grant 4196/1988 under Finsoft project.
* Permanent address: Dept. of Mathematics, Peking University, P.R. China.

(e) low speed. Some efforts to deal with these problems are parameter space decomposition (e.g., [3,4]) and parallel architecture.

Based on a well-known neural network method—the Kohonen map, an extended self-organizing map for curve detection was proposed in [5] by the first two authors of this paper. The method gives some advantages like high accuracy, low storage, and an unbounded extent of the parameter space. Furthermore, by using some key points of the extended map, paper [5] also proposed the basic idea of a new non-neural-network curve detecting method. In this paper, we develop this idea further and propose a novel HT-like method (which we call the Randomized Hough Transform, RHT) for detecting curves from a binary image. Such images may be obtained from grey-level images by some conventional techniques (e.g., the Canny operator). The suggested method can overcome the above mentioned difficulties; it shares with the extended self-organizing map the advantages of high accuracy, low storage and infinite parameter space, and also reduces the computation time drastically as compared with the standard HT.

2. The Randomized Hough Transform

2.1. The basic ideas

For convenience, here we take a straight line as an example to introduce the basic ideas of RHT. and then in Section 2.2 we will give the general procedure of RHT for detecting various curves. Throughout, assume that the original image is binary with coordinates (x, y).

For a straight line expressed by

$$y = \alpha_1 x + \alpha_2, \tag{1}$$

the conventional HT transforms a point (x_1, y_1) into a line $\alpha_2 = y_1 - \alpha_1 x_1$ of the (α_1, α_2) parameter space. However, using two points (x_1, y_1), (x_2, y_2), we can map them into a point (a_1, a_2)[1] of the parameter space simply by solving the following joint equations:

$$\begin{cases} y_1 = \alpha_1 x_1 + \alpha_2, \\ y_2 = \alpha_1 x_2 + \alpha_2. \end{cases} \tag{2}$$

The basic ideas stem just from this key point. First, we put all the bright or 'on' points of the binary image into a pixel data set D. Then, we implement an iterative procedure. At each step of the procedure, we randomly take two points $d_1 = (x_1, y_1)$, $d_2 = (x_2, y_2)$, $d_1 \neq d_2$ out of the set D in such a way that all points of D have an equal probability to be taken as d_1, and then all points of $D - \{d_1\}$ have an equal probability to be taken as d_2. Then we use eq. (2) to solve a parameter point $p_i = [a_1(i), a_2(i)]$ and put this point into a parameter data set P. After a certain number of steps, it is not difficult to see that there will be several p_i points accumulated at the point (a_1, a_2), if the image space contains a line with parameters (a_1, a_2). As a result, by finding out those accumulated points in the set P, we can detect all the lines contained in the image space.

In practice, the search of the accumulated points can be realized as follows. Assume that each element or cell in the parameter set P has both a parameter pair or vector and an integer value called the score. When a point p_i is obtained, we search and check whether there is an element in P with the same parameter pair as p_i. If there is such an element, then we increase its score by one. If none is found, then we create a new cell with parameters equal to p_i and score equal to one and insert it into P as a new element. Finally, those elements which have scores larger than a threshold n_t (which is a very small number, e.g., 2 or 3) are regarded as the accumulated cells.

Furthermore, some improvements on this basic procedure can be made:

(1) The elements of P can be ordered according to their a_1, a_2 values (e.g., the tree structure shown in Figure 1) so that the search time can be minimized.

(2) Given a tolerance δ, if there is an element in P with the distance of its parameter vector to p_i smaller than δ, then the element is regarded the same as p_i. Its parameters are updated by the average of its old parameters and p_i, and its score

[1] In this paper, we use a (a_i) to denote a specific value of the variable α (α_i) resp.

is increased by one. This will reduce the storage.

(3) It is possible to detect the accumulated cells one by one. Once there is an element p with its score larger than the threshold n_t, we take out of D all the d_i points lying on the line represented by p, and reset set $P=null$. As a result, the storage can be greatly reduced, and it will also become easier to find the accumulated cells.

In summary, the basic idea of the RHT consists of the three parts:

(1) At each step, randomly pick two or n (for an n parameter curve) pixels and map them into one point (or sometimes more than one point for an n parameter curve, see Section 2.2) in the parameter space.

(2) Use a set P with each element containing both a real valued vector and an integer score to implicitly represent the parameter space, and update set P at each step by the point mapped from the randomly picked pixels.

(3) Find the accumulated cells in P containing the parameters of the detected curves.

Since the method is mathematically a stochastic method and it still retains some features of the Hough Transform, we roughly call it the Randomized Hough Transform (RHT). However, in a strict mathematical sense, mapping the randomly picked pixels into the parameter space is not a transform because it is irreversible.

2.2. The general RHT procedure

It is not difficult to see that the method given in Section 2.1 can be directly used to detect the lines represented by

$$\alpha_1 x+\alpha_2 y-1=0 \tag{3a}$$

$$\alpha_1 x+\alpha_2 y-\alpha_3=0 \tag{3b}$$

and other curves with expressions linear with respect to the parameters

$$\alpha_1 z_1+\alpha_2 z_2+\cdots+\alpha_n z_n+z_0=0 \tag{4a}$$

where $z_i, i=0,\ldots,n$ only depend on x, y and constants. An example is the quadratic curve expressed by

$$f(A,x,y)=a_1x^2+a_2xy+a_3y^2 + a_4x+a_5y+c=0 \tag{4b}$$

where c is a constant and $A=\{a_1,a_2,a_3,a_4,a_5\}$ denotes the parameter set.

For these curves, the n parameters can be explicitly solved from the randomly picked n pixels by n joint linear equations. However, for curves expressed by equations which are nonlinear with respect to the parameters, the RHT cannot be

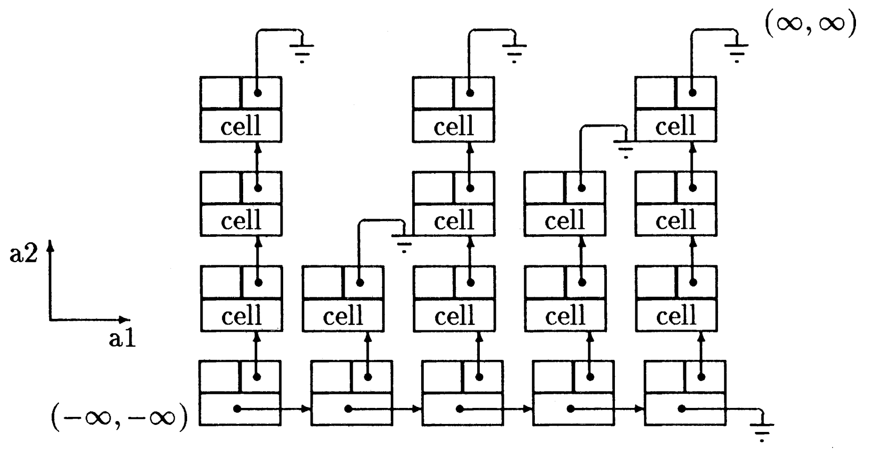

Figure 1. A particular example of the dynamic data structure for the ordered search in Set P (where each cell stores a $p=[a_1,a_2]$ point and its score, and the cells are ordered according to the values of a_1 and a_2).

directly used. The circle expressed by eq. (5) is an exception:

$$(\alpha_1 - x)^2 + (\alpha_2 - y)^2 = \alpha_3^2, \tag{5a}$$

$$(\alpha_1 - x)^2 + (\alpha_2 - y)^2 = R^2. \tag{5b}$$

We assume that in (5a), all the three parameters are unknown, while in (5b) the radius R is known. For eq. (5a), using three randomly picked points (x_1, y_1), (x_2, y_2), (x_3, y_3), the three parameters are not easily solved directly. However, the first two parameters α_1, α_2 can be solved by any two of the following three equations:

$$l_i:\ \left(\alpha_2 - \frac{y_r + y_i}{2}\right) = \left(\frac{y_r - y_i}{x_r - x_i}\right)\left(\alpha_1 - \frac{x_r + x_i}{2}\right),$$
$$r = \mathrm{mod}_3[i+1], \quad i = 1, 2, 3 \tag{6}$$

where if $m = 3k + r$ and k, r are integers, then $\mathrm{mod}_3[m] = r$. There l_i is the midperpendicular of the line segment from (x_i, y_i) to (x_r, y_r), and any two of such midperpendiculars should cross at the center of a circle. Using eq. (5a), we can then obtain the third parameter α_3 directly. As for eq. (5b), i.e., a circle with the known radius R, two pixels will define two parameter points $p_1 = (\alpha_1, \alpha_2), p_2 = (\alpha_1', \alpha_2')$, and we must insert cells corresponding to both p_1, p_2 into the set P.

It should be mentioned that for other curves expressed by nonlinear equations with respect to the parameters, it may still be possible to use the RHT by some other method, e.g., by introducing additional parameters. We will describe the details elsewhere.

By summing up both the above discussion and the one in Section 2.1, for a curve expressed by an n parameter equation $f(\alpha_1, \ldots, \alpha_n, x, y) = 0$, we give the general RHT procedure as follows:

Step 1. Scan a binary image and put the coordinates $d_i = (x_i, y_i)$ of all the 'on' pixels into the pixel data set D. Then, initialize a parameter data set $P = null$ and $k = 0$.

Step 2. Randomly pick n points $d_1, \ldots, d_n$ out of D in such a way that all points of D have an equal probability to be taken as d_1, then all points of $D - \{d_1\}$ have an equal probability to be taken as d_2, etc.; finally, all points of $D - \{d_1, d_2, \ldots, d_{n-1}\}$ have an equal probability to be taken as d_n.

Step 3. Solve n joint equations

$$f(\alpha_1, \ldots, \alpha_n, x_i, y_i) = 0, \quad i = 1, \ldots, n$$

to determine one parameter point $p = (\alpha_1, \ldots, \alpha_n)$ (or it may be more than one when $f(\cdot)$ is nonlinear with respect to its parameters).

Step 4. Search among set P for an element p_c such that $p_c = p$ (or $\|p_c - p\| < \delta$, with δ being a given tolerance). If found, goto Step 6, otherwise, goto Step 5.

Step 5. Attach to p an accumulating cell with score one, and insert it into set P as a new element which is located according to the partial order of a given dynamic structure (e.g., for $n = 2$, see Figure 1). Goto Step 7.

Step 6. Increase the score of the accumulating cell of p_c by one, and then check whether the increased score is smaller than a given threshold n_t (e.g., $n_t = 2, 3$); if yes, goto Step 7, otherwise, goto Step 8.

Step 7. $k := k + 1$; if $k > k_{\max}$ then stop, otherwise, goto Step 2.

Step 8. Take p_c as the parameters of a possible curve, and take out of D all the pixels lying on the curve. If there are m_{p^c} such pixels and $m_{p^c} > m_{\min}$, then goto Step 9; otherwise, p_c represents a false curve, return the m_{p^c} pixels into set D, then take p_c and its accumulating cell out of set P (i.e., $P := P - \{p_c\}$), and goto Step 2.

Step 9. A curve represented by p_c has been detected. Reset $P = null$ and $k = 0$, and goto Step 2.

Remarks. (1) The tolerance δ is introduced to further reduce the storage and adjust the resolution of the parameter space. When $\delta = 0$, the RHT has the highest resolution. The larger δ is, the lower is the resolution but the less storage is used. It should be pointed out that even when $\delta = 0$ the storage used by the RHT is already greatly reduced in comparison with that used by the conventional HT.

(2) After all the pixels of the curves have been taken out, the pixels remaining in set D are noise. In this case, the time required for a p cell to attain a score of n_t should become considerably longer. So, the procedure can be stopped by a suitable threshold $k_{\max}$. The theoretical analysis on how to appropriately choose $k_{\max}$ will be given in a

separate paper [6] where an alternative stopping criterion will also be given.

(3) For detecting lines, the minimal n_t could be 2 (e.g., as used in Section 3). The related detailed analysis will also be provided in [6].

(4) If there are less than $m_{\min}$ pixels lying on a curve, we consider the curve being a false one.

(5) Finally, it deserves to point out that the above procedure can be directly extended to detect surfaces in a 3D scene.

2.3. *The characteristics of the RHT*

Compared with the conventional HT and its variants, the RHT has the following advantages:

Infinite parameter space. Due to the predefined accumulator, the HT and its variants can only observe those curves whose parameters are within a finite window of the parameter space. In contrast, because P stores any parameter point mapped from pixels of the image space, the RHT can implicitly observe the whole extent of an infinite parameter space. There is no problem of predefining an appropriate window as in the case of the HT.

Arbitrarily high parameter resolution. The resolution of the HT is predefined by the grid size in discretizing the selected window of the

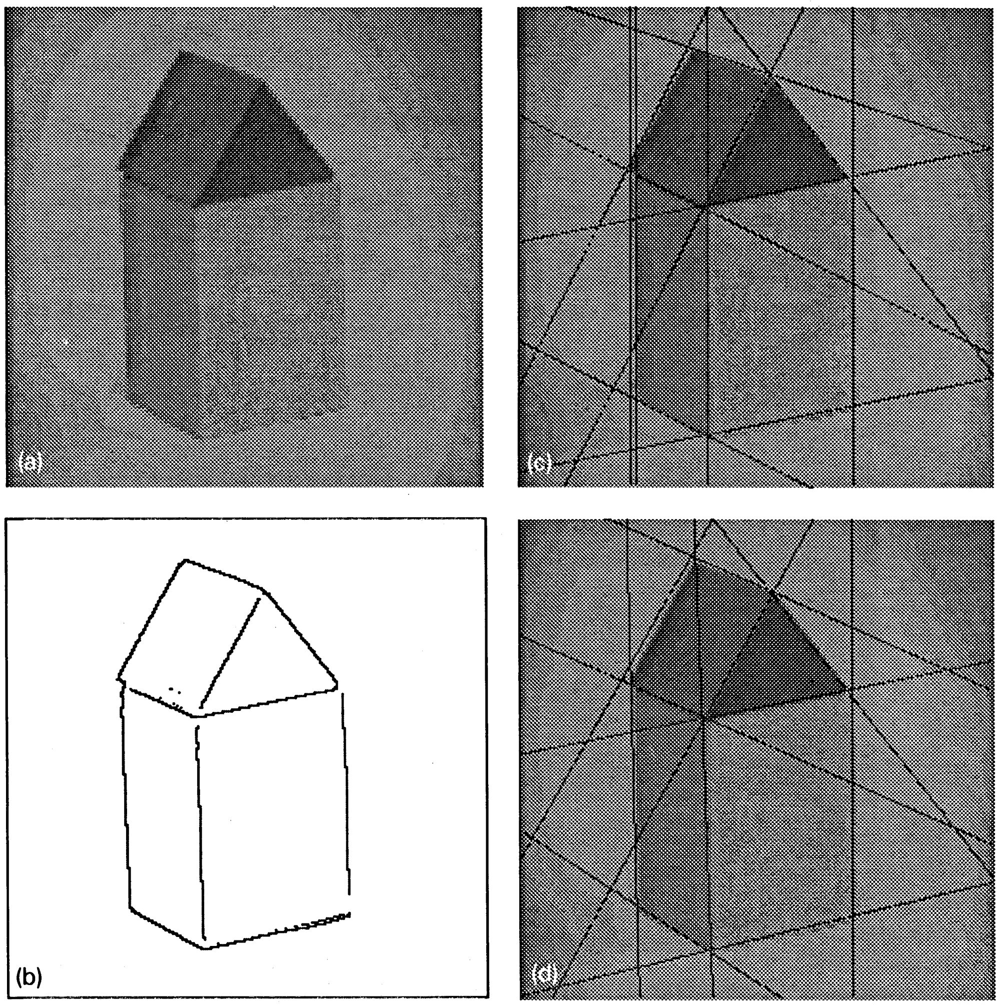

Figure 2. (a) The original grey picture. (b) The edge picture after preprocessing. (c) The result of the RHT (two lines, instead of one line in Figure 2d, were detected for the leftmost edge because of the higher resolution of RHT and the fact that the edge in Figure 2b actually consists of several staged line segments). (d) The result of the HT.

parameter space. It is constrained by the storage and computation time since a high resolution increases the number of array cells. In contrast, P stores the real value parameter points without discretization, so the RHT has inherently high resolution. In fact, by changing the tolerance δ in Step 4 from 0 to some chosen value, its resolution can be arbitrarily adjusted.

Small storage. The storage of the HT depends on the window size and resolution in the accumulator space. Large size and high resolution result in an accumulator array with a great number of cells. In contrast, due to the frequent resetting of set P, the storage of the RHT is always kept quite small. E.g., for the picture shown in Figure 2, the HT uses a 256×256 array while the RHT uses a storage of only around 30 cells (see Section 3).

High computation speed. For the HT, one pixel is transformed into a curve, a surface, or a hypersurface, and all the cells lying on the curve or surface should be accumulated. So the computing speed is constrained by the size of the accumulator array. In contrast, for the RHT, at each step, only one parameter point of P is updated which greatly cuts down the computing time. E.g., in the case of Figure 3, the computing time of the HT is 80 times larger than that of the RHT (see Section 3). It should be pointed out that especially for curves with $n > 2$ parameters the computing time of the RHT reduces much more drastically since in such cases the HT need to accumulate all the cells lying on a surface or hypersurface.

3. Computer experiments

3.1. Detecting straight lines

A grey image is shown in Figure 2a. After

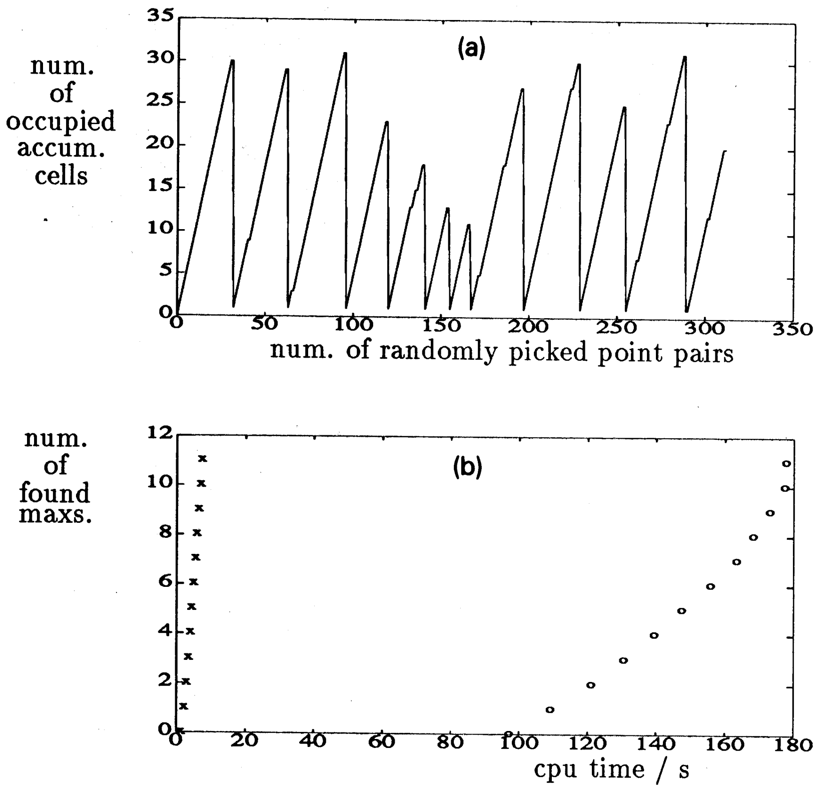

Figure 3. (a) The consumed storage of the RHT (each peak indicates where a line segment is detected and set P is reset to $P = null$). (b) A comparison between the consumed times of the RHT with x marks and of the HT with o marks (each point indicates where a line segment is detected).

preprocessing by the Canny operator and a thinning algorithm, a binary image is obtained in Figure 2b. Then eq. (1) is used for detecting line segments. Although for the HT this equation will constrain the scope and the accuracy of parameter α_1, for RHT there is no such problem due to the advantages of infinite parameter space and arbitrary resolution. The threshold n_t at Step 4 is here $n_t = 2$, and the dynamic structure of set P is given by Figure 1. The result of the RHT is given in Figure 2c superimposed on the original grey picture.

For comparison, the standard HT with line expression

$$\varrho = x\cos(\theta) + y\sin(\theta) \tag{7}$$

is also used on Figure 1a. The accumulator is a 256×256 array with constant sampling on a window

$$\varrho[0, \sqrt{2} \times 256], \quad \theta[-\pi/2, \pi/2],$$

and the recently proposed decremental method [2] is used to improve the extraction of local maxima. The result is shown in Figure 2d.

From Figures 2c and 2d, it can be observed that although a high sampling rate

$$\Delta\varrho = \sqrt{2} \text{ pixel/cell}, \qquad \Delta\theta = \pi/256/\text{cell}$$

is used to implement the HT for a high accuracy, the line segments of the RHT are still generally more accurate than those of the HT. Furthermore, Figure 3a shows that the maximum storage used by the RHT is around 30 cells which is a drastic reduction in comparison with 256×256. Figure 3b shows the comparison of the computing times used by the RHT (with x marks) and the HT (with o marks); again, a significant reduction by the RHT can be observed.

3.2. *Detecting circles*

A test binary image is shown in Figure 4a, and the circles are described by eq. (5a). In this case, the standard HT will transform one pixel into a surface, and all the cells on the surface have to be accumulated. Thus the computation is very expensive both in storage and in time, and it is also quite difficult to make an appropriate window on the 3-D parameter space. However, the RHT can work in this case without any difficulties. It has found all the circles as shown in Figure 4b with quite small computer time and storage.

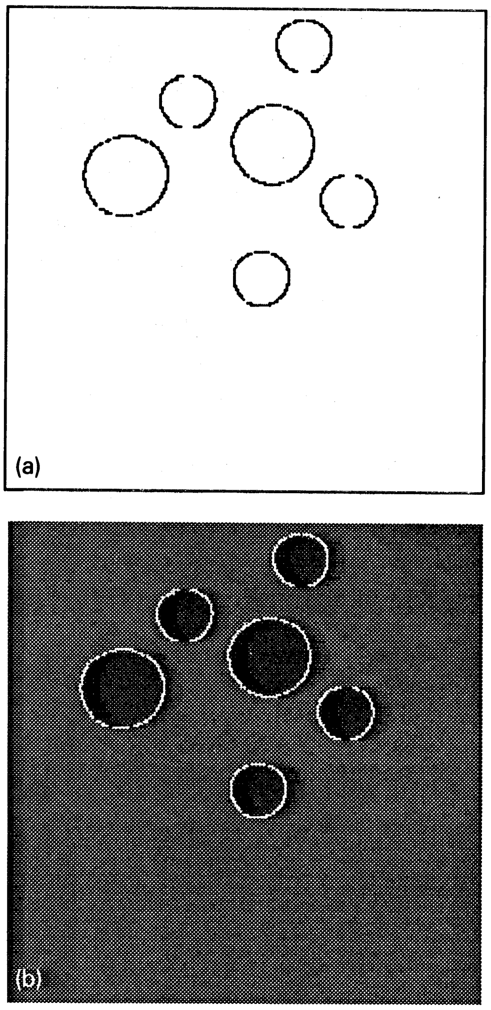

Figure 4. (a) The original edge picture. (b) The result of the RHT.

4. Conclusions

We have proposed a new technique for curve detection. For a curve expressed by an n parameters equation, instead of transforming one pixel into an $n-1$-dimensional hypersurface in the parameter space as the HT and its variants do, we randomly pick n pixels and map them into one point of the parameter space. In comparison with the HT and its variants, we have shown through analysis and experiments that our new method has advantages of small storage, high speed, infinite parameter space and arbitrarily high resolution. In addition, there is no difficulty of choosing an appropriate window and sampling for the accumulator and of finding local maxima. These two difficulties significantly influence the performance of the HT and its variants. While this paper has concentrated on showing the key ideas of the method and on giving some experimental results, the results can be confirmed by a theoretical analysis of the HT and the RHT. We will give the theoretical analysis and some extensions of the RHT elsewhere.

References

[1] Illingworth, J. and J. Kittler (1988). A survey of the Hough transform. *Computer Vision, Graphics, and Image Processing* 43, 221–238.

[2] Risse, T. (1989). Hough transformation for line recognition: Complexity of evidence accumulation and cluster detection. *Computer Vision, Graphics, and Image Processing* 46, 327–345.

[3] Illingworth, J. and J. Kittler (1987). The adaptive Hough transform. *IEEE Trans. Pattern Anal. Machine Intell.* 9, 690–698.

[4] Li, H., M.A. Lavin and R.J. LeMaster (1986). Fast Hough transform: A hierarchical approach. *Computer Vision, Graphics, and Image Processing* 36, 139–161.

[5] Xu, L. and E. Oja (1989). Extended self-organizing map for curve detection, submitted to *1990 IEEE Symposium on Circuits and Systems*, New Orleans, May 1–3, 1989.

[6] Xu, L., E. Oja and P. Kultanen. Randomized Hough transform: Theoretical analysis and extensions. In preparation.