Probabilistic and non-probabilistic Hough transforms: overview and comparisons

Heikki Kälviäinen*, Petri Hirvonen*, Lei Xu† and Erkki Oja‡

A new and efficient version of the Hough transform for curve detection, the Randomized Hough Transform (RHT), has been recently suggested. The RHT selects n pixels from an edge image by random sampling to solve n parameters of a curve and then accumulates only one cell in a parameter space. In this paper, the RHT is related to other recent developments of the Hough transform. Hough transform methods are divided into two categories: probabilistic and non-probabilistic methods. An overview of these variants is given. Some novel extensions of the RHT are proposed to improve the RHT for complex and noisy images. These new versions of the RHT, called the Dynamic RHT, and the Window RHT with its variants, use local information of the edge image. They apply the RHT process to a limited neighbourhood of edge pixels. Tests in line detection with synthetic and real-world images demonstrate the high speed and low memory usage of the new extensions, as compared both to the basic RHT and other versions of the Hough transform.

Keywords: Hough transform, curve detection, random mapping, global feature extraction

The Hough Transform (HT) is a popular method to extract global curve segments from an image[1]. The main bottlenecks of the HT are its computational complexity and storage requirements, and many attempts have been made to alleviate these problems. In recent years, the development has been rapid in the area. A new kind of approach is the set of methods, in this paper called *randomized* or *probabilistic Hough transforms*, reported by many authors[2-17]. All of them utilize random sampling of the edge points of an input image*. Moreover, some of them use many-to-one or converging mapping from the image space into the parameter space, and replace an accumulator array by a list structure. New *deterministic* or *non-probabilistic* approaches have also been suggested[18-28]. In this paper, an overview on the new methods is presented, and the algorithms are briefly introduced and compared.

Inspired by the Kohonen self-organizing map neural network[29], Xu *et al.*[3,30] introduced the Randomized Hough Transform (RHT). The mechanisms of random sampling of points and a many-to-one mapping or converging mapping from the image space to the parameter space was for the first time proposed in this method. The RHT overcomes many problems associated with the Standard Hough Transform (SHT). The RHT has been analysed elsewhere[3-6], and it has been applied to engineering drawing vectorization system[31], to document processing[6] and to motion detection[32-34]. However, the basic RHT may have problems with complex and noisy images. Four novel extensions of the RHT, called the Dynamic RHT (DRHT), the Random Window RHT (RWRHT), the Window RHT (WRHT), and its special version called the Connective

*Department of Information Technology, Lappeenranta University of Technology, P.O. Box 20, FIN-53851 Lappeenranta, Finland (email: Heikki.Kalviainen@lut.fi)
†Department of Computer Science, Chinese University of Hong Kong, Shatin, Hong Kong
‡Laboratory of Computer and Information Science, Helsinki University of Technology, Rakentajanaukio 2 C, FIN-02150, Espoo, Finland
Paper received: 21 October 1993; revised paper received: 31 August 1994

*For this reason, the term *randomized* ought to be preferred. The term *probabilistic* is more diffuse, covering also methods that, for example, model the accumulator space by statistical models. However, for reasons of conformity with current practice, we choose to use the terms *probabilistic* and *non-probabilistic* here.

RHT (CRHT), are proposed in this paper to alleviate these problems. They apply the RHT to a local neighbourhood of a randomly selected binary edge point. The methods are tested with both synthetic and real-world pictures, and compared to the SHT, the basic RHT and several other Hough transform algorithms with good results. The properties of the tested methods are discussed, and some conclusions are given.

RECENT VERSIONS OF THE HOUGH TRANSFORM

The Hough transform has been a popular research topic for many years[1,35]. It is a popular method to extract global features like straight lines, circles, ellipses, etc. from an image. This fundamental task of computer vision can be a significant part of a pattern recognition system; as an example, see Gerig[19]. The Standard Hough Transform (SHT) computation consists of three parts: (1) calculating the parameter values and accumulating the cells in the parameter space*; (2) finding the local maxima which represent curve segments in the input image; and (3) extracting the curve segments using the knowledge from the maximum positions.

The standard approach of the HT involves several computational difficulties: the HT process can be time and memory consuming, depending on the number of input points†, the selection of an optimal and efficient resolution of the accumulator space can be complicated, etc. To overcome these problems, many new Hough versions have been proposed during the past few years. *Table 1* summarizes the approaches. The methods are categorized in this paper as *non-probabilistic* vs. *probabilistic* Hough transforms. The probabilistic algorithms use random sampling for selecting only a small subset of the data. For more Hough techniques see, for example, an excellent survey of recent developments by Leavers[35]. Comparisons of some of these methods have been presented elsewhere[36-39]. In the following subsections, a brief overview of the methods presented in *Table 1* is given.

Table 2 presents characteristics of the HT which are also introduced in *Table 1*. The key characteristic features are random sampling, many-to-one mapping and use of a list structure instead of an accumulator array. These subjects are discussed in the following subsections.

Non-probabilistic Hough transforms

Gerig and Klein[18,19] considered the problem of rapidly finding correspondences between image points and parameter curve points in HT, and they suggested a backmapping which links the image space and the accumulator space. After the accumulation, each pixel in an image can be linked to the most evident location in the accumulator space. In this way, each image point obtains the parameters of the curve it is part of. This post-processing of the accumulator space makes it possible to track different boundaries in the original picture effectively. In practice, to achieve this result, a given boundary point should be a member of only one curve in the image space.

In the Fast Hough Transform (FHT) by Li *et al.*[20], and the Adaptive Hough Transform (AHT) by Illingworth and Kittler[21], an iterative 'coarse to fine' exploration of the parameter space was suggested. In the FHT, the parameter space is divided recursively into hypercubes from low to high resolution, and the HT is performed further only on the hypercubes whose votes exceed a desired threshold T. In other words, the HT is performed again on selected hypercubes which have more accurate resolution now. This subdivision process is repeated until every quadrant reaches a predefined size (resolution) or its score does not reach T, and finally, the surviving quadrants are used for selecting candidate peaks. The merits of the method are that it is easy to make a parallel implementation, and that the data structure is regular which facilitates analysis, e.g. T can be set adaptively. On the other hand, this kind of a structure may cause problems if a peak is close to the borders of quadrants. The Adaptive Hough Transform also utilizes a small fixed size accumulator (for example, 9×9) having a 'coarse to fine' accumulation until a desired resolution is achieved. Instead of using fixed limits for the parameters as in the FHT, the limits for the parameters are selected adaptively in the AHT. This is an evident merit of the algorithm, and it yields effective adaptive sharpening of the accumulator space in general. The candidate peaks are identified by thresholding and a connective components analysis. Another merit is the recovery of special cases in parameter limits like when a candidate peak is adjacent to an edge of the accumulator. The disadvantage of the AHT is that it extracts curves segment-by-segment, and the whole process has to be repeated to find a new curve segment.

Princen *et al.*[22] introduced the Hierarchical Hough Transform (HHT). In the HHT an image is divided into small subimages (for instance, 16×16) and the HT is performed on these subimages. The accumulation is carried out curve-by-curve and, when found, each curve is extracted from the subimage. The procedure is stopped when a predefined maximum value of the accumulator has been obtained. As a result, a group of curves are defined for each subimage. After the low-level line segments have been detected, the line segments are grouped hierarchically level by level with the HT. Because of the small subimages, the size of the needed accumulator array can be kept small, which will lead to an efficient and robust process.

Ben-Tzvi and Sandler[27] presented the Combinatorial Hough Transform (CHT). The algorithm uses two pixels of the image to calculate the line parameters.

*In this article, a parameter space is also called the accumulator space, or simply the accumulator.

†Input points are usually the edge points of the original image.

Each pair of two image pixels determine one (ρ, θ) cell in the 2D accumulator array. For limiting the number of pixel pair combinations, the image is segmented (typically in 64 regions) and the voting process is performed segment by segment. Costa *et al.*[28] have described an improved parameterization for the CHT to achieve higher detection accuracy.

Liang[24] suggested the Curve Fitting Hough Transform (CFHT). He fitted a segment of curve in a small neighbourhood of edge points in the image. If the fitting error is less than a given tolerance, the parameters obtained from the curve fitting are mapped as a single point into a parameter space (called a parameter list in this case). First, the fitting is done to each suitable $M \times M$ window in the edge image, and finally, the parameter space is examined to find curves which are extracted from the image space. An important benefit of the CFHT is its computational speed compared to the SHT although the speed largely depends upon the combination of the selected parameters. There are some similarities in CFHT to the Randomized Hough Transform (RHT)[3] suggested earlier. The CFHT borrows from RHT the combined use of a many-to-one mapping and a dynamic list accumulator, but with the random sampling replaced by windowed fitting. Thus, the CFHT explicitly shares the advantages of the RHT like the arbitrarily high resolution, the infinite accumulator space and the small storage. In spite of

Table 1 Recent versions of Hough transforms

Non-probabilistic Hough transforms	*Probabilistic Hough transforms*
• Backmapping Hough Transform[18,19]	• Parallel Guessing Implementation of the Standard Hough Transform[2]
• Fast Hough Transform[20]	• Randomized Hough Transform[3 6,39–41,48]
• Adaptive Hough Transform[21,46]	• Probabilistic Hough Transform by Kiryati *et al.*[7,42]
• Hierarchical Hough Transform[22,47]	• Monte Carlo Hough Transforms[8,9]
• Hough Techniques by Risse[23]	• Random Calculation of the Intersections among Hough Surfaces by Shiono[11]
• Combinatorial Hough Transform[27,28]	• Random Sampling of Minimal Subsets by Roth and Levine[10]
• Curve Fitting Hough Transform[24,25]	• Dynamic Combinatorial Hough Transform[12,13]
• Probabilistic Hough Transform by Stephens[26]	• Dynamic Generalized Hough Transform[14]
	• Connective Hough Transform[15]
	• Hough Prediction/Correction Approach[16,17]

Table 2 Characteristic features of recent Hough transforms. The first column indicates the method, the second column the parameterization of lines, the third the mapping from the input space to the parameter space, the fourth the structure of the accumulator, the fifth the method of sampling the input points, and the sixth the method of detecting lines in a picture

Non-probabilistic Hough Transforms					
Method	*Param.*	*Mapping*	*Accumulator*	*Sampling*	*Detection*
SHT	(ρ, θ)	one-to-many	2D array	all points	all lines
FHT	(a, b)	one-to-many	2 × 2D quadtree	all points	all lines
AHT	(a, b)	one-to-many	2 × small 2D array	all points	line by line
HHT	(ρ, θ)	one-to-many	varying sized 2D arrays + 2D linked list	all points	line by line
CHT	(ρ, θ)	many-to-one	2D array	all points	all lines
CFHT	(a, b)	many-to-one	2D linked list	all points	all lines
PHT	(ρ, θ)	one-to-many	2D array	all points	all lines
Probabilistic Hough Transforms					
Method	*Param.*	*Mapping*	*Accumulator*	*Sampling*	*Detection*
PGI-HT	(ρ, θ)	one-to-many	2D array	enough	line by line
RHT	(a, b)	many-to-one	2D linked list	enough	line by line
rtRHT	(ρ, θ)	many-to-one	2D linked list	enough	line by line
ProbHT	(ρ, θ)	one-to-many	2D array	subset	all lines
MCHT	(a, b)	one-to-many many-to-many many-to-one	2D array	subset	all lines
DCHT	(ρ, θ)	many-to-one	θ-histogram	all points	line by line

SHT = Standard Hough Transform[23,49]
FHT = Fast Hough Transform[20]
AHT = Adaptive Hough Transform[21,46]
HHT = Hierarchical Hough Transform[22,47]
CHT = Combinatorial Hough Transform[27]
CFHT = Curve Fitting Hough Transform[24,25]
PHT = Probabilistic Hough Transform of Stephens[26]
PGI-HT = Parallel Guessing Implementation of the HT[2]
RHT = Randomized Hough Transform[3–6]
rtRHT = (ρ, θ) Randomized Hough Transform[38]
ProbHT = Probabilistic Hough Transform by Kiryati *et al.*[7]
MCHT = Monte Carlo Hough Transforms[8,9]
DCHT = Dynamic Combinatorial Hough Transform[12,13]

these merits and its own advantages, the CFHT also has disadvantages which the RHT avoids. Some of these difficulties are discussed by Liang[25].

In the probabilistic approach to the Hough transform proposed by Stephens[26], the relationship between the HT and the maximum likelihood parameter estimation has been explored, and it has been shown that the HT is a maximum likelihood method. The Probabilistic Hough Transform (PHT)*, a mathematically 'correct' form of the HT, is defined as a likelihood function in the output parameters. The algorithm has been tested for finding straight lines from oriented edgels, and it has been shown that the conventional Hough method gives a good approximation to the PHT. The PHT has also been applied to a tracking problem, and found to be robust in high dimensional Hough spaces. The PHT is more accurate than the HT, but it is computationally expensive. A special climb algorithm has been developed that enables efficient parallel evaluation of the likelihood function and its derivative.

Probabilistic Hough transforms

Recently, several probabilistic approaches to the HT have been suggested. All of them use the idea of random sampling. One of the major problems of the HT is that the complexity of the method is largely dependent on the size of the input data. By choosing a small subset of points from the original input data set at random, significant computational savings are obtained. These savings reduce the computation time and memory storage. Furthermore, some of the methods also use many-to-one mapping, and they replace an accumulator array by a list structure. These techniques may also save the computation and memory usage. The RANSAC approach and the parallel guessing implementation of the Standard Hough Transform[2], the Randomized Hough Transform[3–6,40,41], the Probabilistic Hough Transform[7,42] and the Monte Carlo Hough Transforms[8,9], among other probabilistic algorithms[10–17], apply random selection. This section is a short overview of these probabilistic methods, with comparisons to the basic RHT algorithm.

In the Randomized Hough Transform (RHT), Xu *et al.*[3] proposed a method for curve detection. For a curve expressed by an n parameter equation, they randomly selected n pixels and mapped them into one point of the parameter space, instead of transforming one pixel into an $n-1$-dimensional hypersurface in the parameter space, as the HT and its variants do. In comparison with the HT and its variants, they showed through analysis and experiments that their new method has the advantages of small storage, high speed, infinite accumulator space, and arbitrarily high resolution[6]. In addition, the difficulties of choosing an appropriate window and sampling for the accumulator and of finding local maxima are avoided. These are two main difficulties which significantly affect the performance of the HT and its variants. The RHT is explained in more detail below.

Fischler and Firschein[2] introduced one of the earliest random approaches, the RANSAC approach, and the parallel guessing implementation of the Standard Hough Transform (here, denoted PGI-HT). RANSAC[43] is a generate-and-test method: n pixels are randomly selected for obtaining a guess on curve parameters, and then other pixels used to test the guess. The PGI-HT randomly selects one pixel which is mapped into a curve of the parameter space[2]. This process is continued until a control process of the PGI-HT determines that a peak in the parameter space is an evident candidate for a maximum. The random data selection is similar to the Randomized Hough Transform, but the two methods are conceptually not the same: the RHT randomly selects n pixel points to map them into one point of the parameter space by solving n curve equations, and after certain iterations finds out all the dense clusters in the parameter space to obtain the curve parameters. Contrary to this, PGI-HT randomly selects only one point and maps it into a line of the parameter space. Therefore,, it is more similar to the original HT. So, the RHT is a many-to-one mapping whereas the PGI-HT is a one-to-many mapping.

Roth and Levine[10] presented a probabilistic Hough method of random sampling of minimal subsets. Their algorithm is a generalization of the RANSAC.

The Probabilistic Hough Transform by Kiryati *et al.*[7] (ProbHT) only uses a small, randomly selected subset of the edge points in the image. This limited poll is used as an input for the HT. Since the size of the subset providing a good performance is usually small, the execution time can be reduced considerably. Thus, the key to a successful application of the method is the dependence of the algorithm's performance, in terms of detection probability and false alarm rate, on the fraction of the data used. The algorithm has been tested for the case of straight lines and for the case of circles. The method differs from the RHT in both randomizing and accumulating. A subset of the edge points is chosen, after which Hough transform is applied to each point of the subset. Again, this is a one-to-many mapping, in contrast to the many-to-one mapping of the RHT. Ylä-Jääski *et al.*[42] proposed several improvements to the ProbHT, like new strategies in peak detection in the accumulator and automatic termination rules in voting. Similar attempts to develop automatic termination rules in voting with the RHT and motion detection have been discussed in brief elsewhere[32,44].

The Monte Carlo Hough Transform (MCHT) was suggested by Shvaytser and Bergen[8,9]. They used random sampling partially similar to the RHT. The main contribution of this work is the observation that it is sometimes advantageous to choose sets of points of arbitrary size randomly. The kernel of the method is as

*Note that the PHT, despite its name, is here classified into the non-probabilistic methods, because no random sampling is involved.

follows: first, the following procedure is repeated N times: (a) choose at random (with replacement) r pixels $p_1, \ldots, p_r$ from the input image; (b) increment the accumulator for all cells satisfying an equation $q(p_j, \phi) = 0$ for $j = 1, \ldots, r$, where ϕ is a parameter vector. For example, in the case of line detection, $x_2 - \phi_1 x_1 - \phi_2 = 0$, where $p_j = (x_1^{(j)}, x_2^{(j)})$. Second, the accumulator is examined to detect local maxima. Parameters N and r have to be predefined. Using some parameters like size, confidence, accuracy and a preset number of cells in the accumulator, the parameters N and r can be optimized. This approach has fewer increments into the accumulator than the HT, and thus speeds up computation dramatically.

Compared to the RHT, the Monte Carlo Hough Transform has a fixed accumulator space, while the RHT has a dynamically-linked structure without any predefined size. To calculate the Hough transform of patterns of points, several parameters have to be defined. When the number of selected points r is less than the number of parameters in a curve segment to be detected, the Monte Carlo Hough Transform increments more than one cell in the accumulator space. When r is equal to the number of curve parameters, it accumulates one cell like the RHT always does. A property of the Monte Carlo HT, not shared by the RHT, is sets of points of arbitrary size in randomizing, and thus the possibility to optimize the number of points r selected at random.

Shiono[11] introduced an arc detection method using the HT with random sampling of pixels. The algorithm, called Random Calculation of the Intersections among Hough surfaces (RCIH), has been designed for detection of occluded circles. This is done by calculating the centre points of gravity and the end point angles of the arcs. The RCIH method does not involve an accumulator array. Instead, a small table for parameter sets is needed.

The Dynamic Combinatorial Hough Transform (DCHT) presented by Ben-Tzvi *et al.*[12,13] also belongs to probabilistic HT algorithms if a seed point is selected randomly among feature points. However, the mechanisms of the random sampling were not explicitly defined[12,3]. In the DCHT, the (ρ, θ) line parameterization is used. All two point combinations with the seed point are accumulated into a single value in a θ-histogram. This process is repeated until a predefined threshold is reached with some seed point in the 1D histogram of θ values. In the next stage, when the threshold is obtained, a detected line is removed, and this sampling procedure is continued until all points have been removed, i.e. lines are detected one by one. If the threshold is not reached, only the seed point is removed from the feature points, and the sampling procedure is continued.

Leavers[14] generalized the DCHT to the Dynamic Generalized Hough Transform (DGHT). In the DGHT, a single connective point is selected and a segment relative to the connective point is determined. A suitable number of points in the segment to calculate the curve parameters are chosen at random, and the solved curve parameters are accumulated. This random sampling is continued until a stopping criterion is satisfied. Finally, once a shape (e.g. a circle or an ellipse) has been successfully parameterized, it is removed from the image, and curve detection is continued. Lam *et al.*[16,17] have considered similar detecting problems, and proposed an approach both for circle detection and ellipse detection.

Yuen *et al.*[15] reported on the Connective Hough Transform (ConHT). The algorithm is similar to the Dynamic Combinatorial Hough Transform, for instance, in having a seed point, in this case selected at random, to calculate a 1-dimensional θ-valued accumulator. The accumulation, however, is different. In the ConHT the accumulation is performed row by row in order to have connectivity between the accumulated points. Since vertical and horizontal lines are considered separately, they are, in fact, two accumulators. Probabilistic information may also be gathered during the accumulation to select useful seed points. Improved performance compared to the DCHT is presented elsewhere[15].

EXTENSIONS OF THE RANDOMIZED HOUGH TRANSFORM

This section shortly introduces the basic ideas of the Randomized Hough Transform[3,6] which belongs to the family of probabilistic Hough transforms. The main elements of the RHT, random sampling, converging mapping, and dynamically-linked list, are briefly explained. New extensions of the RHT are also suggested in this section. Four algorithms have been developed: the Dynamic RHT (DRHT), the Random Window RHT (RWRHT), the Window RHT (WRHT) and the Connective RHT (CRHT).

Basics of the RHT algorithm

The RHT method[3,4] is based on the fact that a single parameter point can be determined uniquely with a pair, triple or generally n-tuple of points from the original picture, depending on the complexity of the curves to be detected. For example, in the case of line detection, each parameter space point can be expressed with two points from the original binary edge picture. Such point pairs (d_i, d_j) are selected randomly, the parameter point (a, b) is solved from the curve equation, and the cell $A(a, b)$ is accumulated in the accumulator space. This random selecting is called *random sampling*. The RHT is run long enough to detect a global maximum in the accumulator space. The parameter space point (a, b) of the global maximum describes the parameters of the detected curve, which can then be removed from the image to start the algorithm again with the remaining pixels.

The main difference between the conventional HT

and the RHT for line detection is that, while a single pixel in the original image is mapped to a curve in the parameter space in the HT, a pair of pixels is mapped to a single cell in the parameter space in the RHT. Thus, while in the HT curves are mapped into the parameter space using a function which generates all parameter combinations compatible with both the observed pixel and the curve model, the RHT only generates a small subset of all parameter combinations[4]. Thus, the RHT uses many-to-one mapping or converging mapping. The following algorithm summarizes the ideas of the RHT*:

Algorithm 1: *The kernel of the RHT to line detection*

1. Create the set D of all edge points in a binary edge picture.
2. Select a point pair (d_i, d_j) randomly from the set D.
3. If the points do not satisfy the predefined distance limits, go to Step 2; otherwise continue to Step 4.
4. Solve the parameter space point (a, b) using the curved equation with the points (d_i, d_j).
5. Accumulate the cell $A(a, b)$ in the accumulator space.
6. If the $A(a, b)$ is equal to the threshold t, the parameters a and b describe the parameters of the detected curve; otherwise continue to Step 2.

To define the distance limits in Step 3 means that the points d_i and d_j must not be too near each other or too far from each other, i.e. $dist_{\min} \leqslant dist(d_i, d_j) \leqslant dist_{\max}$, where $dist(d_i, d_j)$ is the Euclidean distance between the points d_i and d_j. In this paper, this limitation is called *the point distance criterion*. If the edge picture is complex the use of distance limits is necessary. Otherwise, computation yields a lot of waste accumulations.

The accumulation of the cell $A(a, b)$ means incrementing its value by one. The accumulator space can have the form of a dynamic structure like a tree, because now only one cell will be updated at a time. With the usage of the dynamic tree structure, arbitrarily high accuracy and small memory consumption are achieved. More details of the dynamic structure and some other possibilities were given elsewhere[4,6].

The algorithm contains a threshold which an accumulator cell must reach to be detected as the global maximum. The noisier the original edge picture is, the higher the threshold should be. Instead of a constant threshold, a variable threshold can be also used like in motion detection using the RHT[32,44]. By the variable threshold the random sampling can be stopped automatically without a predefined maximum value t in the accumulator. The details on basic mechanisms and algorithms of the RHT are presented elsewhere[3], where the advantages of the RHT are also stated: high parameter resolution, infinite scope of the parameter space, small storage requirements and fast speed. An extension to the Generalized Randomized Hough Transform (GRHT) has been proposed[45]. The (ρ, θ) parameterization of the RHT in line detection instead of the (a, b) one is considered by Hare and Sandler[38]. Improved algorithms and the deep mechanisms behind the advantages of the RHT were discussed by Xu and Oja[3,6], who also discuss the general n – parameter problem.

*In this article, the algorithm is called the basic RHT.

Novel extensions of the RHT

The Dynamic RHT (DRHT) method is an iterative process of two RHTs. First the original RHT is run until the accumulator threshold is reached by some accumulator cell. For the second iteration of the algorithm, the set of feature points is formed by collecting the points that are near to the line found in the first iteration. This is realized by scanning the image along the line of the first iteration at the width of a few pixels. The scan width takes into account a possible error in both the slope and the intercept of the line. Next, the new set of points is accumulated in the zeroed accumulator and, when the accumulator threshold has been exceeded again, the line is found. From that stage the algorithm follows the original RHT. For the second RHT iteration, the accumulator resolution and the accumulator threshold are usually selected to be higher than those of the first iteration. Because both the resolution and the threshold are lower in the first stage of the algorithm, and the number of sampled points is relatively small in the second stage, time can be saved by this combination technique. Algorithm 2 illustrates the new strategy.

Algorithm 2: *Maxima search strategy of the Dynamic RHT (DRHT)*

1. Run the RHT long enough to detect a global maximum in the accumulator space.
2. Zero the accumulator.
3. With higher accumulator resolution and higher threshold, run the RHT long enough for the subset of image points scanned near the line defined by Step 1 to detect a global maximum in the accumulator space.
4. Remove all those points from the set D which lie under the curve determined by the location of the maximum in the accumulator space.
5. Zero the accumulator.
6. Continue to Step 1.

Two novel windowing versions of the RHT, called the Window RHT (WRHT) and the Random Window RHT (RWRHT), were recently presented by Kälviäinen *et al.*[40]. Both of them use random sampling and many-to-one mapping. Both algorithms utilize more local information than the original RHT, but since they have random elements the drawbacks of window sampling may be avoided, and the merits of window sampling obtained. For simplicity, the algorithms are given here for the case of line segment detection (i.e. the number of parameters n is 2), but they have a direct extension to other curves, too.

In the Random Window RHT (RWRHT), a window location is first randomly selected from the binary edge picture. The RHT procedure is performed in the $m \times m$ window, whose size m is also randomized. Random sampling is repeated R times, where R could be a function of the window size m. Window sampling is repeated until a predefined threshold t is reached. Lines are detected one by one until a desired number of lines l_{max} have been found or some heuristic criterion stops the computing. When a line has been found and verified to be a true line its pixels are removed from the binary image. The algorithm for line detection is as follows:

Algorithm 3: *The Random Window RHT Algorithm for line detection*

1. Scan a binary edge image and put the coordinates $d_i = (x_i, y_i)$ of all edge pixels into the pixel data set D. Set a threshold t for a global maximum in the accumulator space A, which is some data structure. Set limits m_{min} and m_{max} to a window size. Set the number of lines to be detected l_{max}. Initialize the accumulator space A. Set $l := 0$.
2. Select one point d_i randomly from the set D. Set $k := 0$.
3. Randomize a window size m where $m_{min} \leqslant m \leqslant m_{max}$.
4. Create a pixel data set W of the $m \times m$ neighbourhood of the point d_i. Points of the set W are denoted $w_i = (x_i, y_i)$.
5. Set a number of random selections $R = f(size(W))$.
6. Select a point pair (w_i, w_j) randomly from the set W.
7. If the points do not satisfy the predefined distance limits, go to Step 6; otherwise continue to Step 8.
8. Solve the parameter space points (a, b) from the curve equation with the points (w_i, w_j). Set $k := k + 1$.
9. Accumulate the cell $A(a, b)$ in the accumulator space.
10. If the score of the cell $A(a, b)$ is equal to the threshold t go to Step 11; otherwise if $k < R$ go to Step 6; otherwise go to Step 2.
11. If the line parameters (a, b) define a true curve take out of D all the pixels lying on the curve and set $l := l + 1$.
12. If $l = l_{max}$ or another stop criterion is satisfied then stop; otherwise set $A = null$ and go to Step 2.

The Window RHT (WRHT) is a simpler version that selects one edge point randomly, fits a curve to a fixed size neighbourhood of the edge point, and defines the curve parameters. The curve fitting can be done, for example, by the least square method. Only the parameters satisfying a certain goodness of the fitting are accepted to update the accumulator space. The WRHT process is continued until the maximum score in the accumulator is equal to the threshold t. This approach determines line segments curve by curve, too. In detail, the WRHT algorithm for line detection is as follows:

Algorithm 4: *The Window RHT Algorithm for line detection*

1. Scan a binary edge image and put the coordinates $d_i = (x_i, y_i)$ of all edge pixels into the pixel data set D. Set a threshold t for a global maximum in the accumulator space A. Set a window size m. Set a number of lines to be detected l_{max} and a maximum for a fitting error c_{max}. Set $l := 0$. Initialize the accumulator space A.
2. Select one point d_i randomly from the set D.
3. If enough points are found in an $m \times m$ neighbourhood of the point d_i, fit a curve to pixels and calculate the line parameters (a, b); otherwise go to Step 2.
4. If the fitting error e is less than e_{max} accumulate the cell $A(a, b)$ in the accumulator space; otherwise go to Step 2.
5. If the score of the cell $A(a, b)$ is equal to the threshold t go to Step 6; otherwise go to Step 2.
6. If the line parameters (a, b) define a true curve take out of D all the pixels lying on the curve and set $l := l + 1$.
7. If $l = l_{max}$ or another stop criterion is satisfied then stop; otherwise set $A = null$ and go to Step 2.

Both the RWRHT and the WRHT renew the random sampling mechanisms of the RHT, but leave the accumulation technique the same as earlier. Some new accumulation approaches have also been proposed[5]: the curve parameters can be stored in quantized values or a mixture structure of two hash tables and one linear list can be used. These two approaches can be combined to the RWRHT and the WRHT, too.

The most critical constraint of the algorithm for correct and reliable operation is the window size. It must be large enough so that desired detection accuracy can be achieved. However, the size is limited by the average separation distance of adjacent curves. Also, in the case of a noisy image as input, a large window tends to 'collect' noisy points giving a high fitting error as a result and thereby some of the lines may not be found. So, the window size is an image dependent value and can be optimized considering those factors.

An extension to the RTHT, called the Connective RHT (CRHT), was recently developed by Kälviäinen and Hirvonen[41] to handle these problematic situations. The extension introduces a connective component search of the windowed points. Now, for the fitting process only those points of the window are used that are connected to the centre point of the window with an 8-path. Furthermore, the connective component search can be performed as sectored. In this context, the sectoring means limiting the search direction to the original one and its two most similar directions. For example, in terms of compass directions, if the search is started with direction of north, allowed directions for a further search are north, northeast and northwest. This is done to cut down connection paths that consist of loops or sharp bends. By these further suggestions, the difficulties of the window line fitting can often be

successfully avoided. Only in the special case of images in which the edges are not 8-connected, i.e. there are gaps between edge pixels, the connective component search cannot be used to improve performance. The kernel algorithm of the CRHT is as follows:

Algorithm 5: *The kernel of the Connective RHT (CRHT)*

1. Create the set D of all edge points in a binary edge picture.
2. Select a point d_i randomly from the set D.
3. Create a $w \times w$ window centred at d_i, and perform the sectored connective component search to the windowed edge points.
4. If enough connective points are found in the window, fit a curve to those points and calculate the line parameters (a, b); otherwise go to Step 2.
5. If the fitting error is within a tolerance, accumulate the cell $A(a, b)$ in the accumulator space; otherwise go to Step 2.
6. If the $A(a, b)$ is equal to the threshold t, the parameters a and b describe the parameters of the detected curve; otherwise continue to Step 2.

The connective component search can easily be done in a recursive way. When it is done as discussed earlier, the search direction is limited. As a result of the process, only the marked pixels are used in curve fitting, instead of using all edge pixels. When the 8-path-connectivity to the centre point of the window is demanded, the centre point is the initial pixel in the sectored connective component search (SC). Thus, the SC algorithm is as follows:

Algorithm 6: *Sectored connective component search (SC)*

1. Set a new searching direction and move to that direction.
2. If the present pixel is 'on' and it is not marked, mark it; otherwise move back to the initial pixel and go to Step 1.
3. Search and mark recursively all the 'on' pixels in the present direction and its two nearby directions.
4. If there are unsearched directions left in the 8-neighbourhood of the initial pixel, move back to the initial pixel and continue to Step 1.

Like the original RHT, the CRHT considers not only the local connectivity, but also retains the random sampling through the random selection of the window location. Thus it, too, is a combination of the key features of the RHT: the random sampling, the many-to-one mapping and the dynamic list accumulator.

TEST RESULTS

Some of the methods reviewed or introduced above were tested on both complex synthetic and complex real-world images*. The algorithms chosen for the tests are as follows:

- Standard Hough Transform (SHT).
- Combinatorial Hough Transform (CHT).
- Curve Fitting Hough Transform (CFHT).
- Probabilistic Hough Transform by Kiryati *et al.* (ProbHT).
- Randomized Hough Transform with the point distance criterion (RHT_D).
- RHT without the point distance criterion (RHT_ND).
- Dynamic RHT (DRHT).
- Window RHT (WRHT).
- Connective RHT (CRHT).
- Random Window RHT (RWRHT).
- Dynamic Combinatorial Hough Transform (DCHT).

A serious attempt was made to select the test parameters for each method as optimally as possible. Several combinations of the parameters were selected, and the best combinations for each method were presented as test results. Of course, the test results may vary according to both the selected parameters and test pictures. Since the algorithms have many parameters *Table 3* summarizes the limits of algorithm constraints in the tests.

Tests with a complex synthetic image

The first test picture is shown in *Figure 1*. The test picture contains 50 randomly generated synthetic lines. Two tests were run: one to detect 50 realistic candidate lines and one to detect as many of those as possible. A *candidate line* is any line produced by the algorithm. A *realistic candidate line* satisfies *line criteria*, i.e. the minimum number of pixels, the maximum gap between

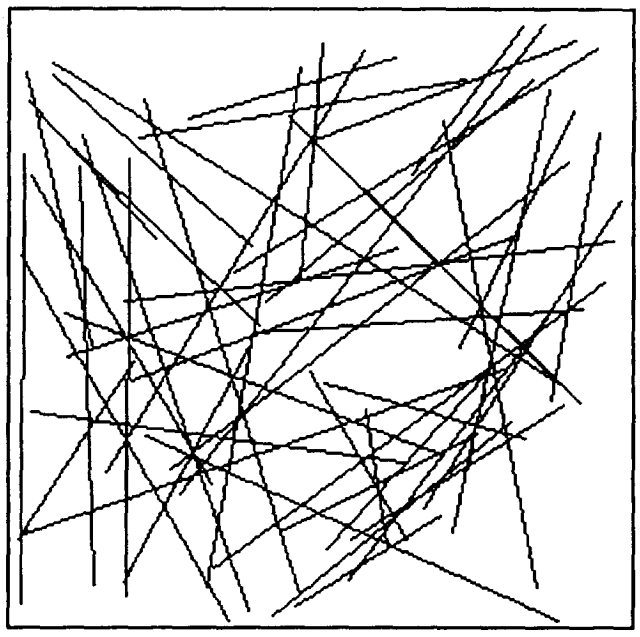


Figure 1 Synthetic binary image of 50 lines

*Test runs were performed on a standard SUN SPARCstation IPX.

Table 3 Constraints of the methods in the tests

Method	*Constraints*
SHT	The 256 × 256 accumulator, the Risse's clustering method[23] to accumulator peak detection.
CHT	The 256 × 256 accumulator, 32 × 32 pixel segments with overlapping of 10 pixels, the local peak finder of a 3 × 3 – 21 × 21 pixel mask to accumulator peak detection
CFHT	The 9 × 9 – 31 × 31 window, the maximum fitting error used was 0.1 – 0.5, the accumulator epsilon 5.0 – 10.0.
ProbHT	The sampling rate was selected as low as possible so that the detection rate was as high as with the SHT. Other constraints were set just like in the SHT.
RHT	The accumulator resolution two decimals, the accumulator threshold 2 – 4, the points distance limits for minimum distance 5 – 15 and maximum distance 24 – ∞.
DRHT	The segment width nine pixels plus the pixels caused by the slope variation, the slope variation which was set to 3°, the accumulator threshold for the second iteration was always equal to or higher than that of for the first iteration.
WRHT, CRHT	The accumulator threshold also one, the window size 11 × 11 – 61 × 61.
RWRHT	The window size 21 × 21 – 141 × 141.
DCHT	The accumulator threshold, e.g. $\frac{4}{5}$ of the minimum line length, the θ-histogram quantized to 256 values.

pixels, etc. For all the synthetic images, the real parameters of the lines were always known, and it was checked after each test if the detected line parameters were among them. The maximum differences between detected and real line parameters allowed were ±5 pixels in ρ and ±0.025 radians ($\approx 1.43°$) in θ. This criterion is called *the real line criterion*. Realistic candidate lines satisfying the real line criterion are called *real lines*. Results of the test are summarized in *Table 4* and the output images corresponding to *Table 4a* are shown in *Figure 2*.

The computation time of the RHT_NT was found to be much higher than the others. A reason for that is in the fact that two randomly selected points from an image of many lines are not very likely to belong to the same line segment. This will yield a large number of useless accumulations. With simpler images this effect is not so obvious, and in such a case RHT_ND works. With more complex images like *Figure 1*, the point distance criterion is essential. To overcome this random sampling problem, the maximum distance of points in a tuple was limited to 25 pixels in the RHT_D, the DRHT and the RWRHT.

The most accurate, but slow methods were the SHT, the CHT and the ProbHT, while the WRHT and the CRHT were the fastest. In addition, the CRHT is one of the most accurate algorithms. Notice that the CFHT was not able to find more than 28 true lines. This number of lines, which was

Table 4 Test results of the line detection from a 50-line image. (a) Detecting 50 lines; (b) detecting as many lines as possible. The first column indicates the method, the second lists the number of real lines detected, and the third lists CPU times in seconds. Note that real lines satisfy both the line criteria and the real line criterion. The fourth column lists the false alarms, i.e. lines not satisfying the line criteria, expressed as percentage of the total number of candidate lines returned by the algorithm. The fifth and sixth columns denote the average and maximum number of active accumulator cells during the test run. Of course, those methods that apply a static accumulator have equal values in both the two last columns

Method	*Lines*	*Time*	*Falses (%)*	*Av. Size*	*Max. Size*
SHT	48	94.60	0.0	65536	65536
CHT	37	53.67	12.0	65536	65536
CFHT	28	3.06	43.8	40	80
ProbHT	47.0	79.52	2.0	65536	65536
RHT_ND	17.2	138.08	9.4	903	3164
RHT_D	25.5	17.03	10.9	131	473
DRHT	27.8	19.09	15.3	179	794
WRHT	36.9	1.47	0.0	0	0
CRHT	43.3	1.80	0.2	0	0
RWRHT	26.9	2.64	13.5	63	261
DCHT	43.9	4.28	0.8	256	256

a

Method	*Lines*	*Time*	*Av. Size*	*Max. Size*
SHT	48	94.93	65536	65536
CHT	47	54.50	65536	65536
CFHT	28	3.06	40	80
ProbHT	47.0	79.63	65536	65536
RHT_ND	37.6	290.10	677	3013
RHT_D	44.6	19.06	134	669
DRHT	46.3	29.13	151	730
WRHT	43.8	2.06	0	0
CRHT	46.4	2.07	0	0
RWRHT	46.0	4.55	52	242
DCHT	48.5	8.12	256	256

b

the best result by the algorithm, was obtained by using a 13×13 window. The reason for such a low detection rate is partly in small window size, which does not give extremely high accuracy for parameter definition, and partly in the accumulation policy that allows the cells to move from their original locations in the parameter space.

When only 50 candidate lines are detected, the SHT, the ProbHT, the CRHT and the DCHT give good results, as displayed in *Table 4a* and *Figure 2*. However, in the case of detecting as many lines as possible (*Table 4b*) the CHT, the DRHT and the RWRHT also obtained reasonable results.

Tests with a complex real world image

The second test picture is presented in *Figure 3a*. The binary edge image of *Figure 3b* is obtained by using a Difference Recursive Filter (DRF) to extract edges from the bookshelf image of *Figure 3a*. The lines to be detected are not ideal as in the synthetic image used in the previous tests. Instead, the lines are wider and not so exactly straight. Basically these 'errors' in edges are due to nonlinear edges of the object, and to poor quantization of the image. Also, the edge detection process may cause distortion or extra noise to the edge image, or cannot detect edges correctly.

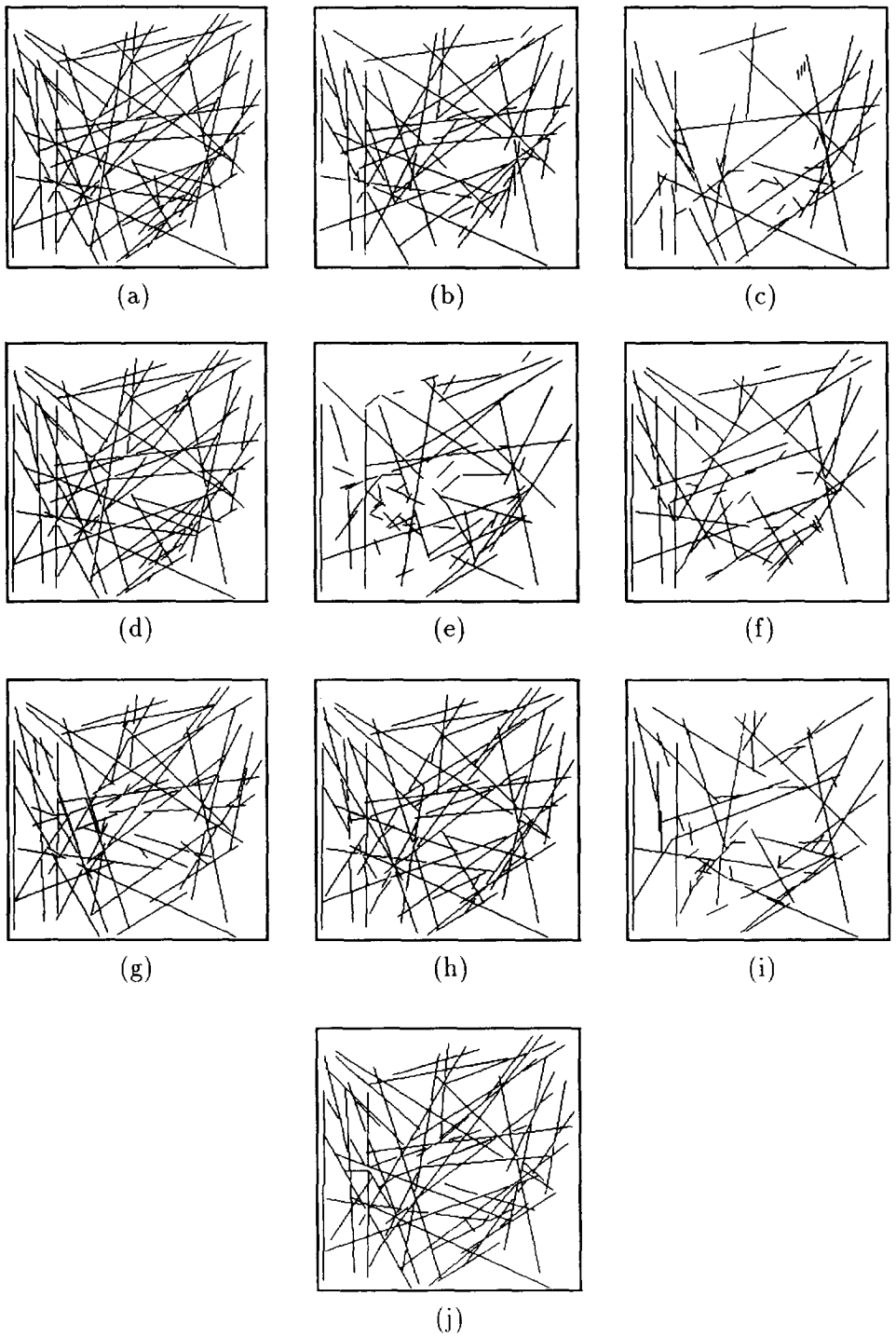
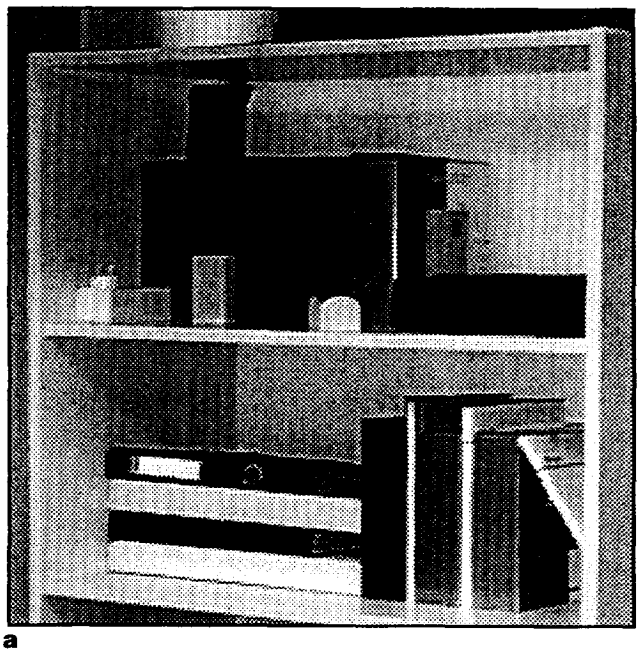
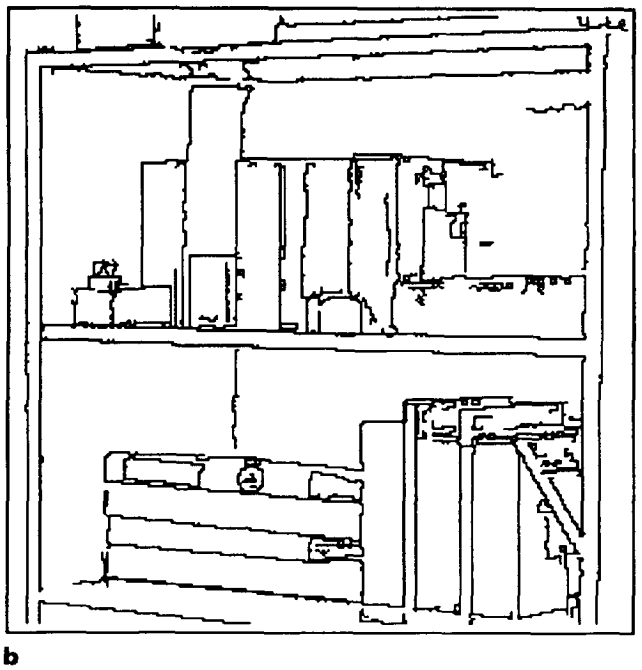


Figure 2 Resulting images of line detection from a 50-line image by (a) SHT, (b) CHT, (c) CFHT, (d) ProbHT, (e) RHT_D, (f) DRHT, (g) WRHT, (h) CRHT, (i) RWRHT, (j) DCHT



a



b

Figure 3 Complex real world image. (a) Original grey-level image; (b) binary edge image corresponding to (a)

Computation times and the numbers of detected lines are summarized in *Table 5*. Corresponding output images of the algorithms are illustrated in *Figure 4*.

The SHT and the ProbHT are slow, as expected, and the RHT_ND becomes very slow when the number of lines is more than a few dozen. The RHT with the point distance criterion (the RHT_D) is far faster than the RHT_ND. The DRHT was able to be tuned a bit faster without major disadvantages to the detection result by setting the accumulator threshold of the first iteration to one, because quite often the first candidate parameters already define an existing line. However, the RWRHT was even faster, and the detection accuracy was similar to the RHT and the DRHT. Also, the DCHT gave a good result, but computation time was longer than the computation times of the extensions of the RHT.

The most inaccurate detection result was received from the CFHT. The method was not able to run with a fitting window larger than 9×9, since there are many groups of two adjacent lines in the edge image. The same reason limited the window size of the WRHT, but the detection result looks still much better and computation time was the smallest. Furthermore, the CRHT allowed to set the window size even to 30×30, since it employs the connective component search of windowed points. The RHT variants detected lines in more segments than the SHT. This is a characteristic property and means a kind of trade-off between speed and the accuracy of the method. The SHT, even if one of the slowest, seemed to be the most accurate method.

DISCUSSION AND COMPARISONS

From analysing the properties of the algorithms and from the previous tests, some conclusions can be made on the relative performance of the methods. The tests were made for the two-parameter line detection problem only, but also more general conclusions can be drawn from the algorithms.

Non-probabilistic methods

The SHT is the most accurate method, but computation speed is very low. Moreover, it needs a large predefined fixed size storage in accumulation. The CHT is faster than the SHT. A possible disadvantage of the algorithm seems to be that it may miss some lines due to small segment size. If the segment size is larger, the computation becomes slower. Also, the performance of the method depends more upon the distribution of the

Table 5 Test results of the line detection from a complex real world image

Method	*SHT*	*CHT*	*CFHT*	*ProbHT*	*RHT_ND*	*RHT_D*
Lines	61	102	83	54.0	115.0	115.0
Time	99.83	58.00	4.85	74.93	130.14	14.81
Method	*DRHT*	*WRHT*	*CRHT*	*RWRHT*	*RWRHT*	
Lines	113.0	114.8	119.7	115.4	95.2	
Time	8.12	3.02	3.56	4.02	28.19	

image points than with the other methods. Generally, the computation time of the CHT is too high compared to the fastest methods.

The fastest of the non-probabilistic Hough transforms is the CFHT. The CFHT borrows the idea of the converging mapping from the RHT, and combines this part of the RHT framework to curve fitting, achieving a high computational speed via a many-to-one mapping. With pictures like those above, the CFHT fails to find several obvious lines. Therefore, the CFHT seems to be one of the most inaccurate methods. Some of the difficulties of the CFHT have already been discussed[25], and some improvements were suggested. We want to emphasize that results may vary with selected parameters.

Probabilistic methods

The ProbHT uses only a subset of image points in Hough transform calculation. According to test simula-

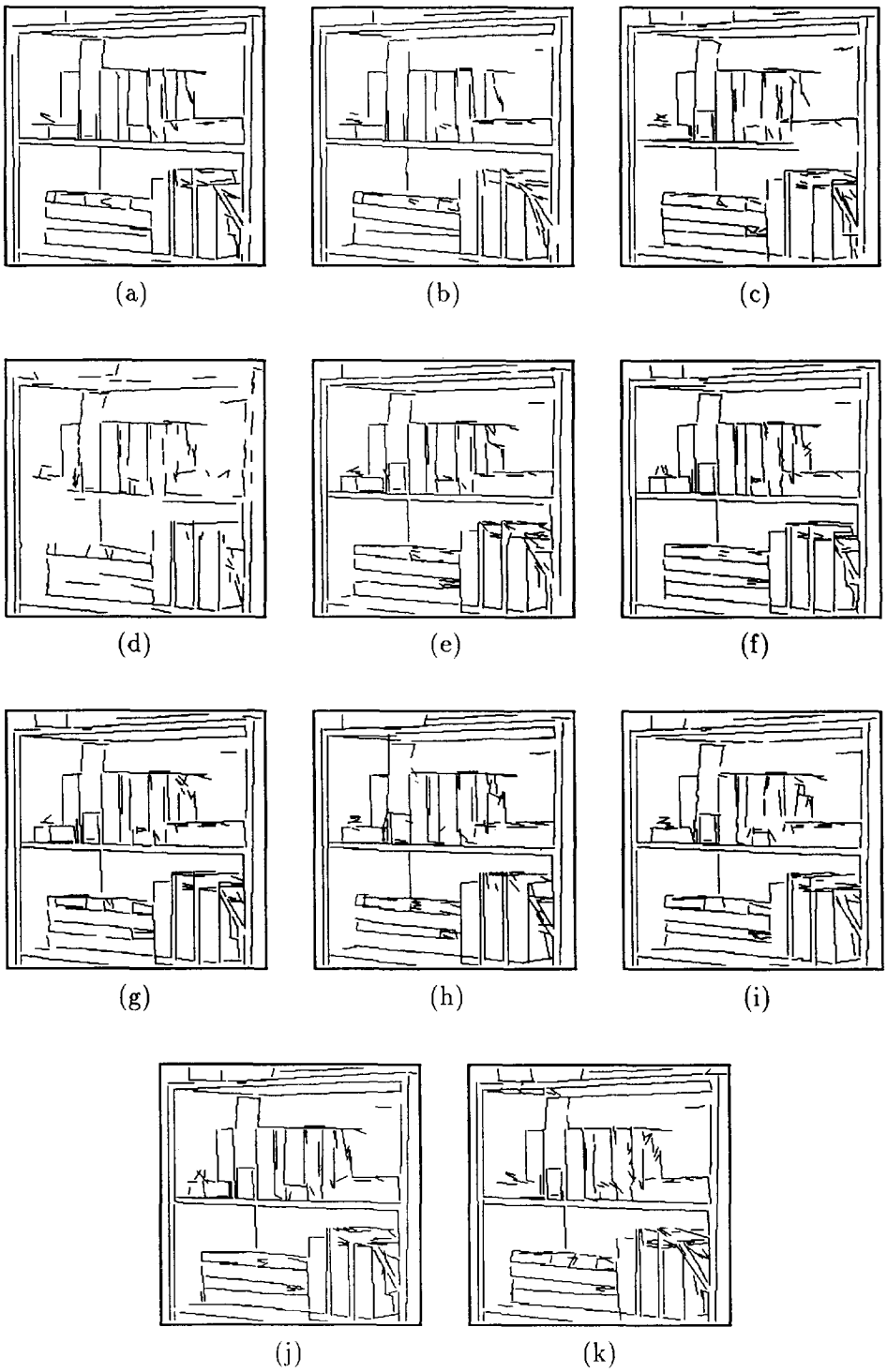


Figure 4 Resulting images of line detection from a complex real world image by (a) SHT, (b) ProbHT, (c) CHT, (d) CFHT, (e) RHT_ND, (f) RHT_D, (g) DRHT, (h) WRHT, (i) CRHT, (j) RWRHT, (k) DCHT

tions, this subset has to be quite large to obtain an accuracy similar to the SHT. However, computing is always faster. New improved strategies to apply the ProbHT are suggested by Mä-Jääski and Kiryati[42].

The DCHT is a simple and fast algorithm which gives reasonably good detection results. Also, it needs only a small amount of memory for the accumulator. However, the new extended RHT methods, CRHT, etc., having similar detection accuracy seem to be faster than the DCHT, especially with complex real-world images.

The performance of the RHT_ND depends greatly upon the number of lines in the image and on the noise level. If either of those increases, the computation time will increase rapidly because of waste accumulations. Detection rate also decreases dramatically, as presented in the test results section above. If an image is simpler containing, for instance, 10 lines, the RHT_ND clearly needs less computation and memory than the SHT.

The RHT_D has a clear advantage over the RHT_ND. Using the point distance criterion it successfully avoids the difficulties of the RHT_ND. Since the point distance is not allowed to be short, the accuracy of the method is much higher. Also, the limitation of very long distances makes the algorithm faster and less storage consuming. The DRHT gives a slightly better detection accuracy than the RHT_D at the cost of computation time and storage needed.

The RWRHT and the WRHT use more local information than the basic RHT. The RWRHT is highly adaptive, since its window size is changing and the number of RHT processes is also varying. The test results show that it is very fast and quite accurate. The local window can extract local lines more powerfully than line extraction from the whole image. The choice of the random window size still needs more analysis.

The WRHT and the CRHT also have low computation time and satisfactory accuracy. The problem of selecting a proper tolerance for the fitting error is the same as in the CFHT, but the idea of random sampling and converging mapping allows a smaller window size than in the CFHT, and thus the problems of too large size of the neighbourhood are avoided. The problems are also avoided by the connective component search of the CRHT. Both the WRHT and the CRHT can be used with accumulator threshold equal to one, i.e. no accumulator is needed. In fact, raising the threshold does not lead to significantly better results.

The RWRHT, WRHT and CRHT seem to exceed the power of the RHT_D, and thus are very promising approaches to further analysis. In particular, the CRHT was in all tests one of the fastest methods, obtaining quite satisfactory accuracy. In addition, the possibility of the WRHT and the CRHT to have no accumulator at all was interesting.

CONCLUSION

We have given an overview of recent developments of the Hough transform. The new versions were categorized as probabilistic or non-probabilistic. Their basic ideas were introduced and their properties compared. We presented new versions of the Randomized Hough Transform to detect curve segments. The new extensions, called the Dynamic RHT, the Random Window RHT, the Window RHT and the Connective RHT, were tested for line detection in synthetic and real-world images and compared to several Hough transforms. The RWRHT, WRHT and CRHT gave promising results.

ACKNOWLEDGEMENT

The authors thank Mr P Kultanen for his contributions to the development of the Dynamic RHT.

REFERENCES

1 Illingworth, J and Kittler, J 'A survey of the Hough transform', *Comput. Vision, Graph. & Image Process.*, Vol 44 (1988) pp 87–116

2 Fischler, M A and Firschein, O 'Parallel guessing: A strategy for high-speed computation', *Patt. Recogn.*, Vol 20 No 2 (1987) pp 257–263

3 Xu, L, Oja, E and Kultanen, P 'A new curve detection method: Randomized Hough Transform (RHT)', *Patt. Recogn. Lett.*, Vol 11 No 5 (1990) pp 331–338

4 Kultanen, P, Xu, L and Oja, E 'Randomized Hough Transform (RHT)', *Proc. 10th Int. Conf. Patt. Recogn.*, Atlantic City, NJ (June 1990) pp 631–635

5 Xu, L and Oja, E 'Further developments on RHT: Basic mechanisms, algorithms, and computational complexities', *Proc. 11th Int. Conf. Patt. Recogn.*, The Hague, Netherlands (August–September 1992) pp 125–128

6 Xu, L and Oja, E 'Randomized Hough Transform (RHT): Basic mechanisms, algorithms, and computational complexities', *GVGIP: Image Understanding*, Vol 57 No 2 (1993) pp 131–154

7 Kiryati, N, Eldar, Y and Bruckstein, A M 'A probabilistic Hough transform', *Patt. Recogn.*, Vol 24 No 4 (1991) pp 303–316

8 Shvaytser, H and Bergen, J R 'Monte Carlo Hough transforms', *Proc. 7th Scandinavian Conf. Image Analysis*, Aalborg, Denmark (August 1991) pp 80–87

9 Bergen, J R and Shvaytser, H 'A probabilistic algorithm for computing Hough transforms', *J. Algorithms*, Vol 12 No 4 (1991) pp 639–656

10 Roth, G and Levine, M D 'Extracting geometric primitives', *CVGIP: Image Understanding*, Vol 58 No 1 (1993) pp 1–22

11 Shiono, M 'Arc detection by Hough transform using RCIH method', *Proc. 1st Korea–Japan Joint Conf. Comput. Vision*, Seoul, Korea (October 1991) pp 60–67

12 Ben-Tzvi, D, Leavers, V F and Sandler, M B 'A dynamic combinatorial Hough transform', *Proc. 5th Int. Conf. Image Analysis and Processing*, Positano, Italy (September 1989) pp 152–159

13 Leavers, V F, Ben-Tzvi, D and Sandler, M B 'A dynamic combinatorial Hough Transform for straight lines and circles', *Proc. 5th Alvey Vision Conf.*, Reading, UK (September 1989) pp 163–168

14 Leavers, V F 'The dynamic generalized Hough transform: Its relationship to the probabilistic Hough transforms and an application to the concurrent detection of circles and ellipses', *CVGIP: Image Understanding*, Vol 56 No 3 (1992) pp 381–398

15 Yuen, S Y K, Lam, T S L and Leung, N K D 'Connective Hough transform', *Image & Vision Comput.*, Vol 11 No 5 (1993) pp 295–301

16 Lam, L T S, Lam, W C Y and Leung, D N K 'A Hough-like prediction/correction approach for circle detection', *Proc. 8th Scandinavian Conf. Image Analysis*, Tromsø, Norway (May 1993) pp 151–158

17 Lam, L T S, Lam, W C Y and Leung, D N K 'A Hough-like prediction/correction approach for ellipse detection', *Proc. 5th Int. Conf. Computer Analysis of Images and Patterns*, CAIP'93, Budapest, Hungary (September 1993) pp 175–182

18 Gerig, G and Klein, F 'Fast contour identification through efficient Hough transform and simplified interpretation strategy', *Proc. 8th Int. Conf. Patt. Recogn.*, Paris, France (October 1986) pp 498–500

19 Gerig, G 'Linking image space and accumulator-space: A new approach for object recognition', *Proc. 1st Int. Conf. Comput. Vision*, London, UK (June 1987) pp 112–117

20 Li, H, Lavin, M A and Le Master, R J 'Fast Hough transform: A hierarchical approach', *Computer Vision, Graph. & Image Process.*, Vol 36 (1986) pp 139–161

21 Illingworth, J and Kittler, J 'The adaptive Hough transform', *IEEE Trans. PAMI*, Vol 9 No 5 (1987) pp 690–698

22 Princen, J, Illingworth, J and Kittler, J 'A hierarchical approach to line extraction', *Proc. IEEE Computer Vision and Patt. Recogn. Conf.*, San Diego, CA (June 1989) pp 92–97

23 Risse, T 'Hough transformation for line recognition: Complexity of evidence accumulation and cluster detection', *Comput. Vision, Graph. & Image Process.*, Vol 46 No 46 (1989) pp 327–345

24 Liang, P 'A new transform for curve detection', *Proc. 3rd Int. Conf. Comput. Vision*, Osaka, Japan (December 1990) pp 748–751

25 Liang, P 'A new and efficient transform for curve detection', *J. Robotic Syst.*, Vol 8 No 6 (1991) pp 841–847

26 Stephens, R S 'Probabilistic approach to the Hough transform', *Image & Vision Comput.*, Vol 9 No 1 (February 1991) pp 66–71

27 Ben-Tzvi, D and Sandler, M B 'A combinatorial Hough transform', *Patt. Recogn. Lett.*, Vol 11 No 3 (1990) pp 167–174

28 Costa, L D F, Ben-Tzvi, D and Sandler, M B 'Performance improvements to the Hough transform', *UK IT 1990 Conf.*, Southampton, UK (March 1990) pp 98–103

29 Kohonen, T *Self-Organization and Associative Memory*, Springer-Verlag, Berlin (1984)

30 Xu, L and Oja, E *Extended self-organizing map for curve detection*, Research Report No 16, Department of Information Technology, Lappeenranta University of Technology, Finland (1989)

31 Kultanen, P and Oja, E 'Randomized Hough Transform (RHT) in engineering drawing vectorization system', *Proc. IAPR Workshop on Machine Vision Applic.*, Tokyo, Japan (November 1990) pp 173–176

32 Kälviäinen, H, Oja, E and Xu, L 'Motion detection using randomized Hough Transform', *Proc. 7th Scandinavian Conf. Image Analysis*, Aalborg, Denmark (August 1991) pp 72–79

33 Kälviäinen, H, Oja, E and Xu, L 'Randomized Hough Transform applied to translational and rotational motion analysis', *Proc. 11th Int. Conf. Patt. Recogn.*, The Hague, The Netherlands (August–September 1992) pp 672–675

34 Kälviäinen, H 'Computational considerations on randomized Hough transform and motion analysis', in G Vernazza *et al.* (eds), *Image Processing: Theory and Applications*, Elsevier, Amsterdam (1993) pp 87–90

35 Leavers, V F 'Which Hough transform?', *CVGIP: Image Understanding*, Vol 58 No 2 (1993) pp 250–264

36 Princen, J, Yuen, H K, Illingworth, J and Kittler, J 'A comparison of Hough transform methods', *3rd Int. Conf. Image Analysis and Its Applic.*, Warwick, UK (July 1989) pp 73–77

37 Yuen, H K, Princen, J, Illingworth, J and Kittler, J 'Comparative study of Hough transform methods for circle finding', *Image & Vision Comput.*, Vol 8 No 1 (1990) pp 71–77

38 Hare, A R and Sandler, M B 'General test framework for straight-line detection by Hough transforms', *IEEE Int. Symposium on Circuits and Systems*, ISCAS'93, Chicago, IL (May 1993) pp 239–242

39 Kälviäinen, H, Hirvonen, P, Xu, L and Oja, E 'Comparisons of probabilistic and non-probabilistic Hough transforms', *Proc. 3rd Euro. Conf. Comput. Vision*, EECV'94, Stockholm, Sweden (May 1994) pp 351–360

40 Kälviäinen, H, Xu, L and Oja, E *Recent versions of the Hough transform and the randomized Hough transform: Overview and comparisons*, Research Report No 37, Department of Information Technology, Lappeenranta University of Technology, Finland (1993)

41 Kälviäinen, H and Hirvonen, P *Connective Randomized Hough Transform (CRHT)*, Research Report No 44, Department of Information Technology, Lappeenranta University of Technology, Finland (1993)

42 Ylä-Jääski, A and Kiryati, N 'Automatic termination rules for probabilistic Hough algorithms', *Proc. 8th Scandinavian Conf. Image Analysis*, Tromsø, Norway (May 1993) pp 121–128

43 Fischler, M A and Bolles, R C 'Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography', *Graphics Image Process.*, Vol 24 No 6 (1981) pp 381–395

44 Kälviäinen, H *Motion analysis by the RHT method in computer vision*, Licentiate Thesis, Department of Information Technology, Lappeenranta University of Technology, Finland (1992)

45 Kultanen, P, Xu, L and Oja, E *Generalized Randomized Hough Transform (GRHT)*, Research Report No 20, Department of Information Technology, Lappeenranta University of Technology, Finalnd (1990)

46 Princen, J, Yuen, H K, Illingworth, J and Kittler, J 'Properties of the adaptive Hough transform', *Proc. 6th Scandinavian Conf. Image Analysis*, Oulu, Finland (June 1989) pp 613–620

47 Princen, J, Illingworth, J and Kittler, J 'A hierarchical approach to line extraction based on the Hough transform', *Comput. Vision, Graph. & Image Process.*, Vol 52 No 1 (1990) pp 57–77

48 Kälviäinen, H *Randomized Hough Transform: New extensions* Doctoral Thesis, Research Papers 35, Department of Information Technology, Lappeenranta University of Technology, Finland (1994)

49 Duda, R O and Hart, P E 'Use of the Hough transform to detect lines and curves in pictures', *Commun. ACM*, Vol 15 No 1 (1972) pp 11–15