Analytic Curve Detection from a Noisy Binary Edge Map using Genetic Algorithm

Samarjit Chakraborty[1] and Kalyanmoy Deb[⋆2]

[1] Department of Computer Science and Engineering
[2] Department of Mechanical Engineering
Indian Institute of Technology Kanpur
Kanpur 208016, India

Abstract. Currently Hough transform and its variants are the most common methods for detecting analytic curves from a binary edge image. However, these methods do not scale well when applied to complex noisy images where correct data is very small compared to the amount of incorrect data. We propose a Genetic Algorithm in combination with the Randomized Hough Transform, along with a different scoring function, to deal with such environments. This approach is also an improvement over random search and in contrast to standard Hough transform algorithms, is not limited to simple curves like straight line or circle.

1 Introduction

Extracting curves from a binary edge image is an important problem in computer vision and robotics. The Hough transform (HT) [7, 21] is recognized as a powerful tool to handle this. Although it gives good results in the presence of small amounts of noise and occlusion, it does not scale well when applied to complex, cluttered scenes, with lot of noise. In a study on the noise sensitivity of the generalized HT by Grimson and Huttenlocher [5], it was concluded that even for moderate amounts of noise and occlusion, these methods can hypothesize many false solutions, and their effectiveness is dramatically reduced. So these techniques are reliable only for relatively simple tasks, where the edge data corresponding to correct solutions is a large fraction of the total data. We confirm this finding in the case of simpler Hough transforms also, used for detecting analytic curves like straight lines and circles. Based on this, we propose a different approach, using a Genetic Algorithm (GA) [4] in combination with the Randomized Hough Transform (RHT) [25, 26] but using a different scoring function than usual. A number of other researchers have considered the problem of detecting lines or other simple curves in noisy images [3, 8, 19, 22, 24]. However, all of them assume a uniform noise distribution and in some cases knowledge about the distribution. Califano *et al.* [2] and Leavers [17] attempted to deal with the effects of correlated noise by preferential segmentation of the image with respect to the shape under detection. Connective HT [27] is another method used in the case of correlated noise. But the method that we present is much more robust

⋆ Author to whom all correspondence should be directed.

A.E. Eiben et al. (Eds.): PPSN V, LNCS 1498, pp. 129-138, 1998.

than any of these and makes no assumption about the noise distribution. It is particularly effective in the case of complex, cluttered and noisy images where the number of pixels belonging to genuine curves is a very small fraction of the total data.

Another problem of the HT is that its computational complexity and storage requirements increase exponentially with the number of parameters of the curve to be detected. In spite of the large number of papers which address this issue, straight lines and circles are still the only curves for which HT can be effectively used. Comprehensive reviews of the development in this field can be found in references [9] and [16].

To alleviate these problems, recently a new kind of approach in the form of a set of methods called Probabilistic Hough Transforms has been developed. Most of them use random sampling of the image data in various ways. A good review of these techniques along with comparisons with standard methods is done in reference [11]. However our method can be used for any analytic curve detection without incurring any additional computational or storage requirements, over those required for simple curves like straight line and circle. The method can be used independently of the RHT, but in the case of images with low noise, combination with RHT yields a considerable speedup.

The concept of using GA for curve extraction has been explored in the past [6, 23]. However the problem of noise was not addressed. Moreover, much simpler cases were considered than we do in this paper.

In the next section we briefly describe the Hough transform, its randomized variant, the RHT, and identify cases where these methods fail due to the presence of excessive noise and clutter, following which we describe our method. In Section 4 we give test results with a complex noisy image and compare the performance with known methods. Section 5 concludes the paper.

2 Motivation

In the conventional implementation, the Hough transform essentially consists of two stages. The first stage is based on a transformation mapping from each edge point to points in the parameter space, represented by an accumulator array, and a voting rule which determines how the transformation mapping affects the contents of the accumulator array. The second stage is an exhaustive search for parameters in the accumulator array which are local maxima. Each such local maximum represents a candidate curve in the edge map.

In the probabilistic versions of the Hough transform, mainly two approaches are used. In the first, due to Kiryati *et al.* [15], image data are randomly sampled and only the sampled subset of points is transformed. In the second approach due to Xu *et al.* [25, 26], Leavers *et al.* [17], Bergen *et al.* [1], and Califano *et al.* [2], for a curve with n parameters, n image points are sampled, the equation of the curve passing through these n points is determined and the corresponding point in the parameter space is incremented.

In all these approaches, points on the same curve result in points in the parameter space which are close together, whereas noise points result in randomly distributed points in the parameter space. Thus a large cluster of points in the parameter space represent a curve in the edge map. The validity of this assumption, however, depends on there being a low likelihood that clusters due to noise points will be comparable or larger in size than clusters due to points on genuine curves. We believe that in many real life images, this assumption does not hold. Fig. 1(a) shows two straight lines L_1 and L_2, where each line is composed of a small number of disconnected points. In Fig. 1(b), random noise is superimposed on the line L_1 (Fig. 4 in Section 4 shows one example where such a situation really arises in practice). Let us call the lines in Fig. 1(a) as *true lines* and the line in Fig. 1(b) that corresponds to line L_1 of Fig. 1(a), as a *pseudo line*. Line L_2 in this figure still remains a true line. Ideally the detection algorithm should detect both L_1 and L_2 from Fig. 1(a) but only L_2 from Fig. 1(b). Note that there are a large number of pseudo lines in the noise region in Fig. 1(b). Since the number of points on each of these pseudo lines is comparable or more than than the number of points on the line L_2, it gets masked in the parameter space by these pseudo lines.

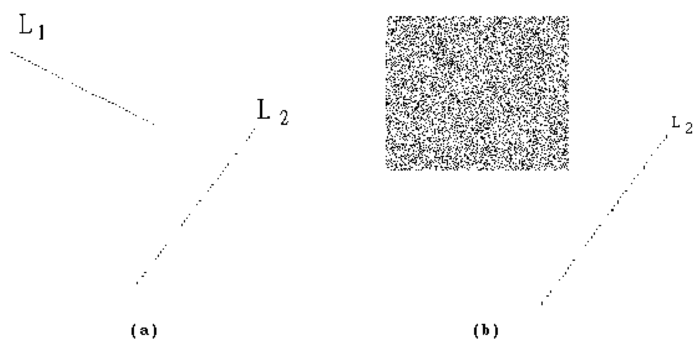

Fig. 1. A binary edge image (a) Two straight lines (b) Noise superimposed on one of the lines

To overcome the effects of noise, extensions of the basic Randomized Hough Transform has been proposed in [10]. These are Window RHT and Random Window RHT. Both of them randomly place a window on an edge point and try to locate a curve within the window. Although it is possible that they will work in Fig. 1(b), the probability that the window is placed on the line L_2 is extremely small. Moreover, when the window is placed on the noise region, pseudo lines will be detected.

The Connective Hough Transform [17, 18, 27] and its randomized version, the Connective Randomized Hough Transform [13, 14] tries to make use of the connectivity between points on a true curve to distinguish between true and pseudo curves. So they fail when the points on the curve are not connected i.e. there are gaps between the edge points, as in the case of line L_2. Such cases do arise in practice, for example in bubble-chamber photographs of particle tracks.

3 A Genetic Algorithm for Analytic Curve Detection

The kernel of our algorithm is conceptually similar to the Window RHT of Kälviäinen *et al.* [10]. But instead of placing a 'window' on a randomly chosen edge point, we place a weighted mask on the edge point. The mask measures the weighted difference between pixels on a real curve and the noise surrounding the curve. The placement of the mask is guided by a genetic algorithm. Since we are considering cases where the number of edge points on real curves is very small compared to the total number of edge points, this leads to an improvement over a simple random search. A Randomized Hough Transform with a very low parameter resolution and a smaller number of sample points than usual, is used to identify prospective regions of the parameter space. The genetic algorithm searches the entire parameter space with a bias towards these regions. For simple curves with low noise, this leads to a considerable speedup.

3.1 The Weighted Mask

Given a binary edge map and the family of curves being sought (such as straight line, circle, ellipse, etc.), the algorithm should produce the set of curves from that family which appears in the image. Let the family of curves being sought be given by $f(\mathcal{A}, \bar{x}) = 0$, where $\mathcal{A} = (a_1, a_2, \ldots, a_n)$ denotes the parameter vector. We say that a curve segment with parameter values $\mathcal{A}_\alpha$ occurs in the given image at location $\bar{x}'$ if $\sum_{\bar{x}\in\mathcal{M}} Z_{\bar{x}} \geq N_{min}$, where $Z_{\bar{x}}$ is the gray level of the pixel $\bar{x}$ (0 or 1 in a binary edge map), $\mathcal{M} = \{\bar{x} : |f(\mathcal{A}_\alpha, \bar{x})| \leq \delta \text{ and } d(\bar{x}, \bar{x}') \leq D\}$, $d(\bar{x}, \bar{x}')$ is the Eucledian distance between the points $\bar{x}$ and $\bar{x}'$, and δ, D and N_{min} are parameters defined by the algorithm. The set $\mathcal{M}$ denotes a mask centered on the pixel $\bar{x}'$ and has length D and width 2δ. N_{min} is the minimum number of edge points that must occur within the mask so that the presence of a curve segment located at $\bar{x}'$, having parameters $\mathcal{A}_\alpha$, can be ascertained.

For images with low or no noise such as Fig. 1(a), this formulation is sufficient, and is in fact similar to the Window RHT, except for the fact that we do not use any transformation from the image to the parameter space but rather simply count the number of points lying within the window or mask. But in the case of noisy images such a Fig. 1(b), whenever the mask is placed on the noise region, pseudo curves will be detected. To extend this method to include such images, we use a weighted mask rather than a simple one. The response of the mask defined with respect to its center location is given by $R = \sum_{\bar{x}\in\mathcal{M}} W_{\bar{x}} Z_{\bar{x}}$, where $W_{\bar{x}}$ is the mask coefficient of the pixel $\bar{x}$. We shall say that a curve segment with parameter values $\mathcal{A}_\alpha$ occurs in the edge map at $\bar{x}'$ if the response, R, of the mask centered at $\bar{x}'$ is greater than a constant R_{min}, fixed, depending on the mask length, width and coefficients. The mask coefficients of the pixels that lie away from the curve $f(\mathcal{A}_\alpha, \bar{x}') = 0$ are assigned negative values. So when a lot of noise is present near a curve, as in the case of pseudo curves, the positive response due to the points on and near the curve is offset by the negative response due to the noise points surrounding it. Hence pseudo curves are not detected. An example of a weighted mask for straight line detection is shown in Fig. 3 in Section 4.

3.2 Using Genetic Algorithm

Instead of placing the mask on a randomly chosen edge point, as done in Window RHT, we use a genetic algorithm to search the space $(\mathcal{A}, \bar{x})$ for all instances of curve segments for which the response of the mask is greater than R_{min}. For this, each of the parameters $a_1, a_2, \ldots, a_n$ of $\mathcal{A}$ and x of $\bar{x}$ are coded as fixed length binary strings. Since the y-coordinate of $\bar{x}$ is the dependent variable, it is not included in the string. The resulting string, obtained by concatenating all these strings, gives the chromosomal representation of a solution to the problem. Note that the domains of each of the parameters may be different and the length of the string coding for a given parameter depends on the required parameter resolution. The fitness of a solution $(\mathcal{A}_\alpha, \bar{x}\,')$ is taken to be the response of the weighted mask, centered at $\bar{x}\,'$, as described in the previous section.

[Step 1] **Creation of initial population and the use of Randomized Hough Transform.** In most GA applications, the initial population consists of entirely random structures to avoid convergence to a local optima. But in this problem, the question is not of finding the global optima, but of finding all solutions with fitness greater than R_{min}. To identify prospective regions of the search space, Randomized Hough Transform with a low parameter resolution and a smaller number of trials than usual, is used.

For a curve expressed by a n-parameter equation, n points are randomly chosen from the edge data and n joint equations are solved to determine one parameter point $\mathcal{A}_\alpha$. In the accumulator corresponding to the parameter space, the *count* of $\mathcal{A}_\alpha$ is then incremented by one. After repeating this process for a predefined number of times, points in the parameter space with counts exceeding a predefined threshold, indicate prospective curves in the image space. Whereas only the global maximum of the parameter space is used in RHT, here, all points with counts exceeding a specific threshold are used. So, the threshold value used here is much less than what is used in RHT. Corresponding to each of these points in the parameter space, a suitable number of solutions, proportional to the count values, with the x-coordinates randomly chosen, are introduced into the initial population. To reduce the effect of the sampling error, due to the low threshold value and hence smaller number of samples, a fixed number of random samples from the solution space are also introduced. The total number of solutions is kept fixed over all the generations.

[Step 2] **Selection.** The selection used here falls into the category of dynamic, generational, preservative, elitist selection [20]. Let there be M distinct solutions in a given generation, denoted by $S_1, S_2, \ldots, S_M$. The probability of selecting a solution S_i into the mating pool is given by :

$$P(S_i) \;=\; \frac{\mathcal{F}(S_i)}{\sum_{j=1}^{M} \mathcal{F}(S_i)}$$

Where $\mathcal{F}(S_i)$ is the fitness of the solution S_i. A fixed number of solutions are copied into the mating pool according to this rule and the remaining solutions are newly created from parameter values indicated by the RHT with the x-

coordinate randomly chosen. In each generation, a fixed number of best solutions of the previous generation are copied in place of the present worst solutions, if they happen to be less fit compared to the former. This is a slight modification of the Elitist model where only the best solution is preserved.

[Step 3] **Crossover.** Since the number of parameters may not be small, it is intuitive that the single point crossover operation may not be useful. So crossover is applied to each substring corresponding to each of the parameters $a_1, a_2, \ldots, a_n$, and x, the operation being the usual swapping of all bits from a randomly chosen crossover site of the two parents, chosen randomly from the mating pool [4]. Hence this crossover is similar to the standard single-point crossover operator, but operated on substrings of each parameter. Therefore, there are $n+1$ single-point crossovers taking place between two parent strings.

[Step 4] **Mutation.** We have used two mutation operators. The first is the classical mutation operation in which each bit position of the solution strings is complemented with a small mutation probability p_{mut1}. The second operation is as follows : let $b_l\, b_{l-1}\, \ldots\, b_0$ be the binary substring corresponding to a parameter a_i whose domain is $[\alpha_i, \beta_i]$, and whose real value is decoded as c. In the parameter space corresponding to the RHT implementation, let the parameter a_i be quantized as $q_1, q_2, \ldots, q_{i_N}$ ($\alpha_i \leq q_1$ and $q_{i_N} \leq \beta_i$) and let $c \in [q_j, q_{j+1}]$. Then after mutation we get c', where c' is chosen uniformly from $[q_j, q_{j+1}]$. So, the mutation of the binary string $b_l\, b_{l-1}\, \ldots\, b_0$ results in the binary string corresponding to the real value c'. This mutation operation is applied with a probability p_{mut2} on each substring corresponding to the parameters $a_1, a_2, \ldots, a_n$, to take care of the low parameter resolution used in the RHT.

The overall algorithm. The initial population consisting of a fixed number of solutions is created as already described. In each generation, the entire population is subjected to selection, crossover and the first mutation. The second mutation operation is applied just after the selection process, only to the solutions generated by the RHT technique.

At the end of each generation, curve segments corresponding to solutions having fitness greater than R_{min} are removed from the edge map and after fixed number of generations, the accumulators are reset and RHT is again applied to the current edge map, so that the solutions corresponding to the removed curve segments are dropped from the population. This iteration is continued until no new curve segments are extracted for a given number of generations, which in our experiments was set to 200. A schematic diagram explaining the algorithm is shown in Fig. 2.

4 Test Results and Comparisons

We have experimented with a complex real world image containing a lot of noise points and a few straight lines. Although the simplest, straight line detection

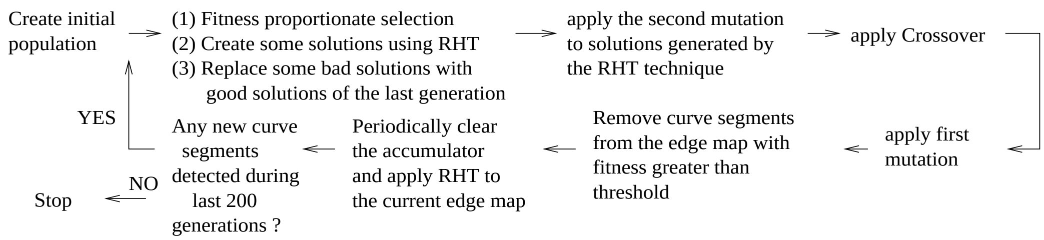

Fig. 2. The proposed algorithm

was chosen for ease of comparison with the various Hough transform algorithms. However, as evident from the previous section, our algorithm is blind to this fact. For comparing the performance of our method with HT, we used a public domain software package for line detection, XHoughtool [12], where a number of non-probabilistic and probabilistic Hough transform algorithms have been implemented.

As indicated in the previous section, there are various parameters that our algorithm uses. Parameters related to the mask are its dimensions, mask coefficients and the threshold response R_{min}. The mask length has an effect on the minimum allowable size of a curve segment, and the quality of the curves detected are determined by the mask coefficients and its width. A wide mask with more than one row of positive coefficients will detect curves whose pixels are spread out along its width. Thus a suitably designed mask, along with a proper threshold value, will be able to distinguish between fuzzy and spread out curves, and noise regions. We have used a mask length of 100 and width 3, to detect only perfect straight lines. The coefficients of all pixels lying on the straight line were set to 2 and the others to -1 as shown in Fig. 3. Too low a value, R_{min}, of the threshold might detect a pseudo line whereas a too high value might miss a faint, disconnected, but visually detectable line. The results shown in this section were obtained with R_{min} set to 100.

-1	**-1**	**-1**	**.......**	**-1**	**-1**	**-1**
2	**2**	**2**	**.......**	**2**	**2**	**2**
-1	**-1**	**-1**	**.......**	**-1**	**-1**	**-1**

Fig. 3. A mask of length 100 and width 3

For the GA parameters, we used mutation probabilities p_{mut1} and p_{mut2} to be 0.1 and 0.5 respectively. Any population size around 100 was found to work well, and in each generation, 25% of the solutions were created using the RHT and the rest copied from the mating pool in accordance with the fitness proportional selection. Further, the best 10% solutions of the previous generation were copied in place of the worst solutions of the current generation. We used the ρ, θ parametric representation of a line, where each line is represented by $x\cos(\theta) + y\sin(\theta) = \rho$. Each of θ, ρ and x were coded as binary strings and concatenating them gives the chromosomal representation of a potential solution.

Fig. 4(b) shows a 512 by 512 binary image obtained after edge detection of the corresponding grayscale image shown in Fig. 4(a). Note the three disconnected, but visible real lines in the image, two at the centre and one the the extreme left end. The straight lines detected by our algorithm are shown in Fig. 4(c). Altogether seven different Hough transform algorithms are implemented in the XHoughtool package. In spite of a serious attempt being made to select the test parameters for each method as optimally as possibly, none of the algorithms gave useful results because a large number of pseudo lines were detected. A typical result is shown in Fig. 4(d). Since the number of edge points lying on the real lines are much less compared to those lying on many of the pseudo lines, no suitable accumulator threshold value exists which can detect only the real lines. Generally these algorithms work well even in the case of noisy images, where the lines are connected and the number of edge points lying on these lines are atleast comparable to the number of noise points.

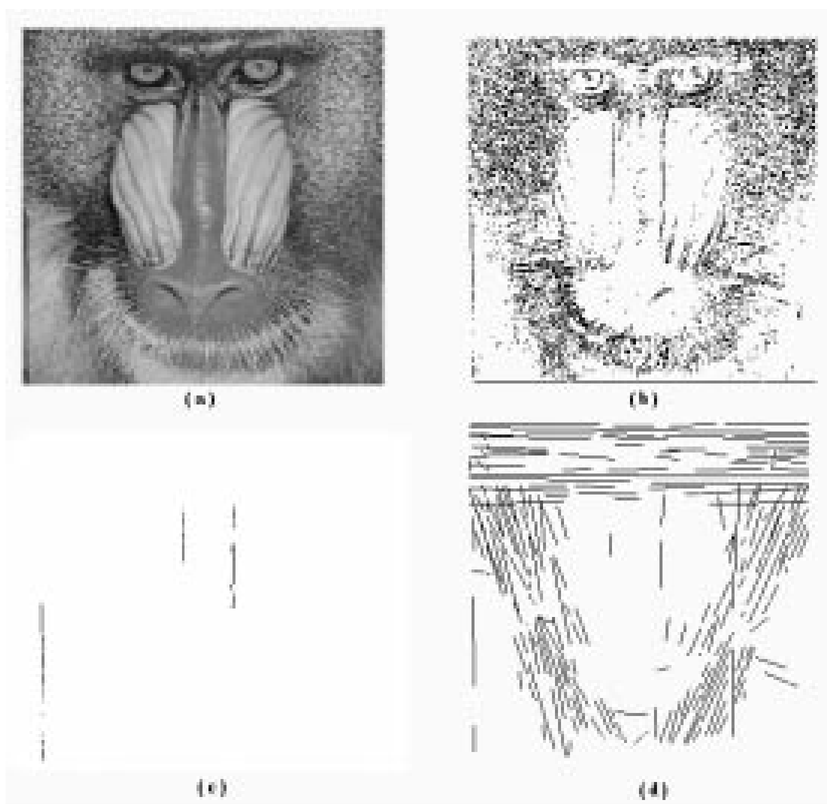

Fig. 4. Test results (a) A 512 by 512 gray scale image (b) The corresponding binary edge map obtained after edge detection (c) Straight lines detected by the proposed method (d) A typical result obtained using a Hough transform algorithm

This example was executed on a moderately loaded DEC Alpha 2000 4/233

(Dual Processor system) workstation and required around ten seconds to complete.

5 Conclusion

The Hough transform and its variants are still the most popular methods for detecting analytic curves from binary edge data. However, we conclude that they can be effectively used only in cases where the noise ratio is not too high. The main bottlenecks of the standard Hough transform are its computational complexity and storage requirements. Hence it is rarely used beyond circle detection. Some efforts to deal with these problems have led to parameter space decomposition, parallel architectures, and probabilistic techniques. In this paper we have seen that apart from its use in complex noisy images, the GA approach can also help to deal with these problems to some extent. Moreover, the concept of using the weighted mask introduced a flexibility in the quality of the curves to be detected, which is difficult, if not impossible to achieve using conventional Hough transform algorithms.

The probabilistic Hough transforms work well in the case of complex images; however, in the kind of data we considered in this paper, a search guided by a GA is probably superior to a simple random search for good edge points. Further, as also mentioned by Roth and Levine [23], another main advantage of using GA is the ease with which it can be implemented over a wide variety of parallel architectures compared to Hough transform, which is more difficult to parallelize.

References

1. Bergen, J. R., Shvaytser, H. : A probabilistic algorithm for computing Hough transforms. J. Algorithms, **12**, 4 (1991) 639–656
2. Califano, A., Bolle, R. M., Taylor, R. W. : Generalized neighbourhoods : A new approach to complex parameter feature extraction. Proc. IEEE Conference on Computer Vision and Pattern Recognition, (1989) 192–199
3. Cohen, M., Toussaint, G. T. : On the detection of structures in noisy pictures. Pattern Recognition, **9**, (1977) 95–98
4. Goldberg, D. E. : Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, M.A. (1989)
5. Grimson, W. E. L., Huttenlocher, D. P. : On the sensitivity of the Hough transform for object recognition. IEEE Trans. Pattern Anal. Machine Intell., **PAMI-12**, (1990) 255–274
6. Hill, A., Taylor, C. J. : Model-based image interpretation using genetic algorithms. Image and Vision Computing, **10**, (1992) 295–300
7. Hough, P. V. C. : Method and means for recognizing complex patterns. U.S. Patent No. 3069654 (1962)
8. Hunt, D. J., Nolte, L. W., Reibman, A. R., Ruedger, W. H. : Hough transform and signal detection theory performance for images with additive noise. Computer Vision, Graphics and Image Processing, **52**, 3 (1990) 386–401

9. Illingworth, J. and Kittler, J. : A survey of the Hough transform. Computer Vision, Graphics and Image Processing, **44**, (1988) 87–116
10. Kälviäinen, H., Xu, L., 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)
11. Kälviäinen, H., Hirvonen, P., Xu, L., Oja, E. : Probabilistic and non-probabilistic Hough transforms : overview and comparisons. Image and Vision Computing, **13**, 4 (1995) 239–252
12. Kälviäinen, H., Hirvonen, P., Oja, E. : Houghtool–a Software Package for Hough Transform Calculation. Proceedings of the 9th Scandinavian Conference on Image Analysis, Uppsala, Sweden, (June 1995) 841–848 (http://www.lut.fi/dep/tite/XHoughtool/xhoughtool.html)
13. Kälviäinen, H., Hirvonen, P. : Connective Randomized Hough Transform (CRHT). Proc. 9th. Scandinavian Conference on Image Analysis, Uppsala, Sweden (June 1995).
14. Kälviäinen, H., Hirvonen, P. : An extension to the Randomized Hough Transform exploiting connectivity. Pattern Recognition Letters, **18**, 1 (1997) 77–85
15. Kiryati, N., Eldar, Y. Bruckenstein, A. : A probabilistic Hough transform. Pattern Recognition, **24**, 4 (1991) 303-316
16. Leavers, V. F. : Which Hough Transform ? CVGIP: Image Understanding, **58**, 2 (1993) 250–264
17. Leavers, V. F. : It's probably a Hough: 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, **56**, 3, (1992) 381–398
18. Liang, P. : A new and efficient transform for curve deection. J. of Robotic Systems, **8**, 6 (1991) 841–847
19. Maitre, H. : Contribution to the prediction of performances of the Hough transform. IEEE Trans. Pattern Anal. Machine Intell., **PAMI-8**, 5 (1986) 669–674
20. Michalewicz, Z. : Genetic Algorithms + Data Structutes = Evolution Programs. Springer Verlag, Berlin (1992)
21. Princen, J., Illingworth, J., Kittler, J. : A formal definition of the Hough transform : properties and relationships. J. Math. Imaging Vision, **1**, (1992) 153–168
22. Risse, T. : Hough transform for the line recognition: complexity of evidence accumulation and cluster detection. Computer Vision, Graphics and Image Processing, **46**, (1989) 327
23. Roth, G., Levine, M. D. : Geometric primitive extraction using a genetic algorithm. IEEE Trans. Pattern Anal. Machine Intell., **PAMI-16**, 9 (1994) 901–905
24. Shapiro, S. D. : Transformations for the computer detection of curves in noisy pictures. Computer Graphics Image Processing, **4**, (1975) 328–338
25. Xu, L., Oja, E., Kultanen, P. : A new curve detection method : Randomized Hough transform (RHT). Pattern Recognition Letters, **11**, 5 (1990) 331–338
26. Xu, L., Oja, E. : Randomized Hough Transform (RHT) : Basic mechanisms, algorithms, and computational complexities. CVGIP : Image Understanding, **57**, 2 (1993) 131–154
27. Yuen, K. S. Y., Lam, L. T. S., Leung, D. N. K.: Connective Hough Transform. Image and Vision Computing, **11**, 5 (1993) 295–301

This article was processed using the LaTeX macro package with LLNCS style