Progressive Probabilistic Hough Transform for line detection

C. Galambos[†] J. Matas[†§] and J. Kittler[†]

[†]CVSSP,
University of Surrey,
Guildford, Surrey GU2 5XH,
United Kingdom

[§]Centre for Machine Perception,
Czech Technical University,
Karlovo náměstí 13, 12135 Praha,
Czech Republic

e-mail: C.Galambos@ee.surrey.ac.uk

Abstract

We present a novel Hough Transform algorithm referred to as Progressive Probabilistic Hough Transform (PPHT). Unlike the Probabilistic HT [7] where Standard HT is performed on a pre-selected fraction of input points, PPHT minimises the amount of computation needed to detect lines by *exploiting the difference in the fraction of votes needed to detect reliably lines with different numbers of supporting points.* The fraction of points used for voting need not be specified ad hoc or using a priori knowledge, as in the probabilistic HT; it is a function of the inherent complexity of the input data.

The algorithm is ideally suited for real-time applications with a fixed amount of available processing time, since voting and line detection is interleaved. The most salient features are likely to be detected first. Experiments show that in many circumstances PPHT has advantages over the Standard HT.

1 Introduction

In this paper we present a novel Hough Transform algorithm for detecting straight lines, its underlying theory and experimental validation. The algorithm is referred to as the Progressive Probabilistic Hough Transform (PPHT) which reflects the progressive nature of the process of line detection inherent to the algorithm, which finds the longest (most salient) first and proceeds to the shortest lines.

The Hough Transform (HT) is a popular method of extraction of geometric primitives. In literally hundreds of papers every aspect of the transform has been scrutinised - parametrisation, accumulator design, voting patterns, peak detection - to name but a few [5]. The introduction of the randomised version of Hough Transform is one of the most important recent developments [14], [7] in the field. The approach proposed in the paper falls in the 'probabilistic' (or Monte Carlo) class of HT algorithms, the objective of which is to minimise the proportion of points that are used in voting while maintaining false negative and false positive detection rates almost at the level achieved by the Standard HT [12]. Unlike the Randomised HT class of algorithms (see [6] for an overview of HT), Probabilistic HT and the Standard HT share the same one-to-many voting pattern and the representation of the accumulator array.

In the original paper on Probabilistic HT [7], Kiryati et. al. show that it is *often* possible to obtain results identical to Standard HT if only a fraction p of input points is used in the voting process. In the first step of Probabilistic HT a random subset of points is selected and a standard HT is subsequently performed on the subset. Successful experiments with p as low as 2% are presented. The poll size is a parameter critically influencing the Probabilistic HT performance. The authors analyse the problem for the case of a single line immersed in noise. Unfortunately the derived formulas require the knowledge of the number of points belonging to the line *a priori*, which is rare in practice. In [2], Bergen and Schvaytzer show that the Probabilistic HT can be formulated as a Monte Carlo estimation of the HT. The number of votes necessary to achieve a desired error rate is derived using the theory of Monte Carlo evaluation. Nevertheless, the poll size remains independent of the data and is based on *a priori* knowledge[1]. If little is known about the detected objects, a conservative approach (poll size much larger than necessary) must be adopted, diminishing the computational advantage of the Probabilistic HT.

The need to pre-select a poll size can by bypassed by an adaptive scheme [17, 16, 13]. In adaptive Probabilistic HT the termination of voting is based on monitoring of the polling process. The criterion suggested by Yla-Jaaski and Kiryati [17] is based on a measure of stability of the ranks of the highest peaks. Shaked et al. [13] propose a sequential rule. Both methods formulate their goal so as to "obtain the same maximum

[1]Perhaps it is more appropriate to speak about assumptions rather than knowledge

0-7695-0149-4/99 $10.00 © 1999 IEEE

as if the HT had been accumulated and analysed". In our opinion, such formulation is unrealistic since in almost all applications the detection of multiple instances of lines (circles, etc.) is required whereas the given stopping rule depends on the prominence of the most significant feature only. For instance if the input contains any number of lines of equal length, it will not be possible to detect a stable maximum regardless of the percentage of input points used for voting.

In the paper we present a new form of an adaptive Probabilistic HT. Unlike the above-mentioned work we attempt to minimise the amount of computation needed to detect lines (or in general geometric features) by *exploiting the difference in the fraction of votes needed to detect reliably lines (features) with different numbers of supporting points.* It is intuitively obvious (and it directly follows e.g. from Kiryati's analysis in [7]) that for lines with strong support (long lines) only a small fraction of its supporting points have to vote before the corresponding accumulator bin reaches a count that is non-accidental. For shorter lines a much higher proportion of supporting points must vote. For lines with support size close to counts due to noise a full transform must be performed. This is achieved by dynamicly controlling the line acceptance threshold as a function of the number of votes cast. The threshold schedule is derived from theoretical considerations. The line detection algorithm based on the theory is extensively validated experimentally. A comparison with benchmark algorithms including the Standard HT shows it is computationally efficient and has other attractive properties.

The proposed algorithm is sufficiently close to the SHT transform that many of the methods of improving its performance can be easily adapted to work with it; from simply detecting other two-parameter shapes such as circles [11], to use with techniques that change the voting scheme from a one to many scheme such as SHT to one-to-one as found in RHT [14] or window based schemes such as [3]. PPHT does not rely on particular aspects of the input data structure such as connectivity unlike [8] though it borrows the idea of removing labelled structures from the input data. The algorithm is also applicable detection of objects with a higher dimensional parameterisation, as it intrinsically keeps the number of votes in the accumulator space low, it works well with sparse representations of the accumulator space [15].

This method is not so easily adapted to techniques that do repeated accumulation and peak detection such as the Adaptive HT [4] and the Multi-resolution HT [1], since PPHT relies on being able to identify the pixels belonging to a line as they are detected. Algorithms that use two passes to gain information about the image content before doing a full transform such as [10] could be adapted, but with more difficulty.

This paper is organised as follows. In the next section the new algorithm with its underlying theory and properties is presented. In Section 3 its performance is evaluated using two performance criteria: line quality and computational efficiency. In Section 4 we investigate the accumulator size required by the proposed algorithm and its dynamics during the voting process. This paper is drawn to conclusion in Section 5.

2 The Algorithm

To minimise the computation requirements the proposed algorithm which we call *Progressive Probabilistic Hough Transform* (PPHT) proceeds as follows. Repeatedly, a new random point is selected for voting. After casting a vote, we test the following hypothesis: 'could the count be due to random noise?', or more formally 'having sampled m out of N points, does any bin count exceed the threshold l which would be expected from random noise?'. The test requires a single comparison with a threshold per bin update. Of course, the threshold l changes with the number of votes cast. When a line is detected the supporting points retract their votes. Remaining points supporting the line are removed from the set of points that have not yet voted and the process continues with another random selection.

The PPHT algorithm possesses a number of attractive properties. Firstly, a feature is detected as soon as the contents of the accumulator allows a decision. The progressive probabilistic algorithm is an *anytime algorithm.* It can be interrupted and still output useful results, in particular salient features that could be detected in the allowed time. The algorithm does not require a stopping rule. The computation stops when all the points either voted or have been assigned to a feature. This does not mean that a full Hough Transform has been performed. Depending on the data, only a small fraction of points could have voted, the rest being removed as supporting evidence for the detected features. If constraints are given e.g. in the form of minimum line length a stopping rule can be tested before selecting a point for voting.

Without a stopping rule the performance of the PPHT and the Standard HT differs only in the number of false positives. False negatives – missed features with respect to the Standard Hough Transform (SHT) – should not pose a problem, because if the feature is detectable by SHT it will be detected by PPHT at the latest when the voting finished and at that point

the corresponding bin counts of PPHT and SHT are identical. PPHT and SHT performance differs from the point of view of false positives and it is exactly identical only where the assignment of points to lines is ambiguous, ie. where points are in the neighbourhood of more than one line.

The decision threshold is set assumimg that all points are due to noise. It is a worst-case assumption, but if many lines are present in the image the assumption is almost valid, since only a fraction of points belong to any single line.

Let us denote the number of bins for the angle parameter as N_θ and the number of bins for the distance from origin as $N_\rho 1$. We adopt the following model of the voting process. Every randomly selected pixel votes in all N_θ bins. A pixel can belong either to no line (a noise point), to a single line, or lie on an intersection of lines. In the first case all N_θ votes cast add noise to bin counts. We assume that, for every θ, a random bin is incremented, ie. each bin is equally likely to be incremented with probability $1/N_\rho$. If a point lies on a line, one vote is cast into the bin corresponding to this line and the remaining $N_\theta - 1$ votes are assumed to fall into random bins, 1 vote for each angle θ. For points on line intersections we assume $N_\theta - 2$ votes fall in random bins. Since $N_\theta >> 1, N_\theta \approx N_\theta - 1 \approx N_\theta - 2$, we do not (and we cannot not) distinguish between the three cases and assume that always random N_θ bins are incremented.

Clearly, the counts in individual bins are not independent. All counts for bins with a fixed θ must add to the number of edge points that have voted. Moreover, counts in bins with similar θ and ρ are not statistically independent either, due to the cosine voting pattern of a single point. Nevertheless to keep computation tractable we will assume that the count in any single bin is an independent random variable with binomial distribution $B(N,p)$, where N is the number of edge pixels that voted so far and $p = 1/N_\rho$ is the probability of selecting a particular bin with a given θ. The hypothesis that is being tested after every bin incremented is:

> Is the count C in bin (ρ,θ) higher than a value likely to occur if $C(\rho,\theta)$ was a realisation of random variable with binomial distribution $B(N,p)$?

We would like to set the threshold so that

$$P(C(\rho,\theta) > thr) < l$$

The significance level l is a user parameter that shall, if the model is accurate, indicate the number of false positives[2]. For a binomial distribution it is easy but laborious to compute the threshold for a given N, since we have to evaluate the sum

$$\sum_{i=0}^{j} P(C(\rho,\theta) = i)$$

for all j till $1-l$ is reached. Since $Np \approx Np(1-p) > 1$ as in a typical image $N \gg 1$ at the beginning of the voting process, the binomial distribution can be well aproximated by Poisson distribution. This approximation makes computation of (1) much easier (no need to compute $\binom{n}{i}$), but the summation for computing $P(C(\rho,\theta) > thr)$ is still needed. For efficiency reasons we therefore adopt a less acurate approximation for $P(C(\rho,\theta))$ and replace the binomial distribution with Gaussian with mean Np and standard deviation $\sqrt{(Np(1-p))}$. With this simplification, probability $P(C(\rho,\theta)) > thr$ can be obtained using a single look-up in the standard error function.

The approximation will infuence the result mostly for small N, since for $Np > 1$ the approximation with a Gaussian can be justified. If, for small values of N, a better model was needed, the values of the threshold can be precomputed for small N.

It is interesting to note that the analysis is independent of the angular resolution of the accumulator N_θ. For small N_θ the algorithm would confuse instances of lines with similar orientations within the same bin. The recovered models could even compound to give virtual lines. As N_θ increases the resolvability of the lines will increase without affecting the voting strategy. One might argue that if the algorithm was based on incrementing parameter N_ρ and voting into N_θ a certain symmetry should be maintained. This would suggest that the resolution N_θ and N_ρ of the accumulator should be of comparable size.

2.1 Algorithm Outline

1. Check the input pixel list, if it is empty then finish.
2. Update the accumulator with a single pixel randomly selected from the list.
3. Remove pixel from input pixel list.
4. Check if the highest peak in the accumulator that was modified by the new pixel is higher than threshold $thr(N)$. If not then goto 1.
5. Look along a corridor specified by the peak in the accumulator, and find the longest segment either

[2] in case of no post-processing

continuous or exhibiting a gap not exceeding a given threshold.

6. Remove the pixels in the segment from input list.
7. Unvote from the accumulator all the pixels from the line that have previously voted.
8. If the line segment is longer than the minimum length add it into the output list.
9. goto 1.

3 Results on real images.

The aim of the experiments which were performed on real images was to compare the quality of the output produced by PPHT with those of the Standard Hough Transform (SHT) and the Randomised Hough Transform (RHT). The Standard HT has been chosen to provide a baseline representing the achievable quality of line detection whereas the RHT algorithm is representative of voting schemes optimised for computational efficiency. We have adopted public domain implementations of the reference algorithms to ensure that all three algorithms tested would have benefited from a comparable degree of tuning and refinement. The algorithms were run with accumulator resolution of 1 pixel for ρ and 0.01 radians for θ.

In [9], we reported the PPHT performance on synthetic images containing variable number of lines (5-100) of variable length and density. Here the advocated and the reference algorithms are compared on five images representing different complexity of line structure and image content. The house image is used as a standard benchmark and the results of other algorithms applied to it can be found in the literature. The arch and machine parts are images of medium complexity, containing not only straight line segments of different length but also curves. The most testing structures are presented by sets of parallel lines which are separated by a gap of a few pixels only. The office image is an example of highly complex image where the background structure constitutes a dense clutter.

Two criteria were used for the comparison: i) quality of the extracted lines and ii) computational efficiency. Whilst the quality of the detected structures was evaluated subjectively, computational efficiency was measured objectively, using different measures. PPHT and SHT were compared using the number of votes cast by each method. This is an unbiased measure for these two algorithms as they perform the same type of operations. Unfortunately, we could not adopt the same approach to measure the computational complexity of the RHT method as the nature of the operations performed by the algorithm is completely different. For this reasons RHT's computational complexity was measured in terms of the CPU time and this was related to the same of the PPHT algorithm.

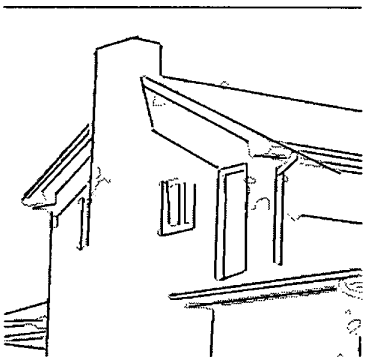


Figure 1: Results of SHT using gradient info for accumulation and line termination.

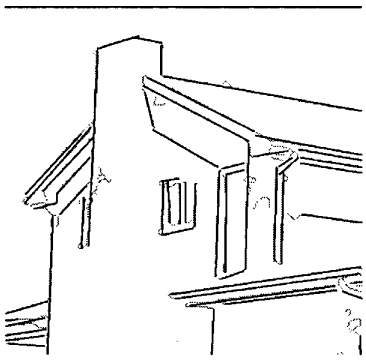
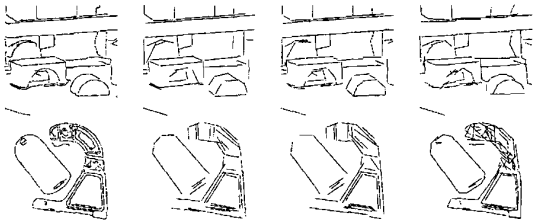


Figure 2: Results of PPHT using gradient info for accumulation and line termination.

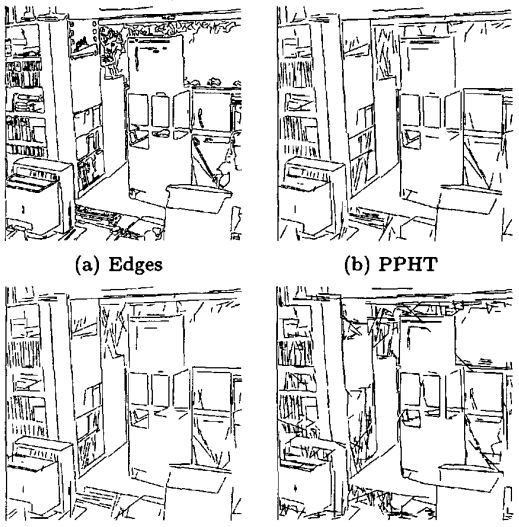
The results on the house image are shown in Figures 1, 2 and 3. Comparing the line output produced by the SHT algorithm shown in Figure 1 with the PPHT output in Figure 3 we note that they are comparable in quality. If anything, the PPHT algorithm is slightly superior as it does not tend to favour long lines over short ones. This tendency of SHT leads to the detection of long virtual lines which are supported by unrelated structures. PPHT has also coped better with the parallel line structures in the bottom right quadrant of the image. It is interesting to note the points used in voting by the PPHT algorithm shown in Figure 3. The dark points correspond to edge pixels which failed to be interpreted on termination of the algorithm when all the pixels were either used in voting or removed from the edge list by virtue of being con-

Figure 3: Points used in voting PPHT.



(a) Edges (b) PPHT (c) SHT (d) RHT

Figure 4: Arch (Top) and Machine Part



(a) Edges (b) PPHT

(c) SHT (d) RHT

Figure 5: Office

Image	SHT Votes	PPHT Votes	Fraction	PPHT Time	RHT Time
Arch	5345	1186	0.22	0.8	1.6
Part	5347	1439	0.27	1.8	6.2
Office	36326	8718	0.24	5.0	28.8

Table 1: Computation time in terms of votes for a variety of real images.

sistent with the lines already detected. The light grey points are the voting pixels providing evidence for the detected structure. It is apparent, that the number of the voting points is considerably lower than the total number of edge pixels used by SHT. Moreover, the distribution of voting pixels is not uniform. The density of voting pixels for shorter lines is much higher than for longer lines. This shows clearly that the subsampling approach of the Probabilistic HT will use an unnecessary large number of voting points for detecting long lines whereas short lines may end up undetected.

The results obtained on the arch, machine parts and office images clearly demonstrate that in more complex data, the Randomised HT tends to detect unacceptable large amount of spurious structure. PPHT and SHT results are of similar quality. In some case the tendency of SHT to form long lines leads to better results as in Top of Figure 4 where the two quasi horizontal lines at the top of the image are detected as single structures whereas PPHT breaks the lines up into several models. This tendency works against SHT in Bottom of Figure 4 when modelling the base of the larger machine part. SHT detects one virtual single line for the base at an angle which allows edge pixels from the two parallel structures to be subsumed by the same model.

The computational efficiency of the advocated method is illustrated in Table 1. The average savings achieved by PPHT is about 75 % of the time needed to perform SHT. To a large extent this appears to be independent of image complexity. Table 1 also indicates that though RHT is relatively efficient for simple data, for more complex problems not only does the relative execution time increase but there is also a loss in the quality of its performance. This is particularly visible in Figure 5 with many of the small features being completely misinterpreted by RHT. For this image SHT takes less time to complete the computation than RHT.

4 Accumulator size statistics

The next objective of our experimentation was to investigate the required size of the accumulator and its dynamics during the line detection process. In

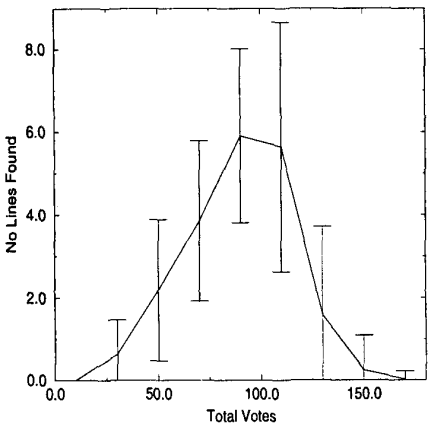


Figure 6: Lines found as a function of votes.

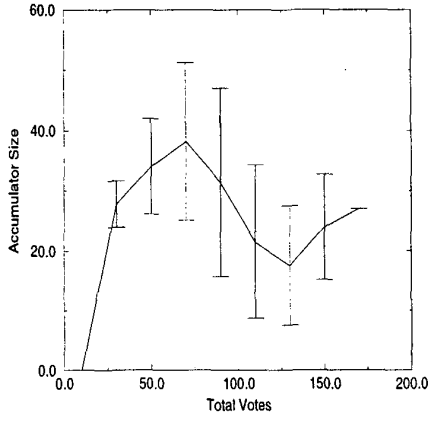


Figure 7: PPHT Accumulator sizes.

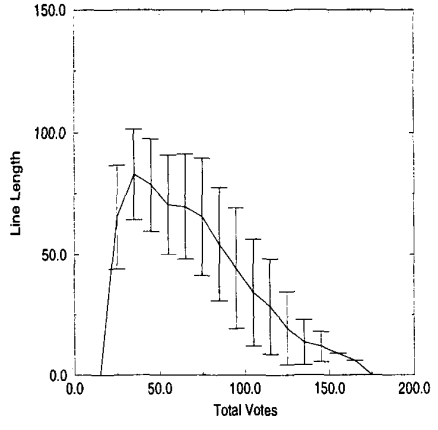


Figure 8: Votes required to recover lines of different lengths.

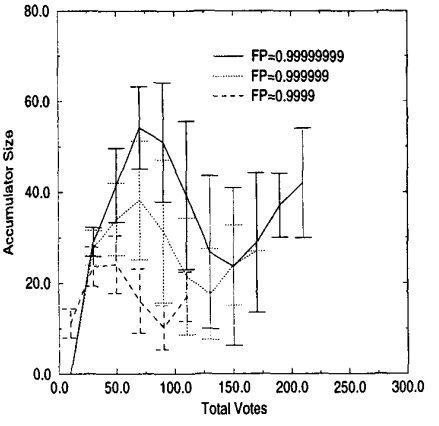


Figure 9: Effect of varying FP on accumulator size.

particular, we were interested in exploring the relationship between the total number of votes, the number of lines found, accumulator size and line length. The experiment was conducted using synthetic images each composed of 20 lines of random orientation and length between 1 and 100 pixels. The statistics of the above quantities were collected over a sample set of 100 such images. Figure 6 shows the relationship between the number of lines found in this ensemble for a given number of votes per image. As each image contains on average 1000 pixels, it is apparent that PPHT which detects all the structure with less than 170 votes is about six times faster than SHT. In 10% of the time taken by SHT, we detect more than 60% structure. Most importantly, from Figure 8 it follows that this 60% relates to the most salient lines (longest lines). Thus after 10% of time most of the pixel structure will be interpreted and most important lines segments recovered. The tendency for PPHT to select long lines first suggests that by terminating the algorithm any time will always guarantee the recovery of the most salient structure. In fact Figure 8 shows clearly that short lines start being recovered only after a substantial number of votes have been collected.

The variation of the accumulator size with the total number of votes is shown in Figure 7. The curve initially increases steeply and linearly with the number of votes. For SHT this linear relationship would continue until the completion of the accumulation process. In contrast PPHT will at some stage start detecting line models and identifying all pixels associated with these models. The pixels assigned to the recovered lines which have voted will undergo the process of unvoting. In other words their effect on accumulator content

will be cancelled and as a result the accumulator size will be reduced. Those pixels that have not yet voted will be removed from the pool of candidate pixels and this is instrumental in accomplishing the claimed computational efficiency gains. Thus the accumulator size will continue diminishing until most of the recoverable structure is pulled out. As all the lines of length above a specified threshold are detected, the uninterpreted clutter will remain in the accumulator and its size will start again growing. The final level will be a function of the saturation value of the line hypothesis test threshold. Thus for PPHT with higher confidence levels the threshold will be higher (see Figure 9) and consequently the terminal accumulator size larger. It is interesting to note that the maximum accumulator size for this type of imagery is less than 5% of that required by the standard HT. This augurs well for the applicability of the proposed algorithm for detecting other parametric curves.

5 Conclusions

In this paper we presented *Progressive Probabilistic Hough Transform*(PPHT) algorithm. We showed that unlike the Probabilistic HT the proposed algorithm minimises the amount of computation needed to detect lines by exploiting the difference in the fraction of votes needed to detect reliably lines with different support.

The dependence of the fraction of points used for voting as a function of the inherent complexity of data was studied. We demonstrated on input images containing N lines that the number of votes (speed) of PPHT is almost independent of line lengths (input points).

Though not extensively explored in this paper the proposed algorithm is sufficiently close to the SHT transform that many of the methods of improving its performance can be easily adapted to work with it.

The algorithm is ideally suited for real-time applications with a fixed amount of available processing time, since voting and line detection is interleaved. The most salient features are likely to be detected first. Experiments show PPHT has, in many circumstances, advantages over the Standard HT.

References

[1] M Atiquzzaman. Multiresolution Hough Transform - an efficient method of detecting patterns in images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(11):1090–1095, 1992.

[2] J R Bergen and H Shvaytser. A probabilistic algorithm for computing Hough Transforms. *Journal Of Algorithms*, 12(4):639–656, 1991.

[3] A Califano and R M Bolle. The Multiple Window Parameter Transform. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 14(12):1157–1170, 1992.

[4] J Illingworth and J Kittler. The Adaptive Hough Transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):690–698, 1987.

[5] J Illingworth and J Kittler. A survey of the Hough Transform. *Computer Vision, Graphics and Image Processing*, 44:87–116, 1988.

[6] H Kalviainen, P Hirvonen, L Xu, and E Oja. Probabilistic and nonprobabilistic Hough Transforms - overview and comparisons. *Image And Vision Computing*, 13(4):239–252, 1995.

[7] N Kiryati, Y Eldar, and A M Bruckstein. A probabilistic Hough Transform. *Pattern Recognition*, 24(4):303–316, 1991.

[8] V F Leavers. 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):381–398, 1992.

[9] J Matas, Ch Galambos, and J Kittler. Progressive probabilistic hough transform. In M. S. Nixon, editor, *Proc British Machine Vision Conference BMVC98*, pages 256–265, 1998.

[10] P L Palmer, J Kittler, and M Petrou. Using focus of attention with the Hough Transform for acccurate line parameter estimation. *Pattern Recognition*, 27:1127–1133, 9 1994.

[11] Soo-Chang Pei and Ji-Hwei Horng. Circular arc detection based on Hough Transform. *Pattern Recognition Letters*, 16:615–625, 1995.

[12] J Princen, J Illingworth, and J Kittler. Hypothesis-testing - a framework for analyzing and optimizing Hough transform performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(4):329–341, 1994.

[13] D Shaked, O Yaron, and N Kiryati. Deriving stopping rules for the probabilistic Hough Transform by sequential-analysis. *Computer Vision And Image Understanding*, 63(3):512–526, 1996.

[14] L Xu and E Oja. Randomized Hough Transform (RHT) - basic mechanisms, algorithms, and computational complexities. *Cvgip-image Understanding*, 57(2):131–154, 1993.

[15] L Xu, E Oja, and P Kultanen. A new curve detection method: Randomized Hough Transform (RHT). *Pattern Recognition Letters*, 11:331–338, 1990.

[16] A Yla-Jaaski and N Kiryati. Automatic termination rules for probabilistic hough algorithms. In *8th Scandinavian Conference on Image Analysis*, pages 121–128, 1993.

[17] A Yla-jaaski and N Kiryati. Adaptive termination of voting in the probabilistic circular Hough transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):911–915, 1994.