Computer Vision and Image Understanding
Vol. 73, No. 3, March, pp. 329–345, 1999
Article ID cviu.1998.0728, available online at http://www.idealibrary.com on IDEAL®

Constrained Hough Transforms for Curve Detection

Clark F. Olson*

Jet Propulsion Laboratory, California Institute of Technology, Mail Stop 125-209, 4800 Oak Grove Drive, Pasadena, California 91109

Received January 7, 1997; accepted July 31, 1998

This paper describes techniques to perform fast and accurate curve detection using constrained Hough transforms, in which localization error can be propagated efficiently into the parameter space. We first review a formal definition of Hough transform and modify it to allow the formal treatment localization error. We then analyze current Hough transform techniques with respect to this definition. It is shown that the Hough transform can be subdivided into many small subproblems without a decrease in performance, where each subproblem is constrained to consider only those curves that pass through some subset of the edge pixels up to the localization error. This property allows us to accurately and efficiently propagate localization error into the parameter space such that curves are detected robustly without finding false positives. The use of randomization techniques yields an algorithm with a worst-case complexity of $O(n)$, where n is the number of edge pixels in the image, if we are only required to find curves that are significant with respect to the complexity of the image. Experiments are discussed that indicate that this method is superior to previous techniques for performing curve detection and results are given showing the detection of lines and circles in real images. © 1999 Academic Press

1. INTRODUCTION

The Hough transform is a method to detect parameterized curves in images by mapping image edge pixels into manifolds in the parameter space [9, 13]. The parameters that are consistent with many of the manifolds correspond to curves in the image and thus methods that find peaks in the parameter space can be used to detect the image curves. For example, we may parameterize lines by their slope and intercept ($y = mx + b$). Each edge pixel, (x, y), in the image is mapped into the line $b = -xm + y$ in the parameter space ($m \times b$), corresponding to all of the lines that pass through (x, y).

The peaks in the parameter space are typically found using a multidimensional histogramming procedure, where each manifold votes for the cells of the histogram that it passes through. The cells of the histogram that receive many votes are taken to indicate curves in the image. Unfortunately, this technique does not take into account the localization and discretization errors that are present in the image edge pixels. While techniques have been proposed to deal with these errors, they have not been fully satisfactory in both robustly locating noisy curves and eliminating false positives from consideration. In addition, many of these Hough transform techniques have been very expensive, in terms of both computation and memory requirements.

In this paper, we consider techniques to improve the efficiency and accuracy of curve detection using the Hough transform. The key insight that enables the design of an improved algorithm is that the curve detection problem can be subdivided into many small subproblems that can be examined independently without a reduction in the curve detection performance. This decomposition of the problem allows the localization error in the image features to be efficiently propagated into the parameter space, and the additional use of randomization yields a robust algorithm for curve detection that requires linear time in the number of image edge pixels.

We first discuss previous work on the Hough transform for curve detection and review a formal definition of the Hough transform given by Princen *et al.* [32]. This definition assumes that a histogramming method is used to perform peak detection and is not directly applicable to methods that propagate the localization error accurately. We give a modification to this definition that allows the formal treatment of the effects of localization error in the Hough transform.

We then consider a technique where sets of pixels, rather than single pixels, are mapped into the parameter space. In this technique, the set of curves that are consistent with small subsets of image edge pixels are determined, and the parameter space peaks are accumulated accordingly. Variants of this method can be found in many papers, for example [2, 5, 9, 20, 23, 48]. Our analysis shows that this technique, by itself, does not change the accuracy with which curves that surpass some arbitrary threshold are found, assuming that a perfect peak finding method is used, since peaks due to random accumulation form in the parameter space with the same frequency as when individual pixels are used. Furthermore, unless additional heuristics are used, the running time of the method is exponential in the number of curve parameters (as is the standard Hough transform[1]).

* http://robotics.jpl.nasa.gov/people/olson/homepage.html.

[1] The base of the exponential is the number of bins per dimension in the accumulator method for the standard Hough transform method, while it is the number of image edge pixels for the techniques where pixel sets are mapped into the parameter space.

1077-3142/99 $30.00

We next show that subproblems of the curve detection problem can be considered that are constrained to examine only those sets of image edge pixels that share some distinguished set of pixels of some fixed size j. This method is adapted from recent work on object recognition [7, 28] and also has been used in a limited form for curve detection [20, 24]. Each such subproblem corresponds to examining only the curves that pass through the distinguished set of pixels (up to the localization error). If we examine each possible subproblem, we suffer no loss in curve detection performance.

Randomization can be used to limit the number of subproblems that need to be examined while maintaining a low probability of failure. Unlike previous uses of randomization in the Hough transform, where the image edge pixels are randomly sampled in some manner during the accumulation process, this does not reduce the detection performance for any particular subproblem. We still examine all of the edge pixels in each of the subproblems. The only reduction in performance is that there exists a small probability, which can be set arbitrarily low, that no subproblem will be examined that uses a distinguished set of pixels belonging to a curve present in the image. This distinction is important. Many Hough transform methods that have incorporated randomization examine a subset of the possible features or a subset of the possible feature combinations when accumulating scores in the parameter space. This not only causes curves in the image to be missed, but also causes false positives to be detected, since only an approximation of the Hough transform is computed. Our technique does not suffer from these problems.

The examination of these subproblems not only allows an effective use of randomization, but it also allows the efficient propagation of the localization errors into the parameter space and makes the parallelization of these techniques simple. We consider the propagation of localization error in detail for the cases of straight lines and circles and we analyze the expected accumulation of votes due to random combinations of pixels. These techniques are compared with previous Hough transform methods with respect to detection performance, both theoretically and empirically, and the new techniques are shown to be a significant improvement over previous methods.

The computational complexity of the algorithm is $O(n)$ (or $O(n \log n)$ depending on the method used to detect peaks) when randomization is used to limit the number of subproblems that must be examined. Results are given demonstrating the detection of lines and circles in real images. We then discuss the application of these techniques to higher-order curves such as ellipses. Finally, the contributions of this work are summarized. An appendix is included that describes parameterizations for some interesting curves and gives methods to solve for the parameters of each from the minimal amount of information.

2. RELATED WORK

The Hough transform was introduced in a U.S. patent in 1962 [13] and was initially used to locate particle tracks in bubble-chamber imagery. It was brought to the attention of the computer vision community by Rosenfeld [35]. A subsequent paper by Duda and Hart [9] refined by technique by suggesting the use of the normal parameterization for lines (see Appendix) and the alternative of mapping pairs of pixels into the parameter space rather than individual pixels.

A complete review of Hough transform techniques for curve detection requires an entire paper. See, for example, [17] and [21] for comprehensive reviews of research on the Hough transform. Here we focus on related work designed to improve either the efficiency or the error propagation of the Hough transform method.

2.1. Error in the Hough Transform

Considerable research has been performed with the goal of analyzing and improving the detection performance of the Hough transform in relation to localization error in the image and discretization error in both the image and the parameter space.

Much early analysis of the Hough transform was performed by Shapiro [36, 37, 39]. This work examined, in particular, approximations to the variance of the location of the points in the parameter space in terms of the variance of the locations of pixels in the image space. Shapiro examined both cases where single oriented image pixels were mapped into the parameter space and cases where pairs of unoriented image pixels were mapped into the parameter space.

In addition, Shapiro and Iannino [40] have given a geometric construction of the region in the parameter space that a point in the image maps to for line detection under a bounded error assumption. This information was applied to the determination of the appropriate size for cells in the accumulator method of peak detection. Shapiro [38] further considered a Hough transform variation where the edge points are mapped into all of the curves in the parameter space that satisfy the error model for the edge point. We argue that this is the correct direction to take in propagating discretization and localization error in the Hough transform, since the optimal use of this information will detect those curves that pass within the bounded error of some specified number of image edge pixels. However, Shapiro's application of these ideas was to use the accumulator method, with the modification that all of the accumulator cells consistent with the error model for a particular point receive votes. This process has extreme computational requirements. Shapiro suggested the use of large grid cells to reduce this problem, despite the inherent loss of resolution and curve discrimination performance.

Van Veen and Groen [46] examined the effects of the discretization errors and the width of the line segments on the performance of the standard Hough transform. They concluded that a method by which the gradient information at each edge pixel is used to reduce the influence function of each pixel [26] yields considerable improvement. Brown [4] argued that the Hough transform is inherently deficient due to the contributions of various pixels on a line to off-peak locations in the parameter space. He suggested that the use of negative votes in the parameter space

can be used to offset this effect and sharpen the peaks. Kiryati and Bruckstein [18] analyzed the Hough transform in terms of sampling of a nonbandlimited signal. They claim that aliasing accounts for many of the problems with the Hough transform and present a method that avoids these problems through the use of an influence function that is essentially bandlimited. However, no implementation of these techniques is described.

Further results can be found in [8, 11, 25, 14, 34, 41, 44]. Unfortunately, all of these techniques are concerned with various aspects of the accumulator method for finding peaks in the parameter space, where the parameter space is discretized and a counter is kept for each of the cells. This method for locating peaks has the inherent problem that it is desirable for the cells to be both large enough that the pixels from a particular line (or a particular curve, in general) accumulate in a single cell and small enough that the random accumulation of votes does not result in false positives. Analysis by Grimson and Huttenlocher [12] has indicated that, when this method is applied to even moderately complex problems, a significant rate of false positives occurs.

There has recently been some work which does not depend upon the accumulator method in which localization error has been treated in an interesting manner. Stephens [43] formulated a variant of the Hough transform in terms of maximum likelihood estimation. A probability density function for the features is used that has a uniform component modeling the pixels that are not on the curve and a component that falls off as a Gaussian with the distance from the curve to model the pixels that are on the curve. While this method yields correct propagation of localization error in terms of a Gaussian error distribution, it is computationally expensive.

Princen *et al.* [33] examined the Hough transform in the framework of hypothesis testing. Each accumulator bin is viewed as a hypothesis and is assigned a test statistic that is the sum of the scores for each of the data features with respect to the hypothesis. The data features yield scores that are a function of the distance of the feature from the hypothesis line according to a smooth kernel function. Princen *et al.* considered the optimal kernel that should be used to assign scores to the data features in this framework. Palmer *et al.* [30] extended this work to consider a two-dimensional kernel that is a function of both the distance of the feature from the line and the difference in orientations. This reduces the number of lines for which each data feature yields a significant score. Palmer *et al.* also considered the optimal two-dimensional kernel for performing line detection. These techniques allow localization error to be propagated into the parameter space and are complementary to the decomposition techniques that we describe below. The hypothesis testing framework has also been applied to the problems of edge detection, circular arc detection, and planar surface segmentation [15].

Soffer and Kiryati [42] also examine continuous kernel functions. By considering the Hough transform as an optimization problem, they are able to examine under what conditions the Hough transform can be guaranteed to converge to the correct solution(s). While they have not been able to achieve useful strict convergence in the presence of localization error, they are able to achieve a wide-sense convergence in this case based on edge and noise models. This analysis provides useful guidelines for selecting the kernel width and the bin size or sampling interval in the parameter space.

Breuel [3] described a line detection technique related to the Hough transform that searches hierarchical subdivisions of the parameter space using a bounded error model and thus avoids some of the problems of the accumulator method. In this technique, the parameter space is divided into cells that are tested to determine whether they can contain a line that passes within the bounded localization error of a specified number of pixels. If the cell cannot be ruled out, the cell is divided and the procedure is repeated recursively. This continues until the cells become sufficiently small, at which point they are considered to be lines satisfying the output criterion.

2.2. Computational Techniques

Another fertile area for research has been methods to improve the computational expense required by the Hough transform. Two methods that have been widely used are mapping sets of image pixels into the parameter space and randomization.

An early paper by Murakami *et al.* [24] described algorithms for performing straight line detection using a one-dimensional accumulator. One of the algorithms maps pairs of feature points into the one-dimensional accumulator by taking the angle of the line between the points. The algorithm is structured in such a way that only the pairs containing one of the feature points are examined in each iteration. This can be viewed as a precursor to the decomposition techniques that we describe. Unfortunately, this algorithm had $O(n^2)$ complexity, where n is the number of image features and did not consider the effects of localization error.

Xu *et al.* [48] described the randomized Hough transform. For the detection of curves with N parameters, they map sets of N image pixels into single points in the parameter space and accumulate votes in a discretized version of parameter space. The pixel sets that are mapped into the parameter space are chosen randomly and the votes are accumulated until a sufficient peak is found or some threshold number of sets have been examined. Xu and Oja [47] give a robust stopping criterion for this procedure and they approximate the time required by the algorithm by modeling it as a generalized Bernoulli process.

Liang [23] discusses a Hough transform technique where sets of image pixels are mapped to single points in a discretized parameter space by fitting curves to the pixels in small windows of the image. This method allows a fast implementation and a low storage requirement, but detection performance will degrade in the presence of image noise, due to poor local fits, and in cluttered images, due to distractors present in the image windows.

Bergen and Shvaytser [2] gave a theoretical analysis of the use of randomization techniques to speed up the Hough transform, although no implementation is described. They consider both mapping individual pixels and mapping sets of pixels into

the parameter space. Their method achieves a computational complexity independent of the number of edge pixels in the image, but with two caveats. First, only curves that represent some predetermined fraction of the total number of edge pixels are found. Second, the method is allowed to be in error by a fractional parameter all of the time and by greater than this fractional parameter with some small frequency. The practicality of this method is questionable, since the constant number of random samples that must be examined is often very large. In fact, this number may often be larger than the number of different samples that are possible.

Kiryati *et al.* [19] used randomization to improve the running time of the standard Hough transform. They simply subsampled the edge pixels in the image and proceeded with the standard algorithm. Their analysis implies that the computational requirements of the Hough transform can be improved, while the performance is degraded little.

Califano and Bolle [5] used a multiple window parameter transform to exploit long distance information in the extraction of parameterized objects from image data. Lateral inhibition is used in a connectionist-like framework to improve the accuracy. In addition, a radius of coherence is defined for each pixel to reduce the computation time required by the method.

Leavers [20] described a technique called the dynamic generalized Hough transform. She used the technique of mapping N image pixels into a single point in the parameter space and, furthermore, in each iteration selected a single image pixel to constrain the transform, which must be present in each of the sets of pixels that are mapped into the parameter space. We propose a similar technique in this paper, in which the problem is divided into subproblems where the pixel sets that are mapped into the parameter space must share $N-1$ image pixels and demonstrate that this is superior to using a single pixel to constrain the transform. Leavers used a storage efficient voting mechanism in a discretized parameter space, where the votes are projected onto each of the parameter space axes and several one-dimensional accumulators are kept. While this method of accumulating votes reduces that amount of memory that is required, it may exacerbate the problem of false alarms if the votes in the parameter space are not sparse.

Finally, we note that an additional technique that has proven useful in the efficient implementation of Hough transform techniques is a multiresolution or coarse-to-fine search of the parameter space to find peaks [1, 3, 16, 22, 29, 31]. For example, Li *et al.* [22] recursively divided the parameter space in hypercubes in a coarse-to-fine hierarchy. At each level of the hierarchy, only those hypercubes that receive enough votes to surpass some threshold are passed on to the next level for examination.

3. THE HOUGH TRANSFORM

Except where noted, we will consider each edge pixel to be located at the point at the center of the image pixel in which it lies, with an insignificant extent. The discretization error inherent in this formalism is treated in combination with localization error below. We note, though, that this formalism is not necessary, and the techniques that are described here apply equally well to arbitrary sets of points.

Princen *et al.* [32] gave a formal definition of the Hough transform that, with some modification, will prove useful in our analysis. Let $X=(x, y)$ be a point in image space, $\Omega=(\omega_1, \ldots, \omega_N)$ be a point in the parameter space, and $f(X, \Omega)=0$ be the function that parameterizes the set of curves. The set of edge pixels in the image, $\mathcal{E}=\{X_1, \ldots, X_n\}$, is represented by the sum of the delta functions at the pixel's locations:

$$I(X)=\sum_{j=1}^{n}\delta(X-X_j). \tag{1}$$

Princen *et al.* used C_Ω to denote a cell in the parameter space centered at Ω. Examining such cells allows the consideration of a discretized parameter space. They define

$$p(X, \Omega)=\begin{cases}1 & \text{if } \{\Lambda : f(X, \Lambda)=0\}\cap C_\Omega \neq \emptyset \\ 0 & \text{otherwise.}\end{cases} \tag{2}$$

So, $p(X, \Omega)$ is 1 if any curve in the parameter space cell, C_Ω, passes through the point, X, in image space. The Hough transform can be written

$$H(\Omega)=\int p(X, \Omega)I(X)\,dX \tag{3}$$

or

$$H(\Omega)=\sum_{j=1}^{n}p(X_j, \Omega). \tag{4}$$

$H(\Omega)$ is thus the number of image pixels that any curve in C_Ω passes through. This definition is correct for the accumulator method that is typically used to implement the Hough transform. Such methods discretize the parameter space into cells and maintain a counter for each cell. These counters record the number of edge pixels that map to a manifold in the parameter space that intersects the cell. However, the performance of such methods is less than optimal for two reasons. First, this assumes that there is no localization error in the edge pixels. When localization error is present, the cell corresponding to the position of the curve may not receive a vote for various pixels that belong to the curve. Second, a single bin may receive votes from multiple edge pixels that cannot lie on the same line, even when localization error is considered. It is implicitly assumed that the bins are large enough to catch the votes for the curves of interest, yet small enough not to catch a large number of votes from false positive curves.

In this work, we discuss a method that solves the first problem and greatly reduces the second through a decomposition of the

problem into constrained subproblems and propagation of the localization error into the parameter space in each of the subproblems. We thus prefer a definition that does not incorporate a discretization of the parameter space, but that does take into account the propagation of the localization error into the parameter space. Let us assume that the true location of each edge pixel lies within some bounded region, N_X, of the determined location X. We can redefine $p(X, \Omega)$ as follows:

$$p(X, \Omega) = \begin{cases} 1 & \text{if } \{Y : f(Y, \Omega) = 0\} \cap N_X \neq \emptyset \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

Now, $p(X, \Omega)$ is 1 if the curve represented by Ω passes through N_X. With this definition we can still use Eqs. (3) and (4) to describe the Hough transform and we achieve our goal of accounting for the propagation of error, without incorporating a discretized parameter space. To be fully general, we could further modify this definition such that $p(X, \Omega)$ was a continuously valued function as in [30, 33, 43]. However, this requires us to assume a model for the image noise and localization error, at least implicitly, rather than just an upper bound on the allowable localization error.

In this definition, the Hough transform is continuous in the parameter space. This gives us freedom in considering methods by which the peaks in the parameter space are found. We are not bound to the accumulator method. We are also not constrained to use the same localization error boundary, N_X, for each point. If we have information as to the relative quality of the localization of the edge points, it can be used here.

Note that it is possible to propagate the localization error into the parameter space in the accumulator method for implementing the Hough transform by determining, for each image edge pixel, precisely which cells in the quantized parameter space contain the parameters of a curve that passes within the bounded error region of the pixel and incrementing the counters accordingly. This method has been proposed by Shapiro [38]. Unfortunately, this process is computationally expensive and addresses only the first of the problems discussed above.

4. MAPPING PIXEL SETS INTO THE PARAMETER SPACE

Let us now consider the technique of mapping sets of pixels into the parameter space. Rather than considering each pixel separately, this method considers the sets of pixels with some cardinality k. For each such set, the curves that pass through every pixel in the set (or through the bounded error region of every pixel in the set) are determined and the parameter space scores accumulate accordingly. The primary benefit of using this technique is that each mapping is into a smaller subset of the parameter space. If $f(X, \Omega)$ is an N parameter function, then, in the errorless case, N nondegenerate edge pixels map to a single point in the parameter space[2]. In this case, the accumulator method needs to increment only a single bin in the parameter space for each set, rather than the bins covering an $N-1$ dimensional manifold for each edge pixel. Such a technique has been used in some form by several researchers [2, 5, 9, 20, 23, 48]. Of course, we need not use sets with cardinality N, we could use any size $k > 0$. If $k \leq N$, then each nondegenerate pixel set maps into an $N-k$ dimensional manifold in the parameter space. The disadvantage to this technique of mapping sets of pixels into the parameter space is that there are $\binom{n}{k}$ sets of image pixels with cardinality k to be considered, which grows very quickly as k increases.

An examination of how this technique is related to the standard Hough transform is informative. Let us denote the transform where sets of k pixels are mapped into the parameter space $H^k(\Omega)$. (The standard Hough transform is thus $H^1(\Omega)$, but we will continue to denote it simply $H(\Omega)$.) An image curve (i.e., a point in the parameter space) now receives a vote only if it passes within the error boundary of each pixel in the set, so we have

$$H^k(\Omega) = \sum_{\{X_{g_1}, \ldots, X_{g_k}\} \in \binom{\mathcal{E}}{k}} p(X_{g_1}, \Omega) \cdot \ldots \cdot p(X_{g_k}, \Omega), \quad (6)$$

where $\binom{\mathcal{E}}{k}$ denotes the set of all k-subsets of the edge pixels, $\mathcal{E}$.

Consider this function at an arbitrary point in the parameter space. For some set of pixels, $\{X_{g_1}, \ldots, X_{g_k}\}$, the product $p(X_{g_1}, \Omega) \cdot \ldots \cdot p(X_{g_k}, \Omega)$ is 1 if and only if each of the $p(X, \Omega)$ terms is 1 and otherwise it is 0. If there are x pixels such that $p(X, \Omega)$ is 1 (these are the pixels that lie on Ω up to the localization error), then there are $\binom{x}{k}$ sets with cardinality k that contribute 1 to the sum. $H^k(\Omega)$ is thus $\binom{x}{k}$. Since the standard Hough transform yields $H(\Omega) = x$ in this case, we can express $H^k(\Omega)$ simply in terms of $H(\Omega)$:

$$H^k(\Omega) = \binom{H(\Omega)}{k}. \quad (7)$$

This result indicates that the method of mapping point sets into the parameter space has the same accuracy as the standard Hough transform. If the standard Hough transform uses threshold $t \geq k$ to find peaks and the method of mapping pixel sets into the parameter space uses a threshold of $\binom{t}{k}$, then the above analysis implies that they will find exactly the same set of peaks, assuming that a perfect peak finding method is used. No correct peaks are missed as a result of using this method and no false positives are eliminated.

In addition, curve detection still requires time that is exponential in N. Let α be the number of bins in each dimension of

[2] This is true for the curves considered in this paper (lines, circles, and ellipses), but it is not necessarily true for arbitrary classes of curves. In general, if we assume nondegeneracy, then a set of N errorless edge pixels maps to a finite set of points in the parameter space.

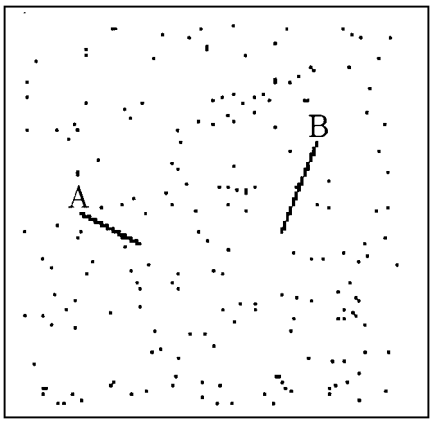

FIG. 1. Line segments A and B amid random noise.

the parameter space. The standard Hough transform increments $O(\alpha^{N-1})$ bins for each image edge pixel, for a total requirement of $O(n\alpha^{N-1})$. The new method increments $O(\alpha^{N-k})$ bins for each of the $O(n^k)$ sets, for a total of $O(\alpha^{N-k}n^k)$.[3] Furthermore, the accurate propagation of localization error in the parameter space appears to be no easier with this method. This technique thus has not gained us much, yet. In fact, it is less efficient than the standard Hough transform method when there are many edge pixels present in the image.

Let us examine what these transforms look like in the parameter space. Figure 1 shows an example image with two short line segments amid some random noise that we use to illustrate the transforms. Figure 2 shows transforms of this image using the ρ-θ parameterization for lines. The peaks corresponding to the line segments in Fig. 1 are labeled with letters. The first plot is the standard Hough transform of the image and the second plot is the transform where pairs of pixels are mapped into the parameter space. While the peaks are much higher in the second case, our analysis indicates that if there was a false positive peak in the standard Hough transform, it would also be present for this case. These plots were made without propagating the localization error into the parameter space. However, to somewhat ameliorate the effects of error, the counters were incremented for each cell in a 3×3 window around the cells that were intersected by the appropriate manifold in the parameter space.

5. DECOMPOSITION INTO SUBPROBLEMS

Let us now consider a new technique where only those pixel sets that share some distinguished set of j edge pixels, $\mathcal{D} = \{X_{d_1}, \ldots, X_{d_j}\}$, are mapped into the parameter space. This is similar to previous work by Murakami *et al.* [24] and Leavers [20], who used a single edge pixel to constrain the curves in the parameter space. However, we do not restrict ourselves to using a single pixel to constrain the curves. This technique has also been used in the context of object recognition to subdivide difficult problems into small subproblems [7, 28].

In these constrained transforms, we vary $k - j$ edge pixels, $\mathcal{G} = \{X_{g_1}, \ldots, X_{g_{k-j}}\}$, in the edge pixel sets. The pixel sets we map into the parameter space are thus $\mathcal{D} \cup \mathcal{G}$. This yields a new transform as follows:

$$H^{\mathcal{D},k}(\Omega) = \sum_{\mathcal{G} \in \binom{\mathcal{E}-\mathcal{D}}{k-j}} \prod_{i=1}^{j} p(X_{d_i}, \Omega) \prod_{i=1}^{k-j} p(X_{g_i}, \Omega). \quad (8)$$

Consider this function at an arbitrary point in the parameter space. Since we do not vary the distinguished pixels, $\{X_{d_1}, \ldots, X_{d_j}\}$, the curve must pass through the localization error boundary of each of these pixels to yield a nonzero response. If x pixels lie on a curve up to the localization error, and we use a distinguished set of j of these pixels, then $x - j$ of these pixels remain in $\mathcal{E} - \mathcal{D}$. We thus have

$$H^{\mathcal{D},k}(\Omega) = \begin{cases} \binom{H(\Omega) - j}{k - j} & \text{if } \prod_{i=1}^{j} p(X_{d_i}, \Omega) = 1 \\ 0 & \text{otherwise.} \end{cases} \quad (9)$$

A threshold of $\binom{t-j}{k-j}$ is appropriate for $j < k$, if we wish to find curves comprising at least t pixels. Perfect peak finding methods will find any curve that passes through the localization error boundaries of each of the distinguished pixels, if it would have been found by the standard Hough transform method with threshold t.

We can formulate an algorithm to recognize arbitrary curves by considering several subproblems, each of which examines a particular distinguished set, as above. A deterministic algorithm using these ideas would consider each possible distinguished set. This would guarantee that we examine a correct distinguished set for each curve. If we are willing to allow a small probability of failure, we can use randomization to reduce the number of distinguished sets that must be examined. Note that even when randomization is used in this manner, each of the subproblems retains full accuracy in terms of discrimination between true curves and false positives, unlike other uses of randomization in the Hough transform. The only drawback is an arbitrarily small possibility of missing a curve due to failing to examine a distinguished set of pixels that belongs to the curve. An analysis of the number of random distinguished sets that must be examined to maintain high accuracy can be found in Section 8.

To gain the maximum decomposition of the problem, we want j to be as large as possible, but note that j cannot be greater than k, and we saw earlier that we want $k \leq N$ for efficiency reasons. Furthermore, when $j = k$, there is only a single set of k pixels containing the distinguished set and thus the only information that is gained is which curves go through every pixel in the

[3] This assumes that $k \leq N$. If $k > N$, we require $O(n^k) > O(n^N)$ time, but this is inefficient, since we can achieve $O(n^N)$ time by using $k = N$. For this reason, we will not further consider using $k > N$.

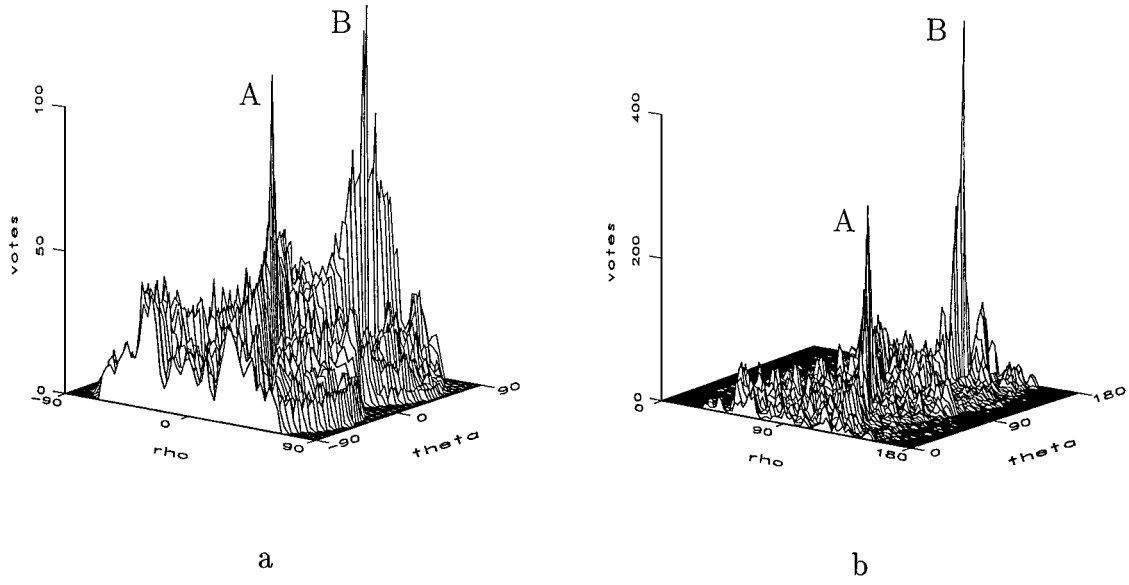

FIG. 2. Transforms of the image in Fig. 1. The peaks labeled A and B correspond to the line segments A and B. (a) Standard Hough transform. (b) Transform mapping pairs of pixels into the parameter space.

distinguished set. In other words, when $j = k$ we have

$$\binom{H(\Omega) - j}{k - j} = \binom{H(\Omega) - j}{0} = 1,$$

regardless of the value of $H(\Omega)$ and thus little information is gained. The optimal cardinality for the distinguished set is thus $j = k - 1 = N - 1$. This maximum decomposition of the problem allows each of the subproblems to be processed quickly and the best efficiency is achieved by the overall algorithm with this decomposition when randomization is used.

Figure 3 shows three examples of constrained Hough transforms for the image in Fig. 1. The first plot shows a case where a pixel on segment A was used as the distinguished pixel, the second plot shows a case where a pixel on segment B was used as the distinguished pixel, and the third plot shows a case where a noise pixel was used as the distinguished pixel. Once again, we have not propagated the localization error into the parameter space in these examples. The propagation of error will be examined in the following section. Note, though, that peaks are present where appropriate, but that no peak is present when a noise pixel was used as the distinguished pixel. Notice also that considering sets of edge pixels that vary in only one pixel (i.e., when $j = k - 1 = N - 1$) constrains the transform to lie on a one-dimensional manifold (a curve) in the parameter space. Let us call this curve the *Hough curve*. When localization error is considered, the transform is no longer constrained to lie on the Hough curve, but the transform points remain close to this curve.

This decomposition of the curve detection problem into subproblems has two very useful properties, both of which derive from the fact that the transform is constrained to lie on the Hough

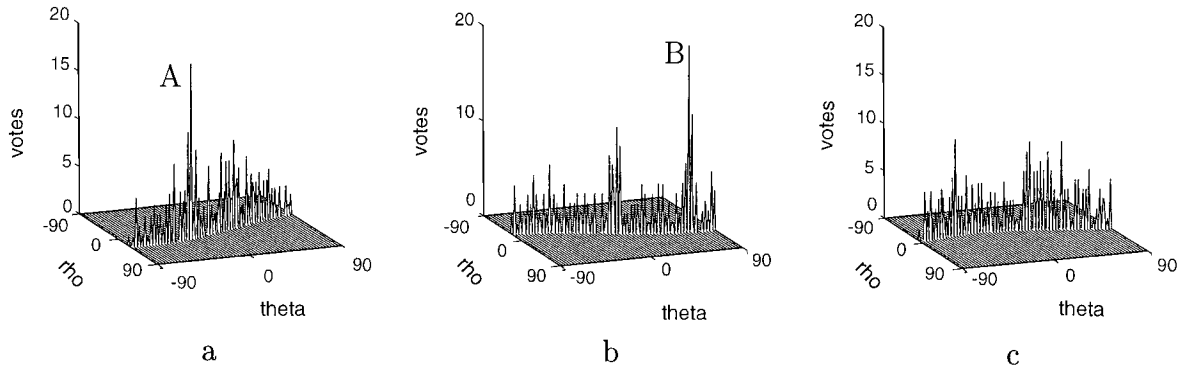

FIG. 3. Constrained Hough transforms of the image in Fig. 1. (a) A pixel from segment A is the distinguished pixel. (b) A pixel from segment B is the distinguished pixel. (c) A noise pixel is the distinguished pixel.

curve in the errorless case. First, since the Hough curve can be parameterized in a single variable, it is much easier to search than the full parameter space. Second, it is now much easier to propagate localization error into the parameter space. This is accomplished by determining tight bounds on the range that a set of pixels can map to in the parameter space under certain localization error bounds.

6. ERROR PROPAGATION

We now consider methods to propagate the localization error into the parameter space. Here we take localization error to encompass both the error in determining the precise position of the edge pixels in the image and any discretization error that is incurred in the image space. The conventional method for treating localization error is to use a binning procedure in the parameter space, where each set of pixels maps into a single bin or a rectilinear volume of cells in the parameter space. As previously noted, this assumes that the bins are large enough to catch the votes from the correct sets of pixels and are small enough not to catch enough votes to result in false positives. This is very difficult to achieve in practice, since the shape of the volume of the parameter space that is consistent with a set of pixels up to the localization error is not rectilinear and the volumes have very different shapes and sizes depending on the locations of the pixels in the set.

An additional problem with such techniques is that the parameter space bins are not infinitesimal. This implies that each bin in the parameter space maps to some noninfinitesimal area in the image space. Let us consider the case of straight lines in particular. Each bin covers some set of parameters, $[\rho_1, \rho_2] \times [\theta_1, \theta_2]$, in the ρ-θ parameterization of lines. This sweeps out an area in the image space that is shaped somewhat like an hourglass (see Fig. 4). A detailed analysis of this shape can be found in [3]. Note that this shape expands as it moves out from its point that is closest to the origin in the coordinate frame. A line passing through the wide region of the hourglass contributes several votes to the bin, even though the line itself has parameters very different from those in the bin. This factor contributes to false positives in Hough transform implementations that use the accumulator method.

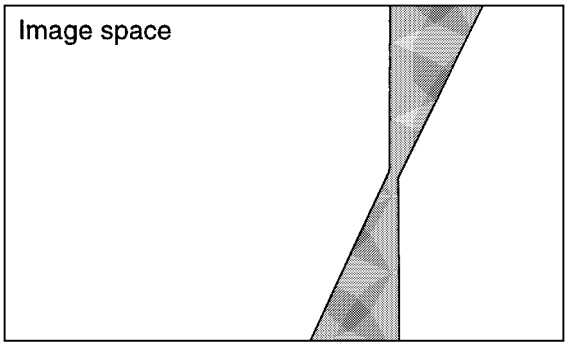

FIG. 4. A cell in the parameter space sweeps out approximately an hourglass shape in the image space for the ρ-θ parameterization of lines.

Now, let us examine how localization error should be propagated ideally in the curve detection process. Each set of pixels maps to a subset of the parameter space under given error conditions. This subset consists of all of the curves that pass through each pixel in the set up to the localization error. Call this subset of the parameter space the *error cloud* of the set of pixels. Ideally, we would locate exactly those points in the parameter space at which some predetermined number of error clouds intersect. This would yield the curves that pass through some minimum number of the points up to the localization error. We do not do this precisely, since it is not practical. However, for the subproblems that we now examine, we can efficiently compute a good approximation to this number.

We first parameterize the Hough curve in a single variable, t. Consider the projection of the error clouds onto the t-axis for each of the pixel sets that are examined in some subproblem (examples of these projections for the cases of lines and circles can be found below). The number of projected error clouds that intersect at some point on the Hough curve yields a bound on the number of error clouds that intersect on a corresponding hypersurface in the full space. Furthermore, since the error clouds do not stray far from the Hough curve, this bound is a good approximation to the actual number of intersecting error clouds, which is the information we desire.

Since we sum the number of projected error clouds that intersect at points on the Hough curve, this corresponds to a kernel function for each error cloud that looks like a top hat. The kernel function takes on a value of 0 or 1 at each point depending on the whether the point is contained in the projection of the error cloud. If desired, we could instead map each error cloud into a smooth, continuous kernel on the Hough curve using the techniques described by Princen *et al.* [33] and Palmer *et al.* [30]. This yields the possibility of achieving better performance at the cost of additional computation.

Once we have projected each of the pixel sets that are examined in some subproblem onto the t-axis, we can find the peaks along the Hough curve using any of several different techniques. We could simply discretize t and perform voting by incrementing the bins consistent with each range in t that an error cloud projects to. This does not suffer from the problems of previous accumulator methods, since we can discretize t finely and increment all of the cells that are consistent with a particular error cloud. Alternatively, we could sort the minimal and maximal t points for each error cloud and use a sweep algorithm. This method examines the extremal points in sorted order and keeps a counter that is incremented each time we hit a minimal point and decremented each time we hit a maximal point. If the counter reaches a large enough value, then a line has been found which passes through (or close to) many edge pixels.

Note that the reason that we can project the error clouds onto a single parameter axis in this method without exacerbating the

problem of false positives is that we have already constrained the error clouds to lie nearly on a one-dimensional manifold of the parameter space. However, we must take care to parameterize this manifold correctly, such that the mapping between points on the manifold and points in the parameterization is one-to-one. The following subsections describe how we can parameterize the Hough curve in t for the cases of lines and circles and how we can project the error cloud for the sets of edge pixels onto the t-axis for each case.

6.1. Lines

If we use the ρ-θ parameterization for lines (i.e., $x\cos\theta + y\sin\theta = \rho$), we can parameterize the Hough curve by θ, since ρ is a function of θ. To project the error cloud for a pair of pixels onto the θ-axis, we simply determine the minimal and maximal θ such that there exists a line with that orientation passing through both of the pixels up to the localization error. If we use square error boundaries, we need only consider the corners of the squares in determining these minimal and maximal θ values. See Figure 5.

6.2. Circles

We can parameterize the space of circles by the coordinates of the center of the circle and the radius, so there are three parameters: (x_c, y_c, r). For this case, the optimal decomposition uses $j = N - 1 = 2$ distinguished pixels. The Hough curve can be parameterized by the distance from the center of the circle to the midpoint between the two distinguished pixels (we take this distance to be positive when the center is to the right of the segment connecting the distinguished pixels, and negative otherwise).

To project the error cloud onto the t-axis, we now need to determine error bounds on this distance given three points and their localization error boundaries. Recall that the center of the circle passing through three points is the point where the perpendicular bisectors of the segments between the points meet. We can thus determine bounds on the location of the center of the circle by examining the set of points at which two of the perpendicular bisectors of the segments can meet (see Fig. 6). The minimum and maximum distance from the center of the circle to the midpoint of the distinguished pixels can be determined by examining the extremal points of this set.

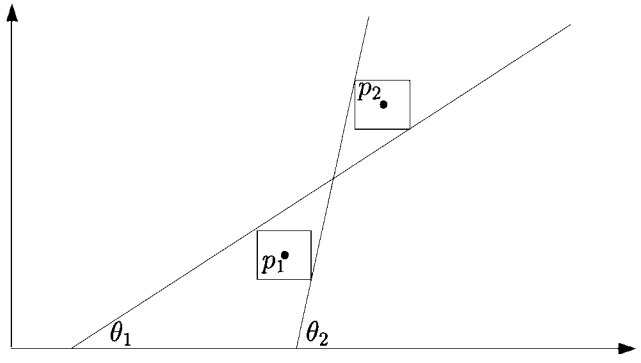

FIG. 5. We can determine bounds on the range of θ for any two pixels by considering lines that pass through the boundaries of their possible localization errors.

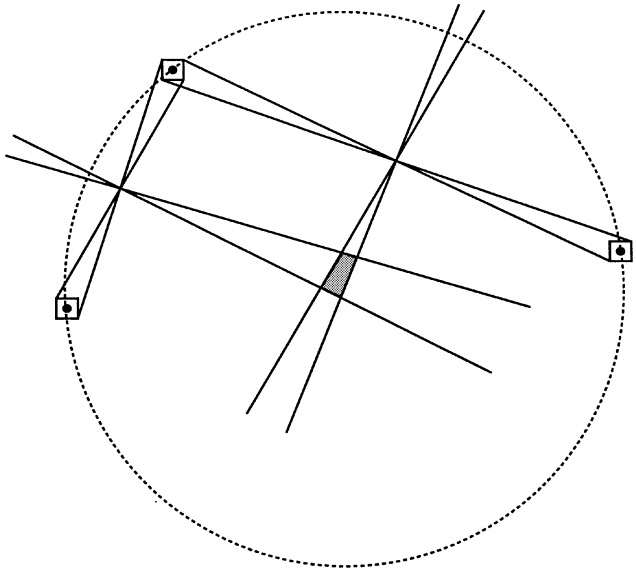

FIG. 6. We can determine bounds on the position of the center of a circle passing through three points (up to the localization error) by examining the range of possible perpendicular bisectors for the segments between the points.

7. ANALYSIS AND COMPARISON WITH PREVIOUS METHODS

In order to study the rate of false positives that these methods yield, let us examine, in the context of line detection, the average number of error clouds that are consistent with a particular point on the Hough curve for a single trial. Since there are $n-1$ such error clouds that are examined and the Hough curve covers π radians, the average number of error clouds consistent with each point on the Hough curve is

$$E[N_C] = \frac{(n-1)E[\theta_d]}{\pi}, \tag{10}$$

where $E[\theta_d]$ is the expected length of the projection of the error cloud onto the Hough curve.

We can place an upper bound on θ_d, the length of the projection of an error cloud for a particular pair of points onto the Hough curve as follows. Consider the triangle formed by the midpoint between the pair of pixels and the two corners of one of the points' localization error boundaries that form an acute triangle with this midpoint (see Fig. 7). θ_d is given by the angle of the triangle formed at the midpoint between the pixels.

The side of the triangle opposite to θ_d has length $\sqrt{2}\gamma$, where γ is the length of a side of the localization error boundary. Let l_1 and l_2 be the other edges of the triangle. The law of cosines

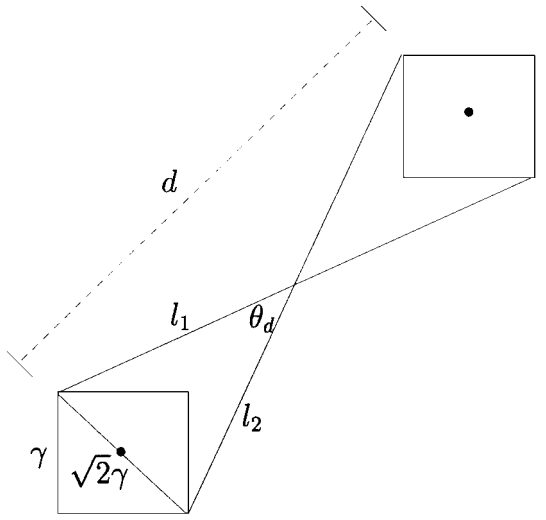

FIG. 7. The range of possible θ values can be determined using the law of cosines.

yields

$$2\gamma^2 = l_1^2 + l_2^2 - 2l_1 l_2 \cos\theta_d. \tag{11}$$

If d is the distance between the pair of pixels, then we have $l_1, l_2 > \frac{d}{2}$ and

$$4\gamma^2 < d^2 - d^2 \cos\theta_d \tag{12}$$

$$\theta_d < \arccos\left(1 - \frac{4\gamma^2}{d^2}\right). \tag{13}$$

$E[\theta_d]$ is not straightforward to determine, but we can compute it empirically by examining the distribution of d in sample images. For example, for the image in Fig. 1, we get $E[\theta_d] = 0.08485$, with $\gamma = 1$ pixel, and since there are 262 edge pixels in the image, we get $E[N_C] = 6.21$. So, the average number of votes that each point on the Hough curve receives (including the peaks due to the true lines) is just over six. Meanwhile, the peaks for the curves in this case are 34 and 41 votes high. Figure 8 shows constrained transforms of the image in Fig. 1 with error propagation. The distinguished points used are the same as in Fig. 3. Note that the signal-to-noise ratio is higher when the error is propagated into the parameter space.

Let us contrast this with a method that simply divides the Hough curve into bins and, for each pair of pixels that is considered, casts votes for some set of bins centered at the bin that the pair of pixels would map into if they had no localization error. In contrast to the previous method, this method can make no guarantee that the correct bin is voted for, even if it intersects the error cloud of the pair of points, since it does not accurately model the localization error. However, the pixels must vote for enough bins that it is likely that they hit the correct bin. This means that the redundancy, $\frac{E[\theta_d]}{\pi}$, should be at least as large as in the case above (and probably larger), yet since some predetermined constant size is used, the redundancy is underestimated for some pixels and overestimated for others. We thus miss some correct lines due to the underestimations and find some false positives due to the overestimations.

For the standard Hough transform and the techniques where set of pixels are mapped into the parameter space, we have exactly the same problems, but now we have underestimates and overestimates in each of the dimensions of the parameter space and thus the problems may be compounded.

We have performed empirical tests using four methods to detect curves in synthetic images. The methods that we compare are:

1. The method described in this paper, where subproblems are examined and the localization error is propagated into the parameter space. A 900 bin accumulator was used.
2. The method where subproblems are examined, but without propagation of the localization error into the parameter space. A 900 bin accumulator was used.

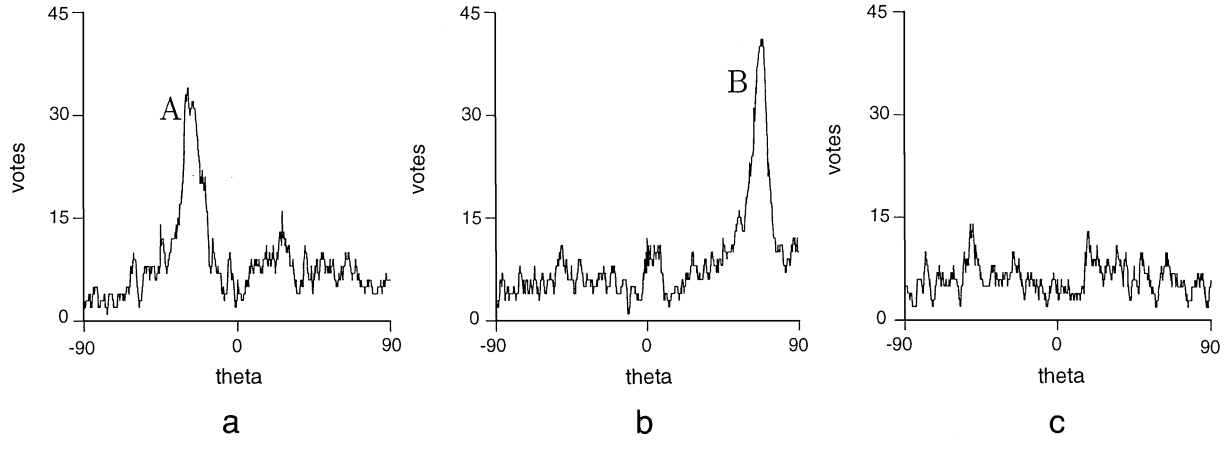

FIG. 8. Constrained Hough transforms of the image in Fig. 1 with error propagation. (a) A pixel from segment A was used as the distinguished pixel. (b) A pixel from segment B was used as the distinguished pixel. (c) A noise pixel was used as the distinguished pixel.

3. The method where pairs of pixels are mapped into the parameter space without propagation of the localization error into the parameter space. A 360×360 accumulator was used.

4. The standard Hough transform method. A 360×360 accumulator was used.

To ameliorate the effects of error in the cases where localization error is not propagated into the parameter space, votes are cast for all cells that are adjacent (including those sharing only a corner) to the cells that are hit precisely in the transform.

The synthetic test images were generated by placing a straight line consisting of 64 pixels in a 256×256 image. In addition, 1000 randomly selected pixels and two distractors were added to the images. The distractors consisted of circular arcs with a radius of curvature of 100 pixels. An example can be found in Fig. 9.

Tests were performed on 1000 such synthetic images. Figure 10 shows the results of these tests. For each method, the probability of detecting the correct line segment is plotted versus the probability of finding a false positive for varying levels of the threshold that is used to determine which lines are detected. The method described in this paper, where the localization error is propagated into the parameter space has by far the best performance among the methods tested. Somewhat surprising is the poor performance of the method where subproblems are examined, but that localization error is not propagated into the parameter space. This performance is due to a combination of the inaccurate propagation of error with the use of constraints (the distinguished pixel) that contain error. When a distinguished pixel with very little error is chosen, the performance is acceptable, but when no such distinguished pixel is examined, the performance becomes poor. This indicates that when constrained subproblems are examined, it is crucial to propagate the localization error accurately into the parameter space.

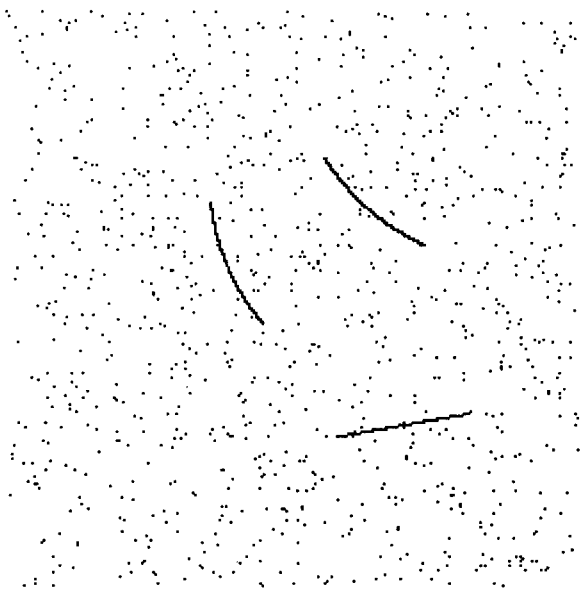

FIG. 9. Example synthetic image that was used in generating receiver operating characteristic (ROC) curves.

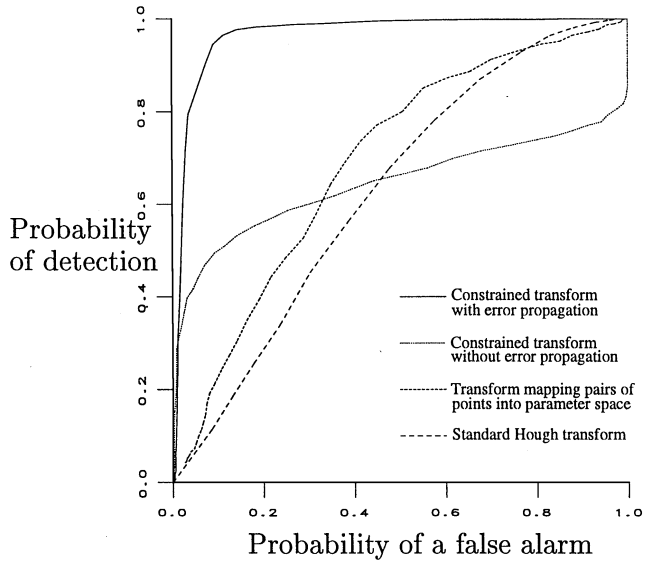

FIG. 10. ROC curves generated using synthetic data.

8. COMPUTATIONAL COMPLEXITY

This section examines the computational complexity of curve detection using the techniques described above. Let us first determine how many of the distinguished sets must be examined to maintain a low rate of failure. We assume that we only need to find curves that comprise some fraction ϵ of the total number of edge pixels in the image and thus are significant with respect to the complexity of the image. The probability that a single set of j random pixels all lie on a particular such curve can be bounded by

$$p_0 \geq \frac{\binom{\epsilon n}{j}}{\binom{n}{j}} \approx \frac{(\epsilon n)^j}{n^j} = \epsilon^j. \tag{14}$$

This follows since we must have $\binom{\epsilon n}{j}$ distinguished sets that lie on the curve among the $\binom{n}{j}$ possible distinguished sets. If we take t such trials, the probability that all of them fail for a particular curve is bounded by

$$p \leq (1 - p_0)^t \approx (1 - \epsilon^j)^t. \tag{15}$$

For each curve, we thus have a probability no larger than p that we fail to examine a set of distinguished pixels that is a subset of the curve in t trials. Since conservative peak finding techniques are used, we can assume that any trial examining a correct set of distinguished pixels leads to the identification of the curve.

We now choose an arbitrarily small probability of failure, δ, and determine the number of trials necessary to guarantee this

accuracy:

$$(1 - \epsilon^j)^t \leq \delta. \tag{16}$$

Solving for t yields

$$t \log(1 - \epsilon^j) \geq \log \delta \tag{17}$$

$$t_{min} = \frac{\log \delta}{\log(1 - \epsilon^j)} \approx \frac{\log \frac{1}{\delta}}{\epsilon^j}. \tag{18}$$

The number of trials necessary is thus dependent on the probability of success desired (but only to a logarithmic factor), the fraction of image edge pixels that must comprise the curve, and the size of the distinguished set. Note that while the number of trials is exponential in the size of the distinguished set, the number of bins we have to increment per trial in the parameter space is inversely exponential in this size. The number of trials is not explicitly dependent on n, although it is implicitly dependent on n. This is because, as the complexity of the image increases, we need to lower ϵ to detect the same curves, although these curves become less significant with respect to the complexity of the image.

Now let us consider the complexity of detecting peaks on the Hough curve for each trial. Recall that we use $j = N - 1$ in our method. For the technique of discretizing the Hough curve and voting for the bins over an appropriate range for each set of pixels that is mapped onto the Hough curve, the complexity is dependent upon how finely the Hough curve is discretized. If there are α bins, then we need to increment $O(\alpha\gamma)$ bins per trial per edge pixel, and this yields a complexity of $O(n\alpha\gamma)$ per trial. The total complexity is thus $O(\frac{n\alpha\gamma \log \frac{1}{\delta}}{\epsilon^{N-1}})$ or simply $O(n)$ when measured by the size of the input (α, γ, δ, ϵ, and N are constants). The storage required by this technique is $O(n + \alpha)$.

If we instead use the sweep algorithm, we must sort the $O(n)$ minimal and maximal points of the error cloud projections onto the Hough curve, which requires $O(n \log n)$ time per trial. Processing the sorted points requires $O(n)$ time. We thus require $O(\frac{n \log n \log \frac{1}{\delta}}{\epsilon^{N-1}})$ total time or $O(n \log n)$ when measured by the size of the input. The storage required by this technique is $O(n)$.

While the time that is required is exponential in the number of curve parameters, N, the base of the exponential is now ϵ rather than n or α as in previous algorithms. We thus have an accurate algorithm in which the time dependence on the number of edge pixels and the accuracy of the algorithm is low.

An additional area where randomization may be used if we are willing to sacrifice some performance in the discrimination of true curves versus false positives is by subsampling the pixels that are examined in each subproblem. The analysis of Bergen and Shvaytser [2] implies that we can achieve a complexity that is independent of the number of image edge pixels if we are willing to allow the method to be in error with slightly increased frequency. We have not fully explored this possibility, since its practicality is questionable (the number of sampled pixels necessary may be larger than the total number of edge pixels in the image).

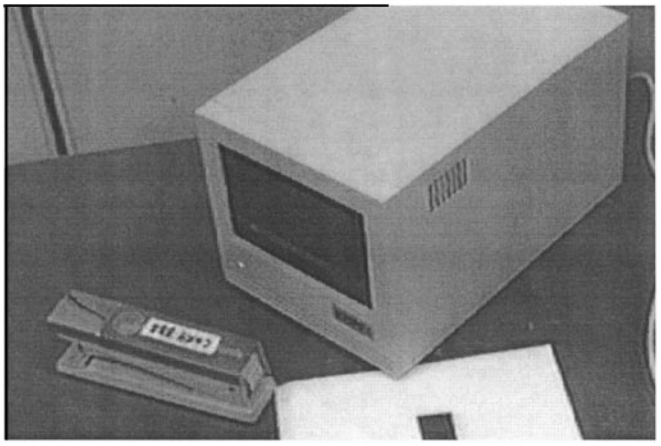

FIG. 11. Example image used to test line detection.

9. RESULTS

These techniques have been applied to real images to test their efficacy. Figure 11 shows an example image (483 × 318 pixels) that was used to test the line detection techniques. Figure 12 shows the edges found in this image. These edges were determined to subpixel accuracy using a version of the Canny edge detector [6].

For tests on this image, square localization error boundaries were used such that the true location of each edge pixel was assumed to be within 0.25 pixels of the measured location in each direction. For each line that surpasses the detection threshold in each subproblem, only the parameters at which the peaks occurred were kept. Furthermore, for any two lines that were detected in separate subproblems that were within some minimum distance of each other, only the line with the higher vote count was kept. Finally, we output the connected segments of

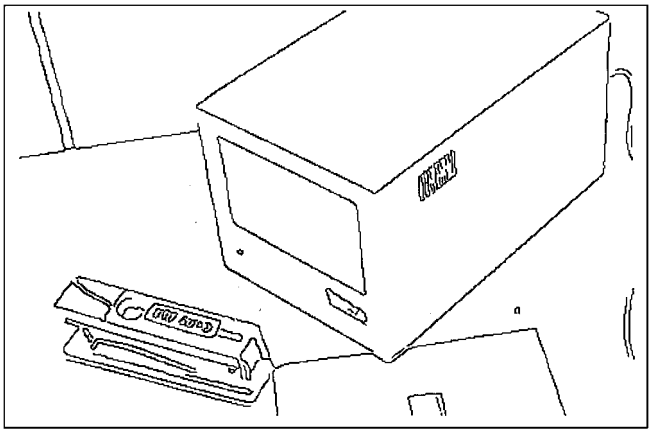

FIG. 12. Edges detected in the image.

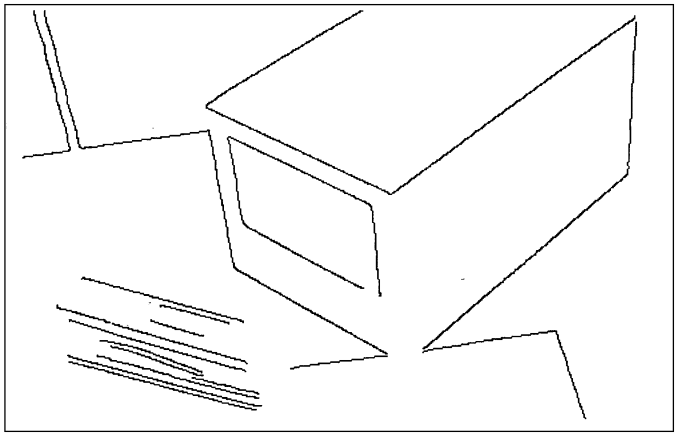

FIG. 13. Straight lines detected with $\epsilon = .01$.

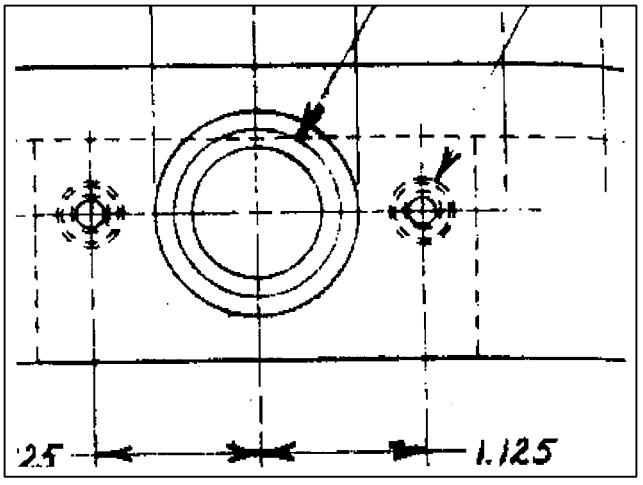

FIG. 15. Engineering drawing used to test circle detection.

edge pixels along the lines that were detected in the image using a method that allowed small gaps to be bridged. Figures 13 and 14 show the lines that were detected at two thresholds. When a large threshold was used ($\epsilon = 0.01$), all of the long lines were found in the image, but short or curving lines were not found. When a lower threshold was used ($\epsilon = 0.004$), even short lines were found in the image.

Figure 15 shows an image (400×300 pixels) that was used to test circle detection using the method described in this work. This image is an engineering drawing that has been scanned. For this reason, it was not possible to determine the location of edge pixels to subpixel accuracy. In addition, the presence of small and dashed circles and the clutter in the image make this a difficult test case. For this circle detection example, we used the additional constraint that each circle that is output should be represented in the image by at least some minimum fraction of its perimeter edge pixels.

Figure 16 shows the circles found with $\epsilon = .04$. While all of the large circles were found, the small and dashed circles did not comprise a large enough fraction of the image to be found. With $\epsilon = .008$, the implementation finds a number of circles, some of which are not perceptually obvious. Figure 17 shows the circles found for this case that are perceptually salient. The implementation had difficulty finding both of the dashed circles with the same center since they were so close together. The circles shown consist of the top half of one of the circles and the bottom half of the other. This may partially be a result of the circles not being perfectly circular in the image. Figure 18 shows circles that were found that are not perceptually salient. Note that in each case, the pixels found form most of the perimeter of a circle. These circles successfully met the acceptance criterion specified, so this is not a failure of the algorithm.

10. ELLIPSES AND OTHER HIGH-ORDER CURVES

When applying these techniques to curves with many degrees of freedom, we must take special care, since the number of trials that are required can become large. Let us consider the detection

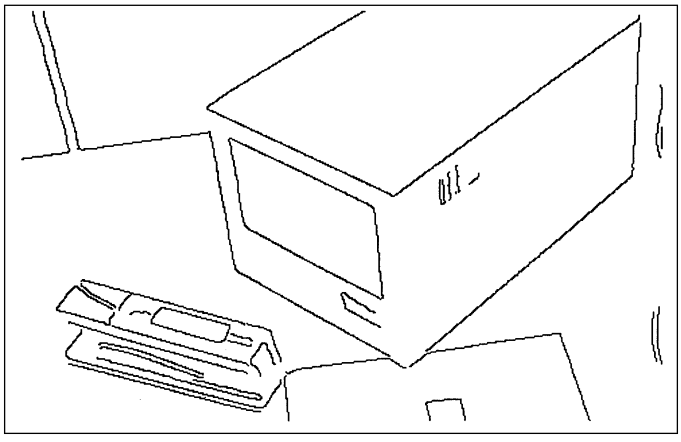

FIG. 14. Straight lines detected with $\epsilon = .004$.

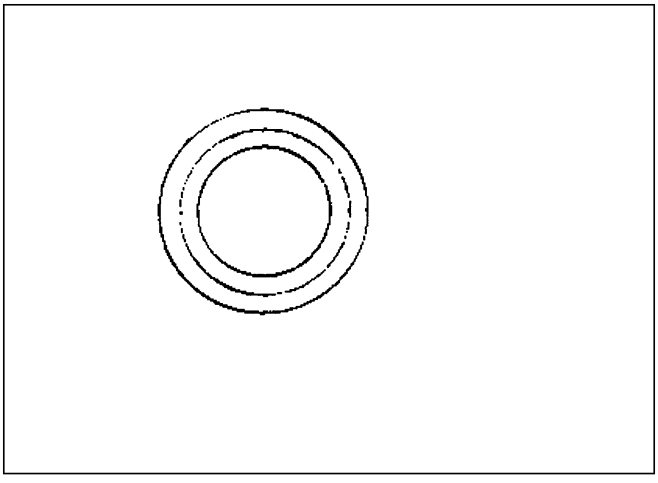

FIG. 16. Circles detected with $\epsilon = .04$.

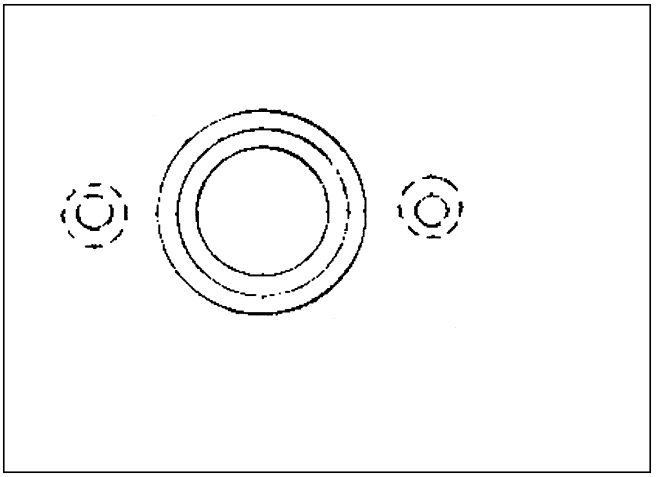

FIG. 17. Perceptually salient circles detected with $\epsilon = .008$.

of ellipses, which have five parameters. If the image is sparse, or we can segment the image, then we should have no problems. For example, if we only need to detect ellipses that comprise 50% of the image edges (or some subset after segmentation), then the number of trials required to achieve 0.99 probability of success is 74. On the other hand, if we wish to detect ellipses that comprise at least 10% of the image edges using these techniques in a straightforward manner, then we require 46,052 trials to achieve 0.99 probability of success.

When we wish to detect high-order curves in complex images, there are additional techniques that we can use to perform curve detection quickly. One simple technique is to use additional information at each edge pixel. For example, we can use the orientation of the curve at each pixel (as determined from the gradient or the curve normal or tangent). When we do this, we require fewer curve points to determine the position of the curve. We can determine the position of an ellipse using three oriented points[4] rather than five unoriented points. We would thus use two, rather than four, distinguished pixels, and we would require many fewer trials to ensure that there is a low probability of failure (461 for the example given above). Of course, we need not restrict this technique to high-order curves. We can use two oriented points to determine the position of a circle, rather than three unoriented points.

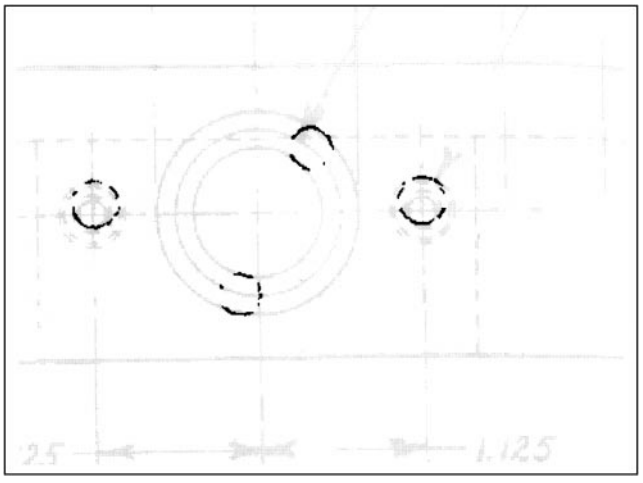

FIG. 18. Perceptually insalient circles that surpassed the threshold with $\epsilon = .008$. The original image is drawn in lightly to show why these were detected.

An alternate technique that can detect high-order curves quickly is to use a two (or more) step technique, where some subset of the curve parameters is determined first, and the remaining parameters are determined subsequently. An example of this technique is the method of finding the center of an ellipse that is described by Tsuji and Matsumoto [45]. They note that the points on an ellipse that have parallel tangent lines lie on opposite sides of the center of the ellipse, and thus ellipse centers can be detected by finding points that are midway between many such pairs of points with parallel tangents. Problem decomposition techniques similar to those described here can also be used with this method to detect ellipse centers. Once the center of the ellipse has been detected, three parameters remain to be determined. These can be determined using a three parameter Hough transform technique, similar to the detection of circles.

11. SUMMARY

This paper has discussed efficient and accurate methods to perform curve detection using a decomposition of the Hough transform that allows localization error to be efficient propagated into the parameter space. To this end, we have modified a formal definition of the Hough transform that allows the localization error to be analyzed appropriately. Under this definition, it was shown that the mapping of pixel sets (rather than individual pixels) into the parameter space did not, by itself, improve the accuracy or efficiency of curve detection.

We then considered a new method where the Hough transform is decomposed into several constrained subproblems, each of which examines a subset of the parameter space by considering only those pixel sets that include some distinguished set of pixels. If each possible subproblem is examined, then no loss in performance results, but no decrease in complexity is gained. However, the examination of these subproblems allows us, first, to propagate the localization error efficiently and accurately into the parameter space, and second, to use randomization techniques to reduce the complexity of curve detection, while maintaining a low probability of failure. The overall complexity of the resulting algorithm is $O(n)$ or $O(n \log n)$ (depending on the method used to find peaks in the parameter space), where n is the number of edge pixels in the image. In addition, only $O(n)$ storage is necessary.

[4] In fact, three points with orientations overconstrain the position of the ellipse in the errorless case.

Analysis of this method indicates that curves can be detected robustly with a lower rate of false positives than previous methods to perform curve detection. An empirical investigation confirmed this result and demonstrated that even in the presence of random noise and correlated distractors, this technique can detect straight lines without a high rate of false positives, where previous methods break down. We have given examples of these techniques detecting straight lines and circles in real images.

Further improvements to these techniques and extensions to curves with many degrees of freedom are possible through the use of additional information. For example, the orientation of each edge pixel can be used as an additional constraint on the location of a curve. This reduces the number of points that is necessary to solve for the parameters of a curve and thus reduces the dependence of the algorithm on the number of curve parameters.

The primary contribution of this work is twofold. First, the technique of subdividing the Hough transform into many small subproblems has previously only been considered in a very limited fashion. We have formalized this technique and shown that it allows randomization to be used in a manner that introduces a small probability of failure, since a correct distinguished set of pixels may not be examined for a particular curve, but that it does not reduce the detection performance, assuming that a correct distinguished set of pixels is examined. Second, we have described new techniques by which the localization error present in the image edge features may be propagated into the parameter space allowing curves to be detected robustly. The combination of these techniques results in an algorithm that combines both efficiency of operation and robustness of curve detection performance, without the detection of false positives that are not supported by the image.

APPENDIX

Parameterizations

This appendix describes parameterizations for lines, circles, and ellipses that can be used in Hough transform implementations.

A.1. Lines

One parameterization for lines is the standard slope–intercept representation:

$$y = mx + b. \tag{19}$$

While this parameterization has the advantage that the points in the image space map to lines in the parameter space, it has the disadvantage that the ranges of m and b are unbounded, since we may have vertical or horizontal lines.

Duda and Hart [9] proposed to use the normal parameterization instead:

$$x\cos\theta + y\sin\theta = \rho. \tag{20}$$

In this parameterization, θ is the orientation of the line and ρ is the distance of the line from the origin. While ρ is theoretically unbounded, for any image we can bound ρ by the maximum distance of any image point from the origin. This parameterization maps points in the image space to sinusoids in the parameter space.

Given two points (x_1, y_1) and (x_2, y_2) we can solve for θ and ρ as follows:

$$\theta = \arctan\frac{x_1 - x_2}{y_2 - y_1} \tag{21}$$

$$\rho = x_1\cos\theta + y_1\sin\theta. \tag{22}$$

For any distinguished point, (x_d, y_d), the Hough curve is given by $x_d\cos\theta + y_d\sin\theta = \rho$. In addition, ρ is a function of θ, so we can parameterize the Hough curve by θ.

A.2. Circles

The standard parameterization for circles is (x_c, y_c, r), where (x_c, y_c) is the center of the circle and r is the radius. The circle is given by the solutions to

$$(x - x_c)^2 + (y - y_c)^2 = r^2. \tag{23}$$

Given three points, (x_1, y_1), (x_2, y_2), and (x_3, y_3), the circle that passes through each of them has parameters

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \frac{\begin{bmatrix} y_3 - y_1 & y_1 - y_2 \\ x_1 - x_3 & x_2 - x_1 \end{bmatrix}\begin{bmatrix} x_2^2 + y_2^2 - x_1^2 - y_1^2 \\ x_3^2 + y_3^2 - x_1^2 - y_1^2 \end{bmatrix}}{(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)} \tag{24}$$

$$r = \sqrt{(x_1 - x_c)^2 + (y_1 - y_c)^2}. \tag{25}$$

A slightly different parameterization that is useful in the context of Hough curves is (x_c, y_c, d), where d is the distance of the center of the circle from the segment connecting the two distinguished points. If we choose d to be positive when the center of the circle is to the right of the segment and negative when it is to the left[5], we can reparameterize the curve in one variable by d. In this case the Hough curve forms a straight line in the parameter space.

A.3. Ellipses

Ellipses are typically parameterized by (x_c, y_c, a, b, θ), where (x_c, y_c) is the center of the ellipse, a and b are the lengths of the major and minor axes, and θ is the orientation of the major axis. Forbes [10] determined a stable method to solve for the parameters (a description was found by this author in [20]). Given (x_i, y_i), $i = 1 \ldots 5$ we can determine the coefficients in the

[5] If the segment is horizontal, then left and right of the segment have no meaning. In this case, we take above be positive and below to be negative.

following equation and solve for (U, V, R, S, T) using Gaussian elimination:

$$x^2 + y^2 - U(x^2 - y^2) - 2Vxy - Rx - Sy - T = 0. \quad (26)$$

We then solve for (x_c, y_c, a, b, θ) in the following equations:

$$e = \frac{b}{a} \quad (27)$$

$$U = \cos 2\theta \frac{1 - e^2}{1 + e^2} \quad (28)$$

$$V = \sin 2\theta \frac{1 - e^2}{1 + e^2} \quad (29)$$

$$R = 2x_c(1 - U) - 2y_c V \quad (30)$$

$$S = 2y_c(1 - U) - 2x_c V \quad (31)$$

$$T = \frac{2a^2b^2}{a^2 + b^2} - \frac{x_c R}{2} - \frac{x_c S}{2}. \quad (32)$$

ACKNOWLEDGMENTS

This research was performed while the author was with Cornell University, Ithaca, NY. A preliminary version of this work appeared in the 1996 European Conference on Computer Vision [27].

REFERENCES

1. M. Atiquzzaman, Multiresolution Hough transform—An efficient method of detecting patterns in images, *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(11), 1992, 1090–1095.
2. J. R. Bergen and H. Shvaytser, A probabilistic algorithm for computing Hough transforms, *J. Algorithms* **12**, 1991, 639–656.
3. T. M. Breuel, Finding lines under bounder error, *Pattern Recog.* **29**(1), 1996, 167–178.
4. C. M. Brown, Inherent bias and noise in the Hough transform, *IEEE Trans. Pattern Anal. Mach. Intell.* **5**(5), 1983, 493–505.
5. A. Califano and R. M. Bolle, The multiple window parameter transform, *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(12), 1992, 1157–1170.
6. J. Canny, A computational approach to edge detection, *IEEE Trans. Pattern Anal. Mach. Intell.* **8**(6), 1986, 679–697.
7. T. A. Cass, *Polynomial-Time Geometric Matching for Object Recognition*, Ph.D. thesis, Massachusetts Institute of Technology, Feb. 1993.
8. M. Cohen and G. T. Toussaint, On the detection of structures in noisy pictures, *Pattern Recog.* **9**, 1977, 95–98.
9. R. O. Duda and P. E. Hart, Use of the Hough transformation to detect lines and curves in pictures, *Commun. Assoc. Comput. Mach.* **15**, 1972, 11–15.
10. A. B. Forbes, *Fitting an Ellipse to Data*, Report NPL-DITC 95/87, National Physical Laboratory, Dec. 1987.
11. G. Gerig, Linking image-space and accumulator space: A new approach for object-recognition, in *Proceedings of the International Conference on Computer Vision, 1987*, pp. 112–115.
12. W. E. L. Grimson and D. P. Huttenlocher, On the sensitivity of the Hough transform for object recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* **12**(3), 1990, 255–274.
13. P. V. C. Hough, Method and means for recognizing complex patterns, U.S. Patent 3069654, 1962.
14. D. J. Hunt, L. W. Nolte, A. R. Reibman, and W. H. Ruedger, Hough transform and signal detection theory performance for images with additive noise, *Comput. Vision Graphics Image Process.* **52**, 1990, 386–401.
15. J. Illingworth, G. Jones, J. Kittler, M. Petrou, and J. Princen, Robust methods of 2d and 3d image description, in *Progress in Image Analysis and Processing II* (V. Cantoni, M. Ferretti, S. Levialdi, R. Negrini, and R. Stefanelli, Eds.), pp. 3–26, World Scientific, Singapore, 1991.
16. J. Illingworth and J. Kittler, The adaptive Hough transform, *IEEE Trans. Pattern Anal. Mach. Intell.* **9**(5), 1987, 690–698.
17. J. Illingworth and J. Kittler, A survey of the Hough transform, *Comput. Vision Graphics Image Process.* **44**, 1988, 87–116.
18. N. Kiryati and A. M. Bruckstein, Antialiasing the Hough transform, *CVGIP: Graphical Models Image Process.* **53**, 1991, 213–222.
19. N. Kiryati, Y. Eldar, and A. M. Bruckstein, A probabilistic Hough transform, *Pattern Recog.* **24**(4), 1991, 303–316.
20. 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**, 1992, 381–398.
21. V. F. Leavers, Which Hough transform?, *CVGIP: Image Understanding* **58**, 1993, 250–264.
22. H. Li, M. A. Lavin, and R. J. Le Master, Fast Hough transform: A hierarchical approach, *Comput. Vision Graphics Image Process.* **36**, 1986, 139–161.
23. P. Liang, A new and efficient transform for curve detection, *J. Robotic Systems* **8**(6), 1991, 841–847.
24. K. Murakami, H. Koshimizu, and K. Hasegawa, On the new Hough algorithms without two-dimensional array for parameter space to detect a set of straight lines, in *Proceedings of the IAPR International Conference on Pattern Recognition, 1986*, pp. 831–833.
25. W. Niblack and D. Petkovic, On improving the accuracy of the Hough transform, *Mach. Vision Appl.* **3**(2), 1990, 87–106.
26. F. O'Gorman and M. B. Clowes, Finding picture edges through collinearity of feature points, *IEEE Trans. Comput.* **25**(4), 1976, 449–456.
27. C. F. Olson, Decomposition of the Hough transform: Curve detection with efficient error propagation, in *Proceedings of the European Conference on Computer Vision, 1996*, Vol. 1, pp. 263–272.
28. C. F. Olson, Efficient pose clustering using a randomized algorithm, *Int. J. Comput. Vision* **23**(2), 1997, 131–147.
29. P. L. Palmer, J. Kittler, and M. Petrou, Using focus of attention with the Hough transform for accurate line parameter estimation, *Pattern Recog.* **27**(9), 1994, 1127–1134.
30. P. L. Palmer, M. Petrou, and J. Kittler, A Hough transform algorithm with a 2d hypothesis testing kernel, *CVGIP: Image Understanding* **58**, 1993, 221–234.
31. J. Princen, J. Illingworth, and J. Kittler, A hierarchical approach to line extraction based on the Hough transform, *Comput. Vision Graphics Image Process.* **52**(1), 1990, 57–77.
32. J. Princen, J. Illingworth, and J. Kittler, A formal definition of the Hough transform: Properties and relationships, *J. Math. Imaging Vision* **1**, 1992, 153–168.
33. J. Princen, J. Illingworth, and J. Kittler, Hypothesis testing: A framework for analyzing and optimizing Hough transform performance, *IEEE Trans. Pattern Anal. Mach. Intell.* **16**(4), Apr. 1994, 329–341.
34. T. Risse, Hough transform for line recognition: Complexity of evidence accumulation and cluster detection, *Comput. Vision Graphics Image Process.* **46**, 1989, 327–345.
35. A. Rosenfeld, *Picture Processing by Computer*, Academic Press, San Diego, 1969.

36. S. D. Shapiro, Transformations for the computer detection of curves in noisy pictures, *Comput. Graphics Image Process.* **4**, 1975, 328–338.
37. S. D. Shapiro, Feature space transforms for curve detection, *Pattern Recog.* **10**, 1978, 129–143.
38. S. D. Shapiro, Generalization of the Hough transform for curve detection in noisy digital images, in *Proceedings of the International Joint Conference on Pattern Recognition*, 1978, pp. 710–714.
39. S. D. Shapiro, Properties of transforms for the detection of curves in noisy pictures, *Comput. Graphics Image Process.* **8**, 1978, 219–236.
40. S. D. Shapiro and A. Iannino, Geometric constructions for predicting Hough transform performance, *IEEE Trans. Pattern Anal. Mach. Intell.* **1**(3), 1979, 310–317.
41. J. Sklansky, On the Hough technique for curve detection, *IEEE Trans. Comput.* **27**(10), 1978, 923–926.
42. M. Soffer and N. Kiryati, Guaranteed convergence of the Hough transform, in *Vision Geometry III, Proc. SPIE 2356, 1994*, pp. 90–100.
43. R. S. Stephens, Probabilistic approach to the Hough transform, *Image Vision Comput.* **9**(1), 1991, 66–71.
44. P. R. Thrift and S. M. Dunn, Approximating point-sets image by line segments using a variation of the Hough transform, *Comput. Vision Graphics Image Process.* **21**, 1983, 383–394.
45. S. Tsuji and F. Matsumoto, Detection of ellipses by a modified Hough transform and simplified interpretation strategy, *IEEE Trans. Comput.* **27**(8), 1978, 777–781.
46. T. M. van Veen and F. C. A. Groen, Discretization errors in the Hough transform, *Pattern Recog.* **14**(1–6), 1981, 137–145.
47. L. Xu and E. Oja, Randomized Hough transform (RHT): Basic mechanisms, algorithms, and computational complexities, *CVGIP: Image Understanding* **57**, 1993, 131–154.
48. L. Xu, E. Oja, and P. Kultanen, A new curve detection method: Randomized Hough transform (RHT), *Pattern Recog. Lett.* **11**, May 1990, 331–338.