Proc. of the European Conference on Computer Vision, pages 263-272, 1996.

Decomposition of the Hough Transform: Curve Detection with Efficient Error Propagation

Clark F. Olson

Department of Computer Science, Cornell University, Ithaca, NY 14853, USA

Abstract. This paper describes techniques to perform fast and accurate curve detection using a variant of the Hough transform. We show that the Hough transform can be decomposed into small subproblems that examine only a subset of the parameter space. Each subproblem considers only those curves that pass through some small subset of the data points. This property allows the efficient implementation of the Hough transform with respect to both time and space, and allows the careful propagation of the effects of localization error in the detection process. The use of randomization yields an $O(n)$ worst-case computational complexity for this method, where n is the number of data points, if we are only required to find curves that are significant with respect to the complexity of the data. In addition, this method requires little memory and can be easily parallelized.

1 Introduction

The Hough transform is a method to detect parameterized models (e.g. curves and surfaces) in data by mapping data features into manifolds in the parameter space [3, 5]. The models are detected by locating peaks in the parameter space (which is typically performed using multi-dimensional histograming). In this paper, we consider methods to improve curve detection by decomposing the Hough transform into many small subproblems. We use randomization to limit the number of subproblems that we must examine and we carefully propagate the effects of localization error in the subproblems that we do examine. While we concentrate on curve detection, similar Hough transform techniques can be applied to surface detection and a number of other problems.

We will use a modified version of the formal definition of the Hough transform given by Princen *et al.* [6]. Let $X = (x, y)$ be a point in the image space, $\Omega = (\omega_1, ..., \omega_N)$ be a point in an N-dimensional parameter space, and $f(X, \Omega) = 0$ be the function that parameterizes the set of curves. We will call the set of data points $\mathcal{E} = \{X_1, ..., X_n\}$.

Standard Hough transform implementations discretize the parameter space and maintain a counter for each cell. The counters record the number of data points that map to a manifold that intersects each of the cells. In the errorless case, each data point maps to an $N - 1$ dimensional manifold in the parameter space. Princen *et al.* denote a cell in parameter space centered at Ω by C_Ω. They define:

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

Thus, $p(X,\Omega)$ is 1 if any curve in the parameter space cell, C_Ω, passes through the point, X, in the image. If we assume that there is no localization error, the Hough transform can then be written:

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

$H(\Omega)$ is now the number of data points that any curve in C_Ω passes through. In an ideal system, the discretization of the parameter space would not be important. Instead, we should consider the error in the localization of the image points. Let's assume that the true location of each data point lies within a bounded region, N_X, of the determined location, X. We can redefine $p(X,\Omega)$ as follows:

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

Now, $p(X,\Omega)$ is 1 if the curve represented by Ω passes through N_X. With this definition we can still use (1) to describe the Hough transform. This yields, for each curve, the number of data points that the curve passes through up to the localization error. Since discretization of the parameter space will not be important for the techniques that we present here, we will use the new definition.

2 Mapping Point Sets into the Parameter Space

A technique that has been recently introduced [1, 2, 4, 7] maps point sets rather that single points into the parameter space. Rather than considering each point separately, this method considers point sets of some cardinality, k. For each such set, the curves that pass through each point in the set (or their error boundaries) are determined and the parameter space is incremented accordingly. The benefit of this technique is that each mapping is to a smaller subset of the parameter space. If the curve has N parameters, then, in the errorless case, N non-degenerate data points map to a finite set of points in the parameter space. For the curves we examine here, this will be a single point. We thus need to increment only one bin in the parameter space for each set, rather than the bins covering an $N-1$ dimensional manifold for each point. Of course, we don't need to use sets of size N, we could use any size, $k > 0$. If $k \leq N$, each non-degenerate set maps to an $N-k$ dimensional manifold in the parameter space. The disadvantage to methods that map point sets into the parameter space is that there are $\binom{n}{k}$ sets of image pixels with cardinality k to be considered.

An examination of how the technique of mapping point sets into the parameter space is related to the standard Hough transform is informative. Let's label the Hough transform technique that maps sets of k points into the parameter

space $H^k(\Omega)$. An image curve (a point in the parameter space) now gets a vote only if it passes within the error boundary of each point in the set, so we have:

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

where $\binom{\mathcal{E}}{k}$ is the set of all k-subsets of the data points, $\mathcal{E}$.

Consider this function at an arbitrary point in the parameter space. For some set of data points, $\{X_{g_1},\ldots,X_{g_k}\}$, the product, $p(X_{g_1},\Omega)\cdot\ldots\cdot p(X_{g_k},\Omega)$, will be 1 if and only if each of the $p(X,\Omega)$ terms is 1 and otherwise it will be 0. If there are x points such that $p(X,\Omega)$ is 1 (these are the points 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. $H^k(\Omega)$ will thus be $\binom{x}{k}$. Since the standard Hough transform will yield $H(\Omega)=x$ in this case, we can express $H^k(\Omega)$ simply in terms of $H(\Omega)$:

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

If the standard Hough transform uses threshold $t \geq k$ to find peaks and the method of mapping point sets into the parameter space uses threshold $\binom{t}{k}$, these methods will find the same set of peaks according to the above analysis. Their accuracy is thus the same.

3 Decomposition into Subproblems

Let us now introduce a new technique, where we map only those point sets into the parameter space that share some set of j *distinguished points*, $\mathcal{D} = \{X_{d_1},\ldots,X_{d_j}\}$. We will still vary $k-j$ data points, $\mathcal{G} = \{X_{g_1},\ldots,X_{g_{k-j}}\}$, in these sets. The point sets we are mapping into parameter space are thus $\mathcal{D}\cup\mathcal{G}$. This yields:

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

Consider this function at an arbitrary point in the parameter space. Since we aren't varying the distinguished points, $\{X_{d_1},\ldots,X_{d_j}\}$, the curve must pass through the error boundary of each of these to yield a non-zero response. If x points lie on a curve and we use a set of j of distinguished points on the curve, then $x-j$ of these points remain in $\mathcal{E}\setminus\mathcal{D}$. We thus have:

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

We should thus use a threshold of $\binom{t-j}{k-j}$ in this case if the standard Hough transform used a threshold of t. We would then find those curves that are found by the standard Hough transform that pass through the distinguished points

up to the localization error. We can formulate algorithms to recognize arbitrary curves by considering several subproblems, each of which examines a particular set of distinguished points, as above. A deterministic algorithm using these ideas would consider each possible set of distinguished points. This would guarantee that we would examine a correct set of distinguished points for each curve. If we are willing to allow a small probability of failure, we can use randomization to considerably reduce the number of sets of distinguished points that we must examine (see Section 5).

To gain the maximum decomposition of the problem, we want j, the number of distinguished points, to be as large as possible, but note that if we choose $j \geq k$, we will have $H^{\mathcal{D},k}(\Omega) = 0$ or 1 for all Ω. Our response will be 1 if Ω goes through the j points and otherwise 0, but it yields no other information. We thus want to have $j < k$. In addition, we want $k \leq N$ or else we will examine sets that are larger than necessary. The optimal choice is thus $j = k-1 = N-1$.

Note that considering sets of N data points that vary in only one point (i.e. when $j = k-1 = N-1$) constrains the transform to lie on a 1-dimensional manifold (a curve) in the parameter space. This can easily be seen since we have N variables (the curve parameters) and the $N-1$ distinguished points yield $N-1$ equations in them. Let's call this curve the *Hough curve*. When localization error is considered, the transform will no longer be constrained to lie on the Hough curve, but the transform points will remain close to this curve. This yields two useful properties. First, since the Hough curve is essentially 1-dimensional, it is much easier to search than the full parameter space. Second, it is now much easier to propagate localization error carefully. This will be accomplished by determining tight bounds on the range that a set of points can map to in parameter space.

4 Error Propagation

Let's now examine how to propagate the localization error in the curve detection process. We will first consider how error would be propagated in the ideal case. Each set of points maps to a subset of the parameter space under given error conditions. This subset consists of the curves that pass through the set of points up to the error criteria. Call this subset of the parameter space the *error cloud* of the set of points. Ideally, we would determine how many error clouds intersect at each point of the parameter space. This would tell us, for any curve, how many of the points the curve passes through up to the localization error. We do not do this since it is not practical, but for the subproblems we now examine, we can efficiently compute a good approximation.

Since the Hough curve is one-dimensional in the noiseless case, we can parameterize it in a single variable, t. Consider the projection of the error clouds onto the t-axis (see below for examples). The number of such projected error clouds that intersect at some point in this projection 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 vary far from the Hough curve,

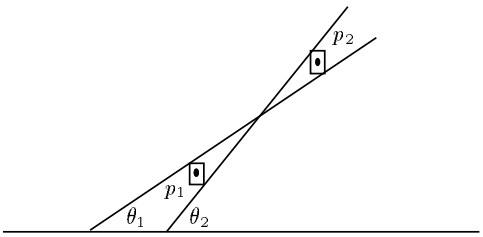

Fig. 1. For any two points, we can determine bounds on the range of θ by considering lines that pass through the boundaries of their possible localization error.

this yields a good approximation to the maximum number of intersecting error clouds, which is the information we want.

Once we have projected each of the sets that we consider in some subproblem onto the t-axis, we can find the peaks along the Hough curve in one of two ways. 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 discretization can be done finely since it is only in one dimension. Alternatively, we could sort the minimal and maximal t points of each error cloud and use a sweep algorithm. This method would examine the extremal points in sorted order and keep a counter that is incremented each time we hit a minimal point and decremented each time we hit an maximal point. If the counter reaches a large enough value, then a line has been found which passes through (or close to) many points.

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 point sets onto the t-axis for each case.

4.1 Lines

If we use the ρ-θ parameterization for lines (i.e. $x\cos\theta + y\sin\theta = \rho$), we can simply parameterize the Hough curve by θ, since ρ is a function of θ. To project the error cloud for a pair of points onto the θ-axis, we simply determine the minimal and maximal θ that a pair of points can yield. If we use square error boundaries, we need only consider the corners of the squares in determining these minimal and maximal θ values. See Fig. 1.

4.2 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 will use $j = N - 1 = 2$ distinguished points. We can parameterize the Hough curve by the distance of the center of the circle from the midpoint of the two distinguished points (which we will take to be positive when the center is to the right of the segment connecting the distinguished points, and negative otherwise).

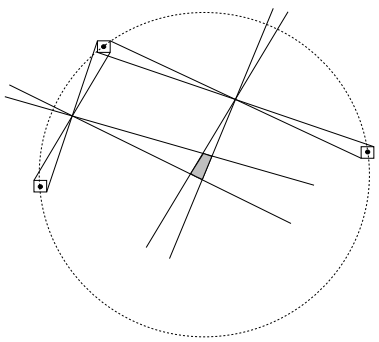

Fig. 2. We can determine bounds on the position of the center of a circle passing through 3 points (up to localization error) by examining the range of possible perpendicular bisectors for the segments between the points.

To project the error cloud onto the t-axis, we now want 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 in which two of the perpendicular bisectors of the segments can meet (see Fig. 2). The minimum and maximum distance from the center of the circle to the midpoint of the distinguished points can easily be determined by examining the extremal points of this set.

5 Computational Complexity

This section determines the computational complexity of the techniques described in this paper. Let's first determine how many of the sets of distinguished points we must examine to maintain a low rate of failure. We'll assume that we only need to find curves that comprise some fraction, ϵ, of the total number of data points. The probability that a single set of j random points lie on a particular such curve is then at least:

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

since we must have $\binom{\epsilon n}{j}$ sets of distinguished points that lie on the curve among the $\binom{n}{j}$ possible sets of distinguished points. If we take t such trials, the probability that all of them will fail for a particular curve is no more than:

$$p \leq (1 - p_0)^t \approx \left(1 - \epsilon^j\right)^t$$

For each curve, we thus have a probability no larger than p that we will fail to examine a set of distinguished points 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 points will lead to the identification of the curve.

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

$$\left(1-\epsilon^j\right)^t \leq \delta$$

Solving for t yields:

$$t \geq \frac{\ln \delta}{\ln(1-\epsilon^j)} \approx \frac{\ln \frac{1}{\delta}}{\epsilon^j}$$

For each trial, we now have to find the peaks on the Hough curve. Recall that we use $j = N-1$ in our method to facilitate the propagation of error. If we use voting, the time is dependent on how finely the Hough curve is discretized. If there are α bins, we need to increment $O(\alpha)$ bins per trial per point, yielding $O(n\alpha)$ time per trial. The total time requirement is thus $O(\frac{n\alpha \log \delta}{\epsilon^{N-1}})$ or simply $O(n)$ when measured by the size of the input (α, δ, ϵ, and N are constants).

If we use the sweep algorithm, we must sort the $O(n)$ maximal and minimal points of the error clouds, requiring $O(n \log n)$ time per trial. Processing the sorted points requires $O(n)$ time. We thus require $O(\frac{n \log n \log \delta}{\epsilon^{N-1}})$ total time or $O(n \log n)$ when measured by the size of the input.

6 Results

These techniques have been applied to real images to test their efficacy. Figure 3 shows an image that was used to test the line detection techniques. The edges were determined to sub-pixel accuracy. For tests on this image, square error boundaries were used such that the true location of each point was assumed to be within 0.25 pixels of the measured location in both x and y. 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 4 shows an image that was used to test the circle detection techniques. This image is an engineering drawing that has been scanned. For this reason, it was not possible to determine the location of pixels to sub-pixel accuracy. In addition, the presence of small and dashed circles and the clutter in the image make this a difficult test case. While all of the large circles were found with $\epsilon = 0.04$, the small and dashed circles did not comprise a large enough fraction of the image to be found. With $\epsilon = .008$, the implementation finds several of circles, some of which are not perceptually obvious. Note that in each of the insalient circles, the pixels found overlay most of the perimeter of a circle and thus if we want to find small and/or broken circles it is difficult to rule out these circles without using additional information. In addition, the implementation has difficulty finding both of the dashed circles with the same center since they are so close together and are imperfect circles. The dashed circles shown in Fig. 4(c) consist of the top half of one of the circles and the bottom half of the other.

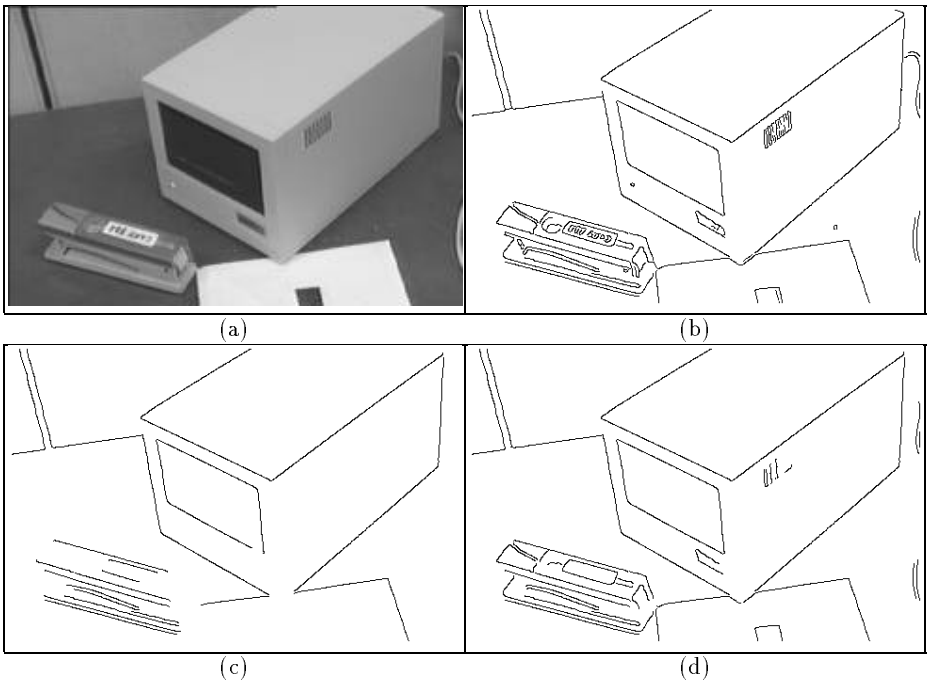

Fig. 3. A test image for line detection. (a) The original image. (b) The edges detected. (c) The lines found with $\epsilon = .01$. (d) The lines found with $\epsilon = .004$.

7 Ellipses and Other High-Order Curves

When applying these techniques to curves with several degrees of freedom, we must take special care, since the number of trials that are required can become large. Let's consider the detection 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 pixels (or some subset after segmentation), then the number of trials required to achieve 99% accuracy is 74. On the other hand, if we wish to detect ellipses that comprise at least 10% of the image pixels using these techniques in a straightforward manner, then this would require 46,052 trials to achieve 99% accuracy.

When we wish to detect high-order curves in complex images there are additional techniques that we can use in order to perform curve detection quickly. One simple technique is to use more information at each curve pixel. For example, we can use the orientation of the curve at each pixel (as determined from the gradient, the curve normal, or the tangent). When we do this, we require fewer curve points to determine the position of the curve. We can determine the

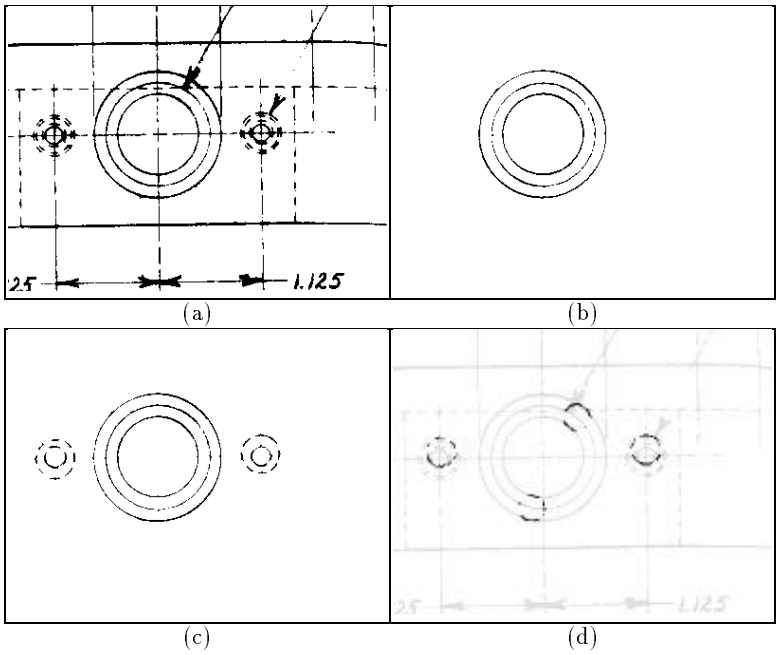

Fig. 4. A test image for circle detection. (a) The original engineering drawing. (b) The circles found with $\epsilon = 0.04$. (c) Perceptually salient circles found with $\epsilon = 0.008$. (d) Insalient circles found with $\epsilon = 0.008$.

position of an ellipse using three points with orientations rather than five unoriented points. We would thus use two, rather than four, distinguished points, and we would require many fewer trials to ensure that there is a low probability of not selecting a correct set of distinguished points. Of course, we do not need to restrict this technique to high-order curves. We can use two oriented points to determine the position of a circle, rather than the three unoriented points used in the previous sections.

An alternate technique that can detect high-order curves quickly is to use a two-step technique, where we first determine a subset of the curve parameters and then determine the remaining parameters. For examine, Yuen *et al.* [8] describe a method for detecting ellipse centers. They note that the center of an ellipse must lie on the line connecting the intersection of the tangents of two points on the ellipse with the midpoint of the segment between the two points. A point of intersection between several such lines yields a likely ellipse center. We can use decomposition techniques similar to those already described in this paper when this method is used to detect ellipse centers. Once the center of the ellipse has been detected, there are three remaining ellipse parameters. These can be detected using a three parameter Hough transform similar to the detection

of circles.

8 Summary

We have considered efficient techniques to perform the Hough transform with careful propagation of localization error. To this end, we have modified a formal definition of the Hough transform to allow localization error to be analyzed appropriately. We then considered a new method where the Hough transform is decomposed into several subproblems, each of which examines a subset of the parameter space, by considering only those point sets that include some set of distinguished points. These subproblems allow us, first, to propagate the localization error efficiently and accurately in the parameter space, and second, to use randomization techniques to reduce the complexity while maintaining a low probability of missing an important curve. The overall complexity of the resulting algorithm is $O(n)$, where n is the number of data points in the image. Finally, we have given results of this system detecting straight lines and circles in cluttered and noisy real images and discussed the application of these techniques to curves with several parameters.

References

1. J. R. Bergen and H. Shvaytser. A probabilistic algorithm for computing Hough transforms. *Journal of Algorithms*, 12:639–656, 1991.
2. A. Califano, R. M. Bolle, and R. W. Taylor. Generalized neighborhoods: A new approach to complex parameter feature extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 192–199, 1989.
3. J. Illingworth and J. Kittler. A survey of the Hough transform. *Computer Vision, Graphics, and Image Processing*, 44:87–116, 1988.
4. 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, November 1992.
5. V. F. Leavers. Which Hough transform? *CVGIP: Image Understanding*, 58(2):250–264, September 1993.
6. J. Princen, J. Illingworth, and J. Kittler. A formal definition of the Hough transform: Properties and relationships. *Journal of Mathematical Imaging and Vision*, 1:153–168, 1992.
7. L. Xu, E. Oja, and P. Kultanen. A new curve detection method: Randomized Hough transform (RHT). *Pattern Recognition Letters*, 11:331–338, May 1990.
8. H. K. Yuen, J. Illingworth, and J. Kittler. Detecting partially occluded ellipses using the Hough transform. *Image and Vision Computing*, 7(1):31–37, 1989.

This article was processed using the LaTeX macro package with ECCV'96 style