FAST LINE AND CIRCLE DETECTION USING INVERTED GRADIENT HASH MAPS

R.Gonzalez
School of ICT, Gold Coast Campus, Griffith University

ABSTRACT

This paper presents fast algorithms for line and circle detection based on inverted gradient hash maps (IGHM). Inverted indices are a common technique for storing a map from content of a dataset to its locations in the dataset. Hash maps are typically used to implement associative arrays and reduce search times in large datasets. In this paper, a hash map is used to store an inverted index of image gradient magnitudes and orientations. Algorithms for detecting lines, and circles using IGHMs are presented and shown to be competitive against existing approaches.

Index Terms— *Line Detection, Circle Detection*

1. INTRODUCTION

The detection of lines and circles is a fundamental step within many image-processing applications. Many well-established algorithms exist for these tasks operating on image gradients in the image's intrinsic Cartesian space or in an alternate parameterized 'Hough' space [1]. As Hough spaces are not always the most computationally efficient for many image-processing tasks many "fast" algorithms operating in these spaces have been proposed that trade speed for robustness or generality [2]. In the image processing literature, inverted indices [3] have only been used for reducing retrieval time in image databases [4, 5]. Yet the concept of an inverted gradient space representation presents a number of attractive features for algorithms that exploit image gradients. It automatically clusters pixels with similar gradients together, virtually eliminating the need for search the image space. This both simplifies gradient-based algorithms and makes them faster.

In Cartesian space line and circle detection can be performed using edge chaining [6, 7, 8]. For each candidate pixel the local neighbourhood is searched to determine which pixels to accept or reject according to a set of criteria. Implementations typically have complexity orders higher than of $O(N^2)$ Gradient prediction can be used to narrow the search area and reduce processing time. Another approach segments the image based on gradient orientation into line-support regions [9]. Lines are then detected by analyzing attributes extracted from the support regions and a planar fit to the underlying intensity surface.

In contrast to searching, Hough approaches rely on each candidate pixel voting using a one-to-many projection in the accumulator space. In the simplest formulation each edge pixel in {x,y} space becomes a line in the slope-intercept parameter space. In more common formulations each pixel becomes a sine curve in a polar parameter space. Hough line detection has complexity in the order of $O(KN)$. The Hough Circle Transform (HCT) has a complexity in the order of $O(\pi N(K^2+K))$. Probabilistic or randomized schemes [10, 11, [12, 13] exist for reducing the computation time that select only a subset of edge points to use in voting. Using edge orientation for this optimization was suggested by Kimme et al. [14] and is used by many fast circle and ellipse detectors.

In both the Cartesian and Hough space approaches, gradient orientation has been used as auxiliary information to reduce amount of computation. In contrast, this paper, like that of [9] considers the orientation as the starting point. However unlike the approach in [9] that operates exclusively in Cartesian space, the proposed line detection approach exclusively operates in gradient space. The proposed circle detection method also predominantly operates in gradient space. Using an inverted gradient space for line and circle detection reduces computational complexity by avoiding the need for one to many voting or searching as is required in Hough and/or Cartesian spaces.

In the following sections the inverted gradient space representation using hash maps is first described. Then algorithms for line and circle detection using IGHMs are presented. Experimental results for the performance of these algorithms relative to existing methods are lastly presented.

2. METHOD DESCRIPTION

A typical image $\boldsymbol{f}$ is a mapping of coordinates $[x,y] \in \mathbf{N}^2$ to pixel intensity values i:

$$\boldsymbol{f} : (x, y) \rightarrow i \tag{1}$$

An image gradient is defined by the magnitude and orientation $\{m, \theta\}$ of the derivate of the intensity at (x, y):

$$m_{x,y} = \sqrt{\left(\frac{\partial}{\partial x} i_{x,y}\right)^2 + \left(\frac{\partial}{\partial y} i_{x,y}\right)^2} \qquad \theta_{x,y} = \arctan\left(\frac{\partial}{\partial x} i_{x,y}, \frac{\partial}{\partial y} i_{x,y}\right) \tag{2}$$

A gradient image $\boldsymbol{g}$ is a mapping from Euclidean coordinates to the local image gradients:

$$\boldsymbol{g} : (x, y) \rightarrow \{m, \theta\} \tag{3}$$

An inverted gradient image $\boldsymbol{h}$ is a mapping from image gradients to coordinates.

$$\boldsymbol{h} : \{m, \theta\} \rightarrow (x, y) \tag{4}$$

In practice, the mapping produced by $\boldsymbol{g}$ is a surjective function. Accordingly the inverted gradient image cannot be efficiently stored as a two dimensional array of coordinates, as each image gradient may map to multiple coordinates. This can be effectively resolved by representing the inverted gradient image as a hash map where collisions in $\{m, \theta\}$ are resolved via chaining. In this case the inverse gradient image becomes a two dimensional array of linked lists. Each list stores coordinates of those pixels in the image having a given gradient magnitude and orientation as in Figure 1. Any image gradients of a given orientation and magnitude can be located by indexing the hash map by those values.

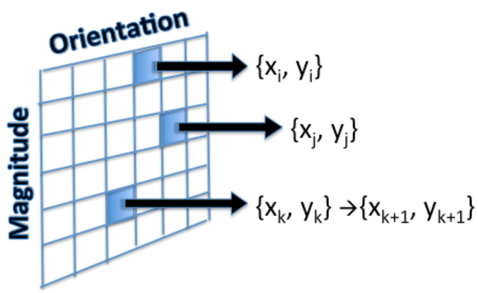

Figure 1. IGHM Data Structure

The performance of typical hash maps is heavily influenced by the load factor. This roughly estimates the average chain length for each list. In the IGHM the load factor will be a function of the angular and scalar resolution used for the values (m, θ) relative to the image size in pixels. In general-purpose hash maps, high load factors are detrimental to access times as lookup is replaced by searching as the primary access method. This is not a concern with inverse gradient hash maps since there is no need to search the lists to find relevant items since all the entries in each list share the same gradient properties within the range defined by the resolution of (m, θ).

2.1 Line Detection

Since all the pixels contributing to a line of any given slope are stored within a single column in the IGHM, lines of any given orientation can be easily found. However, detecting lines is not as simple as adding the contribution of all the pixels in one column since they may belong to lines with different y-offsets. Instead it is necessary to calculate the offsets so that pixel contributions can be apportioned to the correct potential lines. The approach here taken is to calculate these offsets in polar coordinates. In this approach lines are described in terms of their orientation θ and the shortest perpendicular distance R from the infinite extension of the line to the origin as is shown in Figure 2.

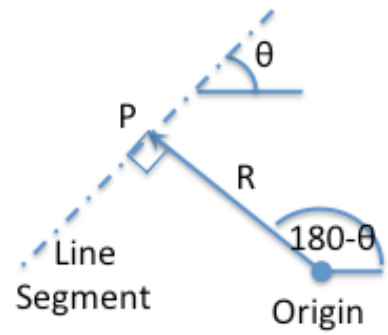

Figure 2. Line Offset Geometry

The equation for a line of slope m passing through a point $p_i = (x_i, y_i)$ where $m = \tan(\theta)$ is given by:

$$y - y_i = \left(x - x_i\right) \cdot m \tag{5}$$

The perpendicular distance R_i from this line to the origin (x_0, y_0) is given by equation (6):

$$R_i = \frac{\left|-x_0 m + y_0 - y_i + x_i m\right|}{\sqrt{m^2 + 1}} \tag{6}$$

All collinear points stored in any given orientation column in the IGHM will share the same distance to the origin. Accordingly a histogram of distances r can be calculated from all points p_i having a given orientation θ by summing up their gradient magnitudes m_i. Considering the pixels for each gradient orientation gives a two-dimensional histogram $H(\theta, r)$:

$$H\left(\theta, r\right) = \sum_{i=1}^{M} m_i \forall \left\{p_i \middle| R_i = r\right\} \tag{7}$$

Lines can then be simply detected in $H(\theta, r)$ as the maxima, representing the total support energy. A variety of parameters can be calculated "on the fly" to further discriminate between lines such as their length given by coordinate range of contributing pixels, the average line intensity, and the line density as the line length divided by the number of contributing pixels. As only a single vote is cast per image gradient, this is much faster than the Hough voting scheme that requires many votes for each pixel, having complexity in the order of $O(N)$. It is also much faster than methods such as that presented in [9] that need to find line support regions within the image.

2.2 Circle Detection

For circle detection, as with other existing methods, the basic approach followed is to only consider the contribution of pairs of pixels having appropriate gradients and magnitudes. The IGHM facilitates this task by eliminating the need to search for them in the image. The orientation of image gradients on opposing sides of a circle's center will have opposing directions. So circle detection can be limited to considering pairs of points on the circumference that differ in orientations by 180 degrees. However, in contrast to the simplistic method reported in [15], it is important to also consider the distribution of support energy to minimize false detections. Ideally the energy should be equally distributed around the circumference of a circle. This implies that point pairs should be further limited to those with roughly similar gradient magnitudes.

In addition a means of assessing the radial energy distribution is required. This proceeds by finding the mid point of an imagery line connecting each pair of points $\{p_i, p_j\}$ being considered. The slope of each such line, gives the octant of the circle within which it falls. For each mid point the average gradient magnitude α of the contributing points is stored for the relevant octant. However, for all possible pairs of candidate pixels that share that same point only the maximum gradient magnitude is retained. This results in a table of magnitudes W indexed by the coordinates of the mid point at (x, y) and the octant ϕ. The value of t_m determines the maximum difference between gradient magnitudes m_i and m_j while the value of t_a determines the maximum angular deviation away from 180 of orientations θ_i and θ_j :

$$\phi = \frac{1}{23}\arctan\left(\frac{\theta_i}{\theta_j}\right) \qquad \alpha = \frac{m_i + m_j}{2}$$
$$\kappa = \left|\theta_i - \theta_j - 180\right| \qquad \mu = \left|m_i - m_j\right| \tag{8}$$
$$W_{x,y,\phi} = \max\left(\left\{\alpha \forall p_i, p_j \left| \kappa \le t_a \wedge \mu \le t_m \right.\right\}\right)$$

Additionally a histogram of the Euclidean distances d between all pairs of points considered is stored for each mid point at (x,y) by summing their gradient magnitudes:

$$E_{x,y,d} = \sum \alpha \forall p_i, p_j \left| \kappa \le t_a \wedge \mu \le t_m \right.$$
$$\mathit{where} \qquad d = \left\| p_i - p_j \right\| \tag{9}$$

After all the relevant data have been collected from the IGHM into a suitable intermediate data structure the circles can be identified. For each potential circle centre the octant ϕ having the greatest magnitude is found from the magnitude table W. A measure of how well the support energy is radially distributed around the potential circle σ is then determined using equation (10) :

$$\sigma_{x,y} = \sum_{\theta=0}^{360} w_{x,y,\theta} \cdot \sin^2\left(\theta - \bar{w}\right)$$
$$\bar{w} = \max(\left\{w_{x,y,\theta} \forall \phi\right\}) \tag{10}$$

While this measure of radial energy distribution can be used on its own to successfully detect circles, it can suffer from false detections. This can be corrected by weighting σ according to the energy of the most likely circle diameter centered at that point using the information from the distance histograms E. The most likely diameter is taken as the one with the highest energy. Accordingly the measure used to detect circles becomes:

$$\mathrm{K}_{x,y} = \sigma_{x,y}^2 \cdot \max\left(\left\{E_{x,y,d} \forall d\right\}\right) \tag{11}$$

The circle centre(s) can now be detected as the maxima in K and the diameter of each circle is simply the index in the distance table E with the highest value for that centre.

3. EXPERIMENTAL RESULTS

The standard Hough transform (SHT) has been shown to be more accurate than a range of probabilistic and other non-probabilistic variants [16]. Accordingly it was used to assess the relative effectiveness of IGHM based line and circle detection. The Sobel operator was applied to the original image before IGHM and SHT line detection. An 11x11 surround suppression kernel was used with the SHT and a 6x18 suppression kernel (m, θ) for the IGHM.

Figure 3 shows a simple binary image of predominantly rounded shapes superimposed with the 18 most likely detected lines. On the left side the lines are those detected via IGHM and on the right side are those detected via the SHT. Apart from the four sides of the small square there is roughly equal support for lines of all other orientations as there are no other straight edges in the image. As can be observed, the SHT has a tendency to pool most of its maxima in a few areas and did not even detect the edges of the square while the IGHM's selection of lines is closer to what might be expected from this image.

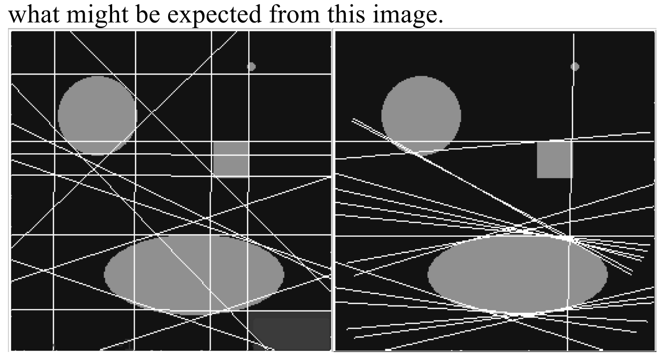

Figure 3. Lines Detected; IGHM on left, HT on right

The line detectors with the same configuration were run on a number of well-known natural images of various sizes. The results are shown in figure 4, with the IGHM lines superimposed on the left side images and the HT lines superimposed on the right side image. The best 16 represented lines are shown for each case. The rows in order from the top are the 256x256 cameraman image, 512x512 clown and bridge images respectively. These results show that the IGHM is less susceptible to false detection.

Similarly circle detection was evaluated using IGHM and HCT for natural images. For the IGHM method, the values of t_a = 15 and t_m = 3 were used. These results, with the detected circles superimposed on the images are shown in figure 5. From the left the images show the original, the IGHM circles and the HCT circles on the right side. The IGHM detector performs well even in those cases where the HCT fails to detect the circles present in the images.

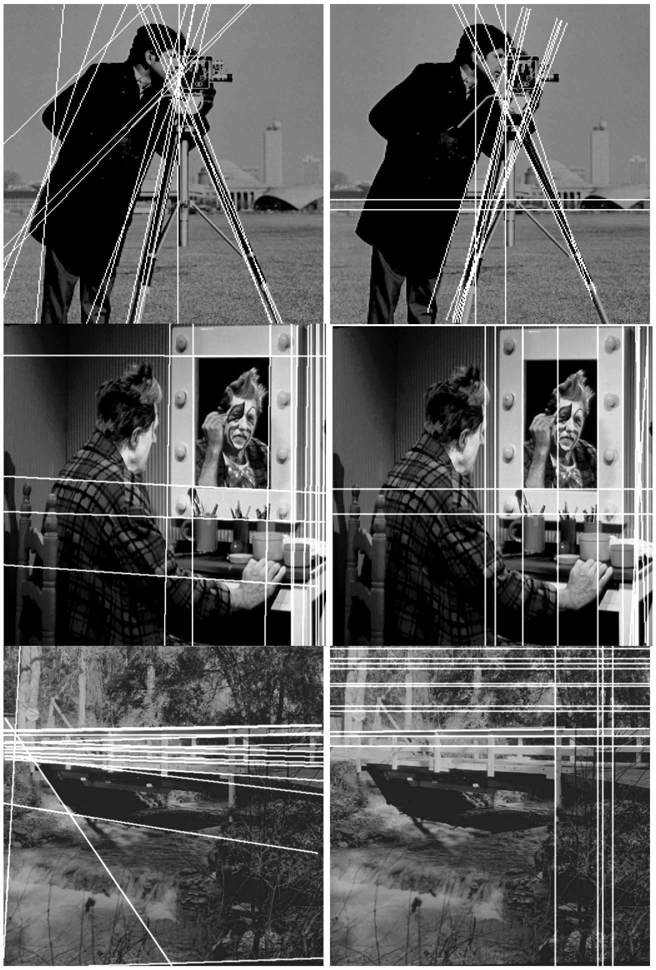

Figure 4. Lines Detected; IGHM on left, HT on right

The computation time for detecting the lines in Figure 4 was measured on a MacBookPro with a 2.6GHz Intel i7 with 16 GB of memory. The results in Table 1 show the execution times for the HT and the IGHM including the Sobel gradient calculations but not surround suppression and peak picking. The IGHM is notably faster than that HT. The computation times for either can be reduced using probabilistic techniques.

Table 1 Line Detection Speed

	IGHM	HT
Bridge	79.9 ms	1520 ms
Clown	56.2 ms	673 ms
Cameraman	15.3 ms	95 ms

Similarly the computation times for circle detection were measured. The images used were the ovals image used in figure 3, and images of Lena and a clown. Table 2 shows the times for both IGHM and the HCT. Suppression is not required for IGHM but a 5x5 kernel was used for the HT. The IGHM is approximately two orders of magnitude faster than the HCT. For comparison, the speed of a state of the art randomized circle detection (RCD) reported in [17] for 225x225 and 430x440 sized images (30% smaller) is included. Also, Zhang's real-time detector [18] requires 120ms for ellipse detection in a simple 320x240 image.

Table 2 Circle Detection Computation Time

	IGHM	HCT	RCD
Clown 512x512	3,065 ms	63,343 ms	4,000 ms*
Lena 256x256	196 ms	4,044 ms	
Ovals 256x256	10 ms	3,226 ms	1,910 ms*

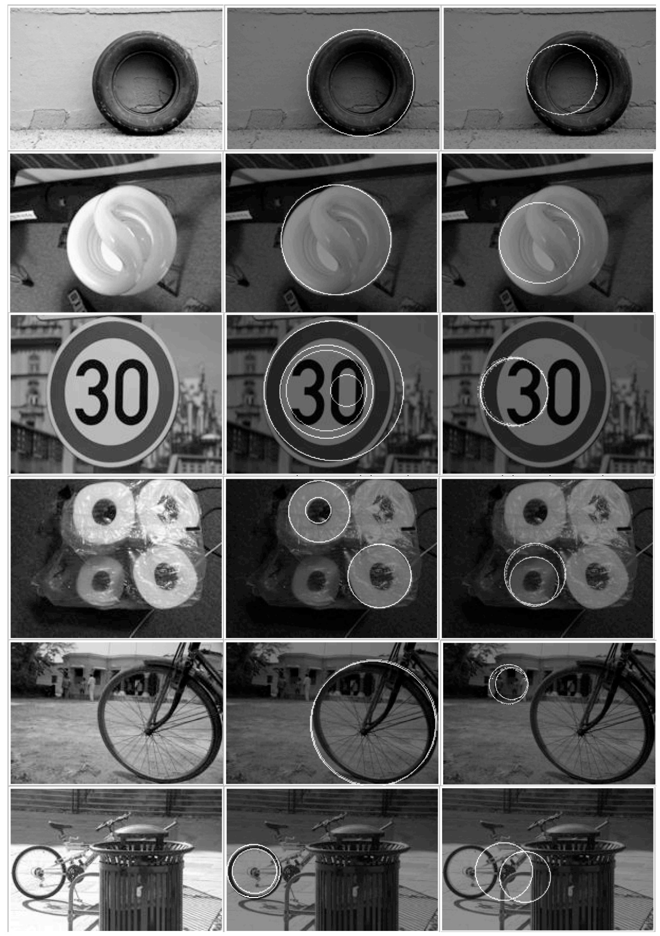

Figure 5. Circle Detection; HCT on right

4. CONCLUSIONS AND FURTHER RESEARCH

While many algorithms exist for line and circle detection they all require either one-to-many voting and/or searching in a Cartesian or parameter space, which is slow. Many probabilistic approaches exist for reducing the amount of pixels processed, and computation time. In contrast IGHM based methods eliminate the need for searching and/or one-to-many voting and are one to two orders of magnitude faster than the standard Hough methods and yield higher detection accuracy. While they compare favorably to state of the art probabilistic methods in terms of speed, the computational requirements of IGHM methods can be further reduced using probabilistic approaches.

12. REFERENCES

[1] R.Duda, and P.Hart, “Use of the Hough Transform to Detect Lines and Curves in Pictures”, *Communications of the ACM* 15, pp: 11–15, 1975

[2] H. Kälviäinen, P. Hirvonen, L. Xu, E. Oja, “Probabilistic and non-probabilistic Hough transforms: overview and comparisons”, *Image and Vision Computing*, Vol. 13, Issue 4, May 1995, pp.239-252

[3] J. Zobel and A. Moffat, “Inverted files for text search engines,” *ACM Computing Surveys*, vol. 38, no. 2, p. 6, July 2006

[4] D. M. Chen, S. S. Tsai, V. Chandrasekhar, G. Takacs, R. Vedantham, R. Grzeszczuk, and B. Girod.. “Inverted Index Compression for Scalable Image Matching.” In Proceedings of the 2010 Data Compression Conference (DCC '10). Washington DC, USA, IEEE pp.525-

[5] J. Mamou, Y. Mass, M. Shmueli-Scheuer, B. Sznajder, “A Unified Inverted Index for an Efficient Image and Text Retrieval,” In Proc. of the 32nd international ACM SIGIR conference on Research and development in information retrieval (SIGIR '09) NY, USA, pp.814-815.

[6] V. Pătrăucean, P. Gurdjos, R. G. von Gioi. “A parameterless line segment and elliptical arc detector with enhanced ellipse fitting.” In Proceedings of the 12th European conference on Computer Vision - Volume Part II (ECCV'12), Springer-Verlag, Berlin, Heidelberg, 2012

[7] Lee, Y.-S., Koo, H.-S., Jeong, C.-S. “A straight line detection using principal component analysis.” *Pattern Recognit. Lett.* **27(14),** 1744–1754 (2006)

[8] R.G. von Gioi,· J. Jakubowicz, J.M. Morel, G. Randall, “On Straight Line Segment Detection.” *J. Math. Imaging Vis*. 32, 3 (November 2008), 313-347

[9] J. B. Burns, A.R. Hanson, E.M. Riseman, “Extracting Straight Lines”, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 8, no. 4, pp. 425-455, 1986

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

[11] Matas, J., Galambos, C., Kittler, J., “Robust detection of lines using the progressive probabilistic Hough transform.” *Computer Vision and Image Understanding* 78 (1), 2000, pp.119–137.

[12] T. C. Chen and K. L. Chung, “An efficient randomized algorithm for detecting circles,” *Computer Vision and Image Understanding*, vol. 83, no. 2, pp. 172–191, 2001.

13 Lei Xu, Erkki Oja, Pekka Kultanen, “A new curve detection method: Randomized Hough transform (RHT)”, *Pattern Recognition Letters*, Volume 11, Issue 5, May 1990, Pages 331-338, ISSN 0167-8655,

[14] Kimme, C., Ballard, D., and Sklansky, J., “Finding circles by an array of accumulators”, *Proc. ACM* 18, pp: 120–122, 1975

[15] A.A. Rad, K. Faez, N. Qaragozlou, “Fast Circle Detection Using Gradient Pair Vectors” Proc. VIIth Digital Image Computing: Techniques and Applications, 10-12 Dec. 2003, Sydney

[16] H. Kälviäinen, P. Hirvonen, L. Xu, E. Oja, “Probabilistic and non-probabilistic Hough transforms: overview and comparisons”, *Image and Vision Computing*, Vol. 13, Issue 4, May 1995, Pages 239-252

[17] L. Pan, W-S. Chu, J. M. Saragi, F. De la Torre, M. Xie.: "Fast and Robust Circular Object Detection With Probabilistic Pairwise Voting," *Signal Processing Letters, IEEE* , vol.18, no.11, pp.639,642, Nov. 2011

[18] S.C. Zhang, Z.Q.Liu, “A robust, real-time ellipse detector” *Pattern Recognition* 38 (2005), pp.273-287