Improved Line Detection based-on Pixel Coordinates: A New Approach to Line Detection

Idris El-Feghi
Faculty of Information Technology
University of Misurata
Misurata - Libya
Email: idrise [AT] ee.edu.ly

Kahina Zitouni
Electrical Engineering Department
University of Tripoli
Tripoli - Libya

***Abstract*—Determining the location and orientation of straight lines in images is a problem of great importance in the field of image processing. The Hough transform, has been widely used to solve this problem for binary images, and is still by far dominating the scene when it comes to line detection and its concept was further developed to detect other shapes like circles.**

In this paper, a new method for straight line detection is proposed. The formulation of this new method is based on the use of the coordinates of pixels in the digital image, which relates the parameters determining the location and orientation of the lines in the image. The advantage of this method is that it provides an approach to the problem of line detection within a framework that enhances the detection of lines by incorporation of prior information of the images nature. This approach proves useful and efficient in computational schemes where it is needed to detect lines in 0, 45, 90, and 135 degree directions, and results in optimized performance. Experimental results of this new approach shows how the image coordinates method can be viewed as an alternate optimal method for noisy images and dotted lines in terms of detection accuracy and accumulator size reduction.

Keywords—line detection; Hough Transform; digital image; processing; line location; line direction.

I. INTRODUCTION

An image may be defined as a two-dimensional function, *f(x, y)*, where *x* and *y* are plane coordinates, and the amplitude at any pair of coordinates *(x, y)* is called the Intensity or gray level of the image at this specific point. When *x*, *y*, and the amplitude values of f are all finite, discrete quantities, the image is called a digital image [1].

"Digital images are composed of *pixels* (short for picture elements). Each pixel represents the color (or gray level for black and white photos) at a single point in the image, so a pixel is like a tiny dot of a particular color. By measuring the color of an image at a large number of points, it is possible to create a digital approximation of the image from which a copy of the original can be reconstructed. Pixels are a little like grain particles in a conventional photographic image, but arranged in a regular pattern of rows and columns and store information somewhat differently. A digital image is a rectangular array of pixels sometimes called a *bitmap.*" [2].

In analysis of digital images, a sub-problem often arises of detecting simple shapes, such as straight lines, circles or ellipses. In many cases an edge detector can be used as a preprocessing stage to obtain image points or image pixels that are on the desired curve in the image space. Due to imperfections in either the image data or the edge detector, however, there may be missing points or pixels on the desired curves as well as spatial deviations between the ideal line/circle/ellipse and the noisy edge points as they are obtained from the edge detector. For these reasons, it is often non-trivial to group the extracted edge features to an appropriate set of lines, circles or ellipses [3].

The purpose of line detectors is to address this problem by making it possible to perform groupings of edge points into detected object candidates (lines for example) by performing an explicit voting procedure over a set of parameterized image objects [3].

II. HOUGH TRANSFORM

The Hough Transform is an algorithm presented by Paul Hough in 1962 for the detection of features of a particular shape like lines or circles in digitalized images. Although Hough Transform is a standard algorithm for line or circle detection and has broad applications today (e.g. it is used in traffic observation systems, land monitoring and robotics), it has weak points, especially its computational complexity [4].

A. Principles of Line Detection Using Hough Transform

The simplest case of Hough transform is the linear transform for detecting straight lines. A straight line can be described as $y = mx + b$ and can be graphically plotted for each pair of image points (x, y). In the Hough transform, the main idea is to consider the characteristics of the straight line not as image points (x_1, y_1), (x_2, y_2) but instead, in terms of its parameters, i.e., the slope parameter m and the intercept

parameter *b*. Based on this, the straight line $y = mx + b$ can be represented as a point (*b*, *m*) in the parameter space [5].

However, the vertical lines give rise to unbounded values of the parameters *m* and *b* which is a problem, so, for computational reasons, it is therefore better to use a different pair of parameters, denoted *r* and θ (*theta*), for the lines in the Hough transform. These are the Polar Coordinates [5].

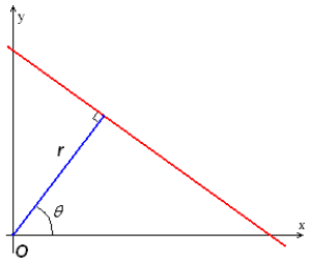

Figure1 - the (*r*, θ) representation of a point in Hough Transform

The parameter "*r*" represents the distance between the line and the origin, while θ is the angle of the vector from the origin to this closest point. Using this parameterization, the equation of the line can be written as indicated in the equation 1 [5].

$$r(\theta) = x_0 \cos\theta + y_0 \sin\theta \quad (1)$$

B. *Implementation of Hough Transform*

The most important aspect in the Hough Transformation is that every line is represented in a polar coordinate system or in what is called "Hough Space"; hence, every possible line can be represented by a unique *r* and θ. And further, every pixel on a given line will transform to the exact same *r* and θ for that line. The Hough transform algorithm uses an array, called an "Accumulator", to detect the existence of a line. An Accumulator is nothing more than a 2 dimensional matrix with θmax columns and *rmax* rows. So each cell, represents a unique coordinate (r, θ) and thus one unique line in Hough Space. Figure 2 illustrates how the points on a straight line are stored in the Accumulator [6].

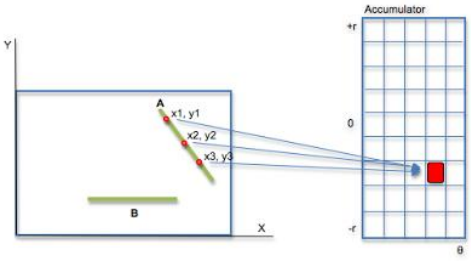

Figure2 – Illustration of the Hough Transform Representation

The Accumulator in the Hough Transform Algorithm initially holds all *0s* values, then at every point in the image the value of *r* is calculated for $\theta = 1$ *to* 180. Finally, the Accumulator is incremented by *1* for every value of *r* found. Once done, the Accumulator contains a count of pixels for every possible line. Only the values that are above the threshold are kept (for example, it could be decided that a line consists of a minimum pixel count of 5) [6].

The result is a set of coordinates (r, θ), one for every straight line in the image that is above the threshold. And these polar coordinates can easily be transformed back to the (x,y) coordinates by solving the equation for *x* and *y* as indicated in the equations 2 and 3.

$$x_0 = (r - y_0 \sin\theta)/\cos\theta \quad (2)$$

$$y_0 = (r - x_0 \cos\theta)/\sin\theta \quad (3)$$

Since the Hugh Transform is a kind of Brute-Force method it is very complex in computation. It has two main drawbacks: large memory requirement and slowness when high accuracy is required. In order to find the plane parameters accurately, parameter space must be divided finely in all three directions, and an accumulator assigned to each block [7]. Also it takes a long time to fill the accumulators when there are so many. The Fast Hough Transform (FHT) gives considerable speed up and reduces memory requirement [4]. Instead of dividing parameter space uniformly into blocks, the FHT homes in on the solution, ignoring areas in parameter space relatively devoid of votes. The relative speed advantage of the FHT increases for higher dimensional parameter spaces. Another weakness of the Hugh Transform is that it often recognizes many similar lines instead of the correct one. The algorithm only returns a line without a starting and ending point. For that different post processing algorithms have to be used [8]. Hence, arises the need for a new method, simple, and accurate.

III. Line Detection Using Pixel Coordinates

Despite the robustness and effectiveness of the Hough Transform, there is a need for a simpler method that does not include complex calculations, large storage, and long running time [9] .The proposed new method delivers results in a much simpler manner while delivering accurate results in a scenario where the lines to be detected are known to be in one of 4 predetermined positions: 0, 45, 90, and 135 degree angles.

Since digital images are pixel arrays, the position of every pixel in the image is known and fixed and since each straight line is a set of pixels lying in straight extension, so, each straight line can be referred to by the position of the fixed pixel rather than slope and intercept, as in Hough, which eliminates the complexity of calculating the intercept. By avoiding the calculation of line intercept, there will be no

need to round the number resulting from the intercept calculation and hence the accuracy improves.

By using the discrete representation of digital images to refer to the intercept of any straight line it is possible to express its distance from the point of origin by the number of pixels between the line and the origin. And since the row or column in which the line falls is known, it is possible to build an accumulator array consisting of positions of the straight lines, and this will reduce the size of accumulator considerably compared to Hough Transform. Hence, this fixes the problem of memory size of the Hough Transform.

A. Principles of the Pixel Coordinates Method

Bearing in mind that every straight line (vertical, horizontal, or diagonal) in a digital image can be expressed by its coordinates, and the coordinates are already read in edge detection process, so, by utilizing the information resulting from the edge detection, there will be no need to perform new calculations further more. Building on this, to detect the lines, the image is mapped to flags corresponding to vertical, horizontal, or diagonal straight lines that occurred in the image. Using this principle, there will be no voting mistakes. Each straight line will be described by an index calculated using the coordinates of any point on it.

Also, to reduce the size, the indexes of the flags will be the same as the indexes of the accumulator and when applying the detected straight lines, instead of comparing each edge pixel with all detected straight lines parameters to detect if it lying to any one of them, the accumulator array is mapped to the line occurrence indicators.

B. Implementation of the Pixel Coordinates Method

Because of the discrete nature of digital images, the coordinates are in matrix format and lines cannot fall between two pixels i.e. the line slope angle must be 0, 45, 90 or 135 degrees or a line consisting segments of lines with these four slopes. Hence, finding possible straight lines is done by checking only in four line slope angles (0, 45, 90 or 135 degrees) in each edge pixel in image. And to facilitate the referencing process, the lines in the four specific directions will be referred to as indicated in the Table 1 as following:

Line angle	***Reference Name***
0	*Horizontal*
45	*Right diagonal*
90	*Vertical*
135	*Left diagonal*

Table 1 – Line Referencing

A unique identifier, "*index*", is developed to each straight line falling in any of the four directions of straight lines; indexes can be determined by the following equations:

- Vertical: $index = x$
- Horizontal: $index = y$
- Right diagonal: $index = (w - 1) + (h - 1) - x - y$
- Left diagonal: $index = x - y + (h - 1)$

Figure 3 provides a numerical illustration of the calculation of indexes based on the above equations:

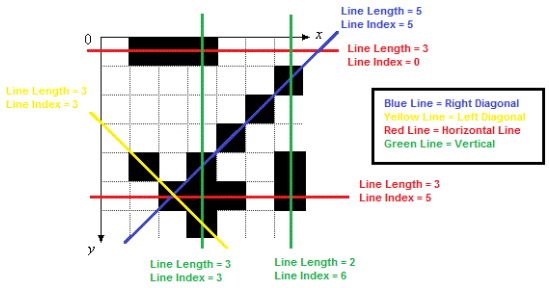

Figure2 – Illustration of the Hough Transform Representation

Now that each straight line in the image has been tagged by a unique identifying number (index), the next step is to find an efficient method to save these results. A two dimensional array is created to save the length of each line and the equivalent index of that line. Table 2 illustrates the process of mapping the calculated indexes to the Accumulator array for the line pixels in Figure 3.

		Line Indexes														
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
90°	**0**	0	0	0	3	0	0	2	0	0	0	0	0	0	0	0
45°	**1**	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0
0°	**2**	3	0	0	0	0	3	0	0	0	0	0	0	0	0	0
135°	**3**	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0

Table 2 – Mapping the indexes to the Accumulator in the Pixel Coordinates Method

The Accumulator Array will first hold initial values of 0 in all cells, and later, as the edge pixels are read, the value of the cell belonging to the specific Line Index will be changed if a straight line is detected. The value of the cell refers to the number of consecutive pixels in a straight line, i.e. its length. Obviously there are two ways to do this, it is possible to either stop checking the rest of the pixels in a specific direction once the a line is detected and that to save time (since the target here is to detect lines and not to retrace the image), or it is possible to check all pixels in the line even if a line is detected, and that is to save the length of the longest line with that specific index.

C. Comparison: Hough vs. Pixel Coordinates

Hough Transform, the line intercept calculation in detection process serves the following purposes:

- Distinguish between parallel straight lines (lines with same slope) in the detection stage.
- Guarantee that each pixel is voting to the right straight line that it lying to.

- Use the detected lines parameters to verify if an edge pixel is lying on one of the detected straight lines.

Thus, Hough Transform spends a portion of time calculating the line intercept for each checked edge pixel in one slope (due to the gradient direction constraint). [10] To solve this problem when using the coordinates method, the coordinates of the original image edge pixels flag arrays has an index and a value referring to each possible straight line where: a row has 90 degree line slope angle, column has 0 degree, left diagonal is at 135 degree and right diagonal is 45 degrees. Hence, the Coordinates method:

- Reduces computational cost considerably by using line occurrence flags instead of intercept calculation.
- Corrects the voting mistakes by using the coordinates of image pixels to refer to each straight line instead rounding of the value of calculated intercept for each line.
- Reduces the memory space that required to allocating the accumulator array.

D. *Experimental Results*

The two methods were used on an actual set of images and it was found that for noisy images and dashed lines, the Coordinates Method registered a better detection than the Hough method. The Hough Method had an overall detection percentage of 90.6% while, as expected, the Coordinates method registered an absolute no-error percentage of 100%. The type of images, in which the Coordinates method performed better than the Hough Transform, were the slightly noisy images and dashed lines.

For high-quality and clear images, both methods deliver a 100% accurate results. To illustrate this, one of the high resolution pictures that were used in the experiments is given below in Figure 3a. A triangle and the results of its edge detection are shown below.

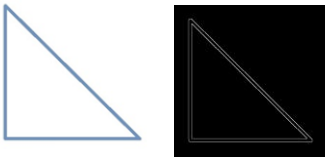

Figure 3a – High Resolution Triangle before and after the edge detection

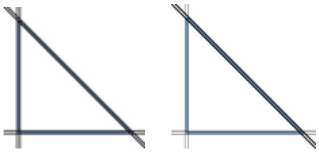

Figure 3b – Hough Result on the left - New Method Result on the right.

As expected, and as illustrated in figure 3b, both methods yielded perfect results with 100% accuracy for high resolution images.

Moving on to lower resolution pictures, one of the pictures that were used in the experiments is given below in figure 4a. The image is a basic triangle with a small square indicating the 90 degrees angle.

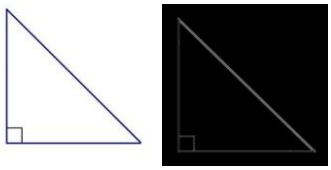

Figure 4a – Lower Resolution Triangle before and after the edge detection

The edge detection of the Triangle edge detection resulted in small “bumps” or imperfections that can be seen when zooming on the edges detected. The results of the line detection using both methods are given in the images below in Figure 4b. As seen, the Hough Transform resulted in missing lines, and that is due to the imperfections in the edge detection, while using pixels, the all lines were detected.

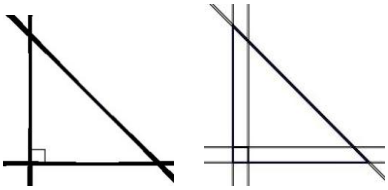

Figure 4b – Hough Result on the left - New Method Result on the right.

Trying the methods on low resolution images containing dashed lines, the image below composes of squares which after the edge detection should constitute the lines to be detected. The edge detection resulted in a bit of distortion in the detected lines.

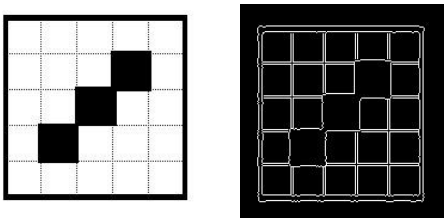

Figure 5a – Image before and after the edge detection

The results of running the Hough vs. the Pixel Coordinates method resulted in the images below in figure 5b. As seen, despite the distortion in some lines, the new method still detected the part that is not distorted while the Hough simply resulted in missing lines. It should be noted that to indicate the missing lines, the

results of both methods were superimposed (laid over) on the original image. The detection percentage for this image was approximately 80% for Hough, and 100% for the new method.

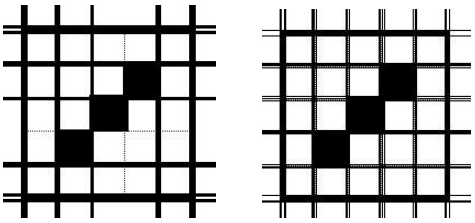

Figure 5b – Hough Result on the left - New Method Result on the right.

Another example is given below in figure 6a, where the image used in the experiment is of low quality. The figure shows the original image and the result of the Canny edge detection which was used as input to test the two line detection methods.

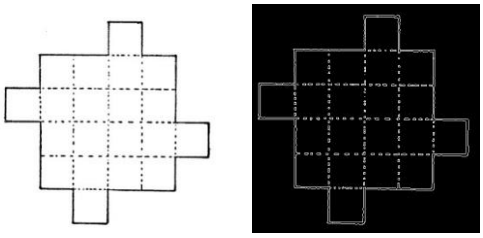

Figure 6a – A low resolution image before and after the edge detection

As expected, the edge detection resulted in slightly distorted detected lines; running the two methods, yielded the results shown below in figure 6b.

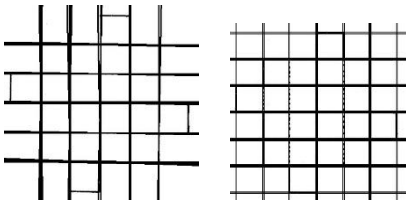

Figure 6b – Hough Result on the left - New Method Result on the right.

Because the results are superimposed on the original image, it is possible to see clearly which lines the Hough missed. The detection percentage for Hough in this specific example was 73% while that of the new method was again 100%.

This variation in the results accuracy is expected, as the new method uses absoluetely no approximations or rounding at any stage, hence the results are absolute. While in the case of Hough, heavy calculations are performed and the rounding of the results is necessary to reduce the size of the Accumulator.

As for the size of the accumulator, the Coordinates Method's Accumulator achieved a 65% size reduction compared to the Hough Transform, and again, this is expected because Hough performs a lot of complex mathematical operations and its accuracy depends on the number of floats allowed and rounding, hence, accuracy comes at the expense of size in Hough, unlike the new method.

Table 2 provides a summary of the main differences between the Hough Transform and the Pixel Coordinates method.

Aspect	***Hough Transform***	***Proposed technique***
Detection Technique	*Uses line intercept*	*Uses pixel position*
Calculations	*Uses calculations both in line detection and application stages*	*Never calculates. Flags are used instead*
Accumulator Array	$2*\sqrt{(h^2 + w^2)} + 1$	$w+h-1$
Conveniency of Use	*Requires a long search in the accumulator*	*Use flags to perform the search rapidly*
Memory Allocated to the Accumulator Array	*Selected as reference (100%)*	*Reduced to be approximately (65%) of Hough size*
Accuracy	*(90.6%)*	*Selected as reference (100%)*

Table 2 – Comparison of the important properties of Hough and Pixel Method

IV. Conclusion

Because the robustness of the Hough Transform method comes with its complexities, like rounding errors and significant accumulator size, this paper presented a less complex basic line detection method, which can work, fool-proof, while reducing time and memory allocation. Although such a method would be limited in scope, it would still be useful in simple applications where the complexity of Hough is not appreciated.

The objective of the Coordinates Method is to detect straight lines in pre-fixed directions (0, 45, 90, and 135) and that by simply using the pixel coordinates as opposed to performing heavy calculations like Hough does.

In a nutshell, the Pixel Coordinates Method is built on the fact that any straight line (vertical, horizontal, or diagonal) in a digital image can be expressed by the pixel coordinates. So, by developing a unique identifier for every line, called index, it was possible to use the data previously read in the edge detection process and use it to fill an Accumulator that will be then thresholded and saved.

The method basically takes the output of an edge detector (Canny was used) and then moves pixel by pixel. In every pixel the code checks if there are pixels adjacent to it in one of the four given direction, if yes, the equivalent the Accumulator array is changed accordingly by putting the number of the detected pixels of the given line in the Accumulator array according to its Index.

The Pixel Coordinates Method successfully outperformed the Hough Transform in terms of accuracy, and Accumulator size.

The run time was not used as a comparison factor because computers are pretty fast nowadays that both methods produce their results in a matter of a couple of seconds even in the case of very large images. Also, Hough

offers a tradeoff between the different parameters (accuracy, processing time, memory allocated), meaning that the "accuracy" of Hough Method can be increased or decreased by: Increasing or decreasing the theta step in the loop and by increasing or decreasing the allowed float numbers (rounding). Hence, experimental results were based on the Hough Transform in its generalized form.

Results of the experiments showed that it is advisable to use the Pixel Coordinates Method as opposed to Hough, in cases when the user is familiar with the nature of the images and is confident that the lines to be detected occur only in the directions covered by the Coordinates Method.

REFERENCES

[1] R.C. Gonzalez and R.E. Woods, Digital Image Processing 3rd ed., Pearson Education, Inc., 2008.

[2] Sachs, Jonathan. "Digital image basics." *Digital Light & Color* 1999 (1996).

[3] Shapiro, Linda G., and George C. Stockman. *Computer Vision*. Upper Saddle River, NJ: Prentice Hall, 2001. Print.

[4] S.K. Naik and C.A. Murthy, "Hue-preserving color image enhancement without gamut problem," *IEEE Trans. on Image processing*, vol.12, no.12, pp.1591-1598, December 2003.

[5] Michael R. Lyu Jiqiang Song, "A houghtransform based line recognition method utilizing both parameter space and image space," Pattern Recognition, vol. 38, pp. 539–552, 2005.

[6] P. V. C. Hough. Method and means for recognizing complex patterns. US Patent 3069654, 1962

[7] Smith M. C. and Winter E. M., "Feature space transform for multitarget detection," in Proc. of the IEEE Conference on Decision and Control, (Albuquerque, NM, Dec. 1980), 1980, pp. 835–836.

[8] Murmu, Rabindra Kumar, and Meena Jhaniya. "Image Segmentation Using Hough Trasform." Diss. National Institute Of Technology, Rourkela, 2009. Web.