Improving the Angular Accuracy for a Fast Window Based Hough Transform

Wolfgang Niemann
GERHARD-MERCATOR-UNVIVERSITÄT,
DUISBURG
Niemann@BitSmart.com

Norbert Härle
QUEENSLAND UNIVERSITY OF TECHNOLOGY,
EESE, BRISBANE
n.harle@qut.edu.au

Abstract

We propose a method, based on the Hough transform, improving the angular accuracy in an edge-detected image. This approach shows good performance in both space and time. We also propose a Hough transform based method for detecting 3D-objects in a 2D edge detected grey scale image. We demonstrate the power of this new method on real grey scale images.

1 Introduction

The Hough transform (HT) [DUDA et al., 1972] has emerged in recent decades as a powerful method for many image processing and pattern recognition applications. The development of fast variants of the original Hough transform has been proposed, including generalised [WONG et al., 1995], parallel implementation [CHUNG et al. 1994] and recursive algorithm [GOETZ et al., 1995] in binary images and gradient based schemes [CUCCHIARA et al., 1998] in grey scale images. Algorithms for multi-sensor-systems are proposed [FORESTI et al. 1992] for 3D object recognition as well. In this paper we propose a method for evaluating the Hough transform for detecting straight lines of 3D-objects in a grey scale image, based on the windowing approach [ZHANG et al. 1994]. Without any loss of speed, the proposed algorithm calculates a straight line twice the precision compared with the one proposed in [ZHANG et al. 1994]. By using information out of the grey scale image, the novel algorithm leaves out a skeletonization to its Hough transform evaluation.

1.1 Hough Transform for Feature Detection

Paul Hough first introduced the Hough transform in 1962 [HOUGH 1962].
The Hough transform can be regarded as a generalised template matching method. It is a well-known method for detecting predefined features like edges and curves in digital images. It can also be regarded as an edge linker because it groups edge pixel and describes them by a higher order entity such as a line equation.
The Hough transform can be used to extract more complex objects like circles and even generalised perhaps non-symmetrical shapes, which can also be seen as pattern matching algorithm. Further the Hough transform is invariant to rotation and translation. The input of a Hough transform is normally an image that has been edge detected. Sobel or Canny edge detectors are well-known methods for edge detection. The information of the transform is stored in "Hough Spaces". Different Hough spaces are used to fit different kinds of shapes like circles, ellipsoids or straight lines.

Two parameters are required to define a straight line in Cartesian coordinates. If we take a point (x', y') in the image-space (x, y), all lines which pass through that pixel have the form

$$y' = mx' + b \qquad \text{(I)}$$

with varying slope m and intersection b. This equation can also be written as,

$$b = -x'm + y' \qquad \text{(II)}$$

where x and y are now constants and m and b are varying. Each different line through the point (x', y') corresponds to one of the points of the line in the (m, b) - space shown in Figure 1. When two pixel p and q lie on the same line, the following assumptions can be considered as reasonable.

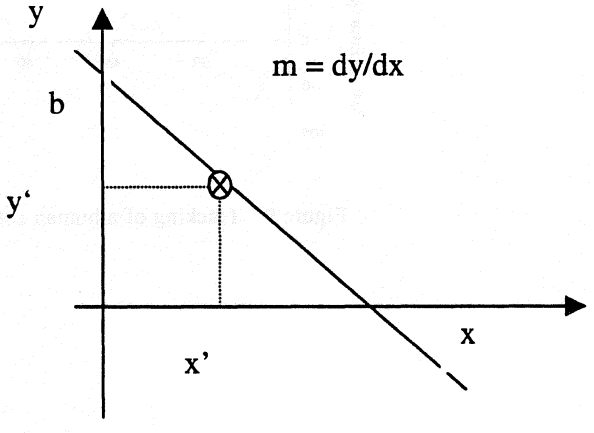


Figure 1 - Straight line in Cartesian coordinates

All possible lines through a pixel are represented by a single line in the (m, b)–space. All pixel which lie on the same line in (x, y) - space are represented by lines, which all pass through a single point in the (m, b)–space. This single point describes the line uniquely and gives the two values m and c. A possible way to detect lines in an image is:

1. Quantise the parameter space (m, b) into a two-dimensional accumulator array A, called Hough space
2. Initialise all elements of $A(m, b)$ to zero
3. For each pixel (x', y') in the (x, y) - space, increase all elements of $A(m, b)$ whose indices m and b satisfy
   $y' = m * x' + b$
4. Local maxima in the accumulator array now correspond to collinear points in the image array. The value of the accumulator array provides a measure of the number of points on the line

This technique is generally known as the Hough technique [HOUGH 1962]. Usually polar coordinates are used, because m becomes infinitely large for lines that are nearly parallel to the y-axis. In Figure 2 a straight line with its polar coordinates is shown. The slope m can be written as

$$m = \frac{dy}{dx} = \tan\theta = \frac{\sin\theta}{\cos\theta} \qquad \text{(III)}$$

The intersection with the x – axis can be computed as

$$x_0 = -\frac{b}{m} \qquad \text{(IV)}$$

From Figure 2 we get

$$\cos\theta = \frac{\rho}{x_0} = -\frac{\rho}{b}m \qquad \text{(V)}$$

and

$$\sin\theta = \frac{\rho}{b} \qquad \text{(VI)}$$

So we can write

$$y = \frac{\sin\theta}{\cos\theta}x + \frac{\theta}{\sin\theta} \qquad \text{(VII)}$$

In polar coordinates one can write

$$\rho = x' \cos\theta + y' \sin\theta \qquad \text{(VIII)}$$

We either vary the angle θ from 0 to 2π and keep ρ positive or vary the angle from 0 to π and let ρ be positive and negative.

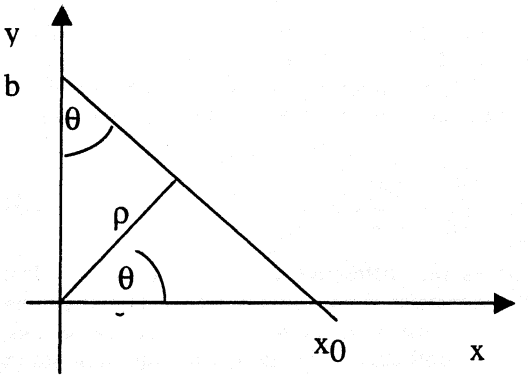


Figure 2 – Straight line in polar coordinates

The votes for the detected line are stored in an accumulator array, the so-called Hough space. The array is an N by M – matrix, where N and M are the parameters for the angles and the distance of the detected line.

A useful property of the Hough transform is that the line segments do not need to be contiguous. This can be very useful when trying to detect lines with short breaks due to noise or overlap with other objects. On the other hand the Hough transform can give misleading results when objects happen to be aligned. The Hough transform gives an infinite line expressed by the pair of ρ and φ (m and b) values, rather than a finite line segment with two well-defined endpoints. For the Hough transform with more complicated curves with more degrees of freedom, the computational needs increase drastically because of the higher number of dimensions for the Hough Space. The needed memory for the Hough – space increases as well.

1.2 A Window Approach to Detecting Line Segments Using the Hough Transform

Because the generalised Hough transform is extremely time and memory consuming many algorithms are developed to make the Hough transform more efficient. One of these, the Window–Hough transform is described in this section. All possible pairs of pixel that describe a straight line segment need to be considered to calculate all intersection points in an N x N – size image. So there are

$$\frac{N(N-1)}{2}$$

possible combinations. The method of the Window Hough transform reduces this to N – 1.
The idea is to determine the pairs of pixels to be

evaluated in a set of small windows (frames). In particular, we define a window centred on the evaluated pixel for each edge pixel. The size of this frame has to be chosen such that it is small enough that very few line segments can be contained in. If there are more than one line with the same distance from the origin and angle in a frame, we provide that these segments are not collinear. Ideally the window contains only one line–segment. The disadvantage is that the smaller window, the fewer angles can be calculated. For example in a frame with the size of 3 x 3 pixel, you can only evaluate eight different angles. Then

$$\partial\varphi = \frac{2\pi}{8} = \frac{360°}{8} = 45° \qquad \text{(IX)}$$

where $\partial\varphi$ is the difference between the angles. The algorithm consists of two main stages. The first one detects line segments in each window and the second combines the collinear and overlapping line segment in the different windows.

Detecting Line Segments in the Windows

Similar to the normal Hough transform, we detect the line segment in a window by calculating the intersection points of sinusoidal curves in the parameter space. Two pixels are necessary to describe an intersection point. This point describes the line that belongs to these pixels uniquely.

$$\rho = x_1 \cos\theta + y_1 \sin\theta$$

$$\rho = x_2 \cos\theta + y_2 \sin\theta \qquad \text{(X)}$$

Let the centre of the frame be (x_1, y_1). For each other edge pixel (x_2, y_2) in the window, we can determine an intersection point (ρ', θ') as follows.
Solving this by equating we get

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

and

$$\rho' = x_1 \cos\theta' + y_1 \sin\theta$$

or

$$\rho' = x_2 \cos\theta' + y_2 \sin\theta \qquad \text{(XI)}$$

The line segments found can be stored in a linked list. A possible structure is described in [ZHANG et al. 1994]. Recall our assumption that no line segment in a frame is collinear with one other. This lets us conclude, that if two line segments in a window have the same angle and the same distance from the origin (they are collinear), then these segments belong to the same line in this window. That means, these segments can be combined into one line. The endpoints are chosen so that the length of this line is maximised.

After processing all pixel in the frame except in the centre, the line segment with the highest vote for ρ corresponds most likely to an existing line segment in the window. You can choose a threshold to detect if a line really exists and is not caused by noise. This procedure is applied to all pixels of the image. The detected lines are stored in a linked list as well. Then detected lines are compared with each other and linked together if the lines have the same angle and distance but are not collinear.

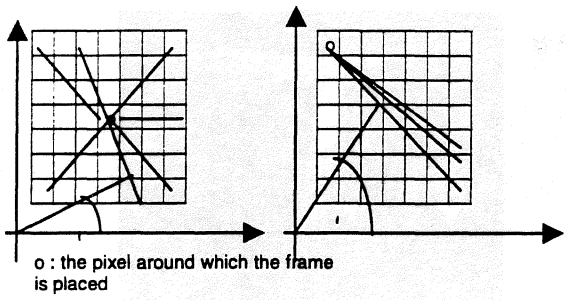
2 The Modified Approach

Some possibilities to speed up and to get a higher resolution for the Windows – Hough – transform are shown in this section.
It is easy to see that the resolution only depends on the frame size. For example in a window of 7x7 computed using the normal windows Hough transform the resolution is at least

$$\Delta\varphi = \tan\frac{\Delta y_1}{\Delta x_1} - \tan\frac{\Delta y_2}{\Delta x_2} = 45° - 33{,}69° = 11.31° \qquad \text{(XII)}$$

with $\Delta x1 = \Delta x2 = 3$ pixels
$\Delta y1 = 3$ pixels
And $\Delta y2 = 2$ pixels as shown in Figure 3a.

For example in a 13x13 frame, the maximal resolution is 5.194°. The basic idea is to set the computed pixel not in the centre, but in the corner (see Figure 3b). With this modification the resolution of a 13x13 frame is available for the 7x7 frame. But Figure 3b also shows that this modification can only recognise lines with an angle φ from zero to 90 degrees. It does not find lines with an angle φ more than 90 degrees. Therefor two different frames are needed to find all lines. The first frame is placed in a way that the reference pixel is in the left top corner. The second frame is placed so that this pixel is in the right top corner. This way, all line segments can be found. The computation time is at least twice as high as the standard windows Hough transform because every pixel has to be computed with two different frames.



(a) Current version of the Win-Hough-transform

(b) New approach of the Win-Hough-transform

Figure 3 – The novel Window-Frame

In order to achieve the same calculation time as the traditional window HT, each pixel is evaluated with alternating frames for each row. Because one frame can either detect lines with an angle from zero to 90 degrees or form 90 to 180 degrees an error of one pixel at the end of the line segment can occur but regarded as noise and neglected. Also because the current pixel is always at the top of the frame, lines at the bottom of the images are already found if they lie inside the frame and it is not necessary to calculate the last FrameSize/2 rows of the image. So you save for a 7x7

$$XFrameSize * YFrameSize + 3 * XFrameSize$$

calculation steps. For an image of the size 256x256 and a window frame size of 3x3 3027 pixel from all 65536 image pixels, approximately 5% without any loss of information. The line segments are already detected. The second improvement is to evaluate only pixels, which have the grey scale in a certain range, called grey scale mode in the following. The reason for the grey scale mode is the smooth transition, including more pixel at the detected edges. So only line segments with a small grey-scale-gradient deviation from the current pixel will be found. This novel method speeds up the calculation time because less angles and distances have to be calculated in the window and it finds lines in an edge detected image more accurately so that a skeletonization can be omitted.

3 Experimental Results

In this section, we use real 2 and 3 dimensional images to measure the performance of the method proposed in this paper. All data are 256x256 grey images taken with a standard RS 170 camera.

3.1 2D-Line Recognition

Figure 4 shows a grey image of a bookshelf. The contours (Sobel Edge detection) of the bookshelf are shown in Figure 5. Figure 6 shows the line segments detected by the current algorithm using a skeletonization and a grey scale threshold of 100 for the range of 255 (8 Bit Colour depth). The lines that shall be detected by the current approach using the grey scale mode are shown in Figure 7. The results, using the new algorithm is shown in Figure 8. Lines with a length shorter than 10 pixels are occluded by thresholding to beware a comfortable handling of these lines. The deviation of the grey scale for the new algorithm is chosen with 5. So pixels with deviation of 5/256 are chosen to belong to the same line.

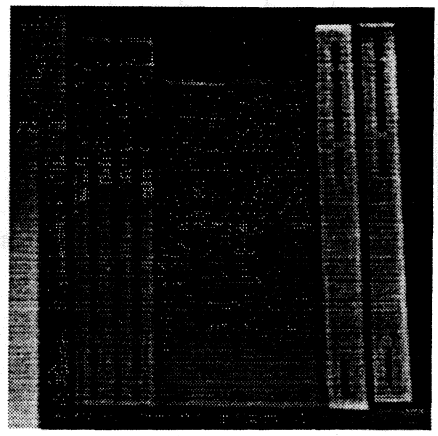


Figure 4 – The grey image of the bookshelf

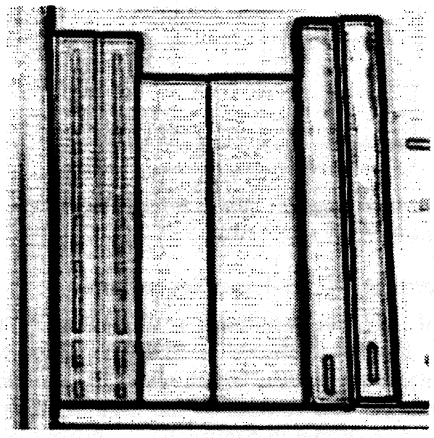


Figure 5 – The contour of the bookshelf

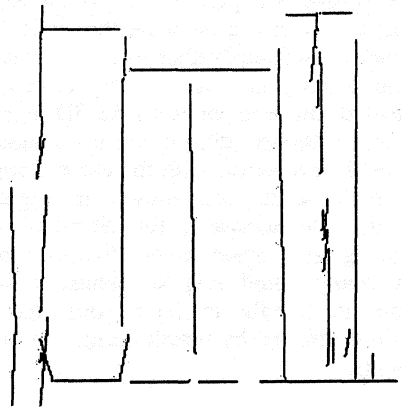


Figure 6 – The line segments obtained from the contour image of the bookshelf by using the current approach with a 3x3 window and a grey scale threshold of 100 and a skeletonization

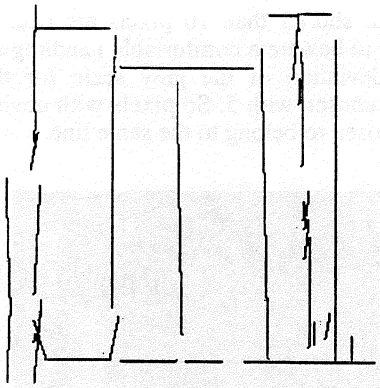


Figure 7 – The line segments obtained by the current approach by using the grey scale mode

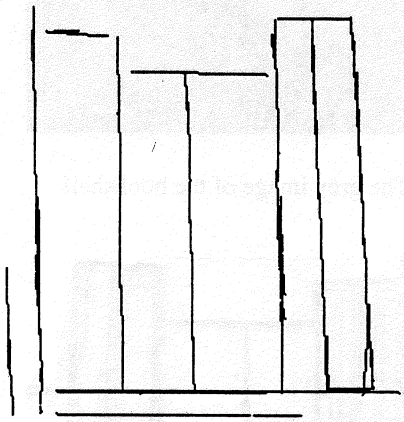


Figure 8 – The line segments that to be detected by the new approach using the grey scale mode

3.2 Detection of 3D-Line Segments

To power of the novel approach is shown in the next experiment. Figure 9 shows a 256x256 grey scale image (8 Bit) of a monitor, which shall be detected by its line segments. The contour of the monitor after the edge detection is shown in Figure 10. It shows that the 3D edges are very smooth, so that a binary Skeletonization algorithm cannot find the exact line. The lines that shall be detected using the current algorithm are shown in Figure 11. Using the Standard Hough Transform with a grey scale threshold (binary or current mode) can found either to much or less line segments than existing, because the grey scale value of the edge detected image depends on the 3D orientation of the edge that a thresholding could not handle. The objects that shall be detected with the current approach using grey scale mode are shown in Figure 12. Because of its low accuracy, the Standard Hough Transform using grey scale mode divides long line segments in many small line segments, which are uncomfortable to handle in later computer vision applications and omitted by thresholding (10 pixel) in our experiments.

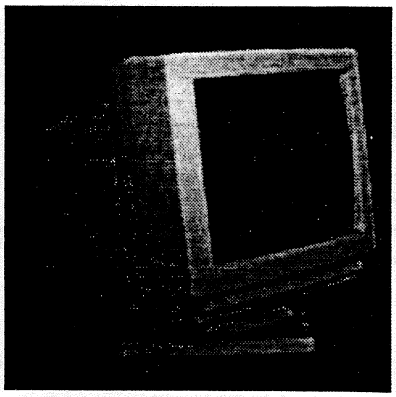


Figure 9 – The 256x256 grey image of the monitor

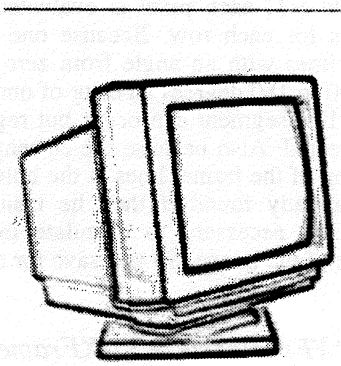


Figure 10 – The contour of the monitor after the edge detection algorithm

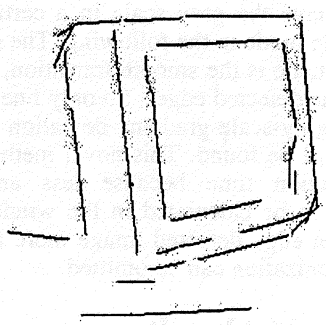


Figure 11 - The detected line segments of the monitor using the "Standard Windows Hough Transform" and a grey scale threshold of 100

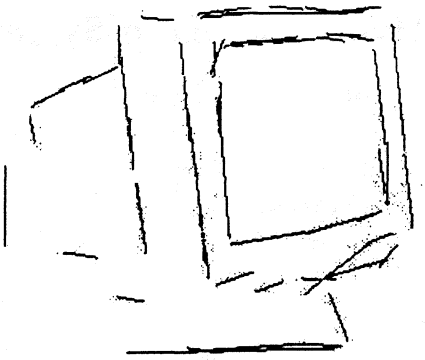


Figure 12 - The detected line segments of the monitor with the current method using grey scale mode.

In Figure 13 the detection of long straight lines with the new approach of the Windows Hough Transform is shown.

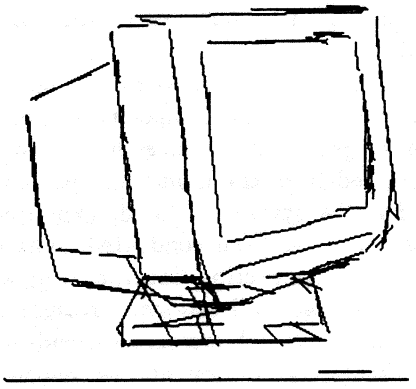


Figure 13 – The detected straight lines using the new algorithm

Because of its higher accuracy, more lines segments are detected, which improves further processing applications.

4 Conclusion

In this paper, we presented a new Hough transform algorithm, improving the angular accuracy for the detection of line segments. A set of small windows is used to divide the task of finding line segments to reduce the operation time. Storing the line segments in a linked list reduces the Hough space as well. Because it works with grey scale images a time intensive skeletonization can be left out. The new Hough transform handles the grey scale gradient for speeding up the calculation time, improves the angular accuracy, and so is a good solution for 3D-line segment-detection as shown in the experiments

5 Reference

[FORESTI et al. 1992] Gianluca Foresti, Vittorio Murino, Carlos S.Regazzoni, and Gianni Vernazza. Grouping of Rectlinear Segment by the Labeled Hough Transform. Image Understanding 58, No. 3, 22-42. 1994.

[WONG et al., 1995] K.C.Wong, H.C.Sim, and J.Kittler. Recognition of two dimensional objects based on a noval generalized h\Hough transform method. 3, 376-383. 1995. IEEE international conference on image processing.

[CHUNG et al. 1994] Kuo-Liang Chung and Horn-Yi Lin. Hough Transform on Reconfigurable Meshes. Computer Vision and Image Understanding 61, No. 2, 278-284. 7-25-1994.

[HOUGH 1962] P.V.C.Hough. A method and means for recognising complex patterns. U.S.Patent No.3,069,654 . 1962.

[CUCCHIARA et al., 1998] R.Cucchiara and F.Filicori. The vector-gradient transform. IEEE transactions on pattern analysis and machine intelligence 20(7), 746-750. 1998.

[DUDA et al., 1972] R.D.Duda and P.E.Hart. Use the Hough transform to detect lines and curves in pictures. Commun.ACM, 1972 15, pp 11-15. 1972.

[GOETZ et al., 1995] W.A.Goetz and H.J.Druckmueller. A fast digital randon transform-an efficient means for evaluatin the Hough transform. Pattern Recognition 28(12), 1985-1995. 1995.

[ZHANG et al. 1994] Y.Y.Zhang and Robert Webber. A Window Approach to Detecting Line Segments Using Hough Transform. Pattern Recognition 29, No. 2, 255-265. 1996.