Line Detection in Digital Images: A Modification to Randomized Line Detection Algorithm

Mr. T P Mithun1 Dr. A Hari Prasad2 Mr. Siddappa Karabannavar3
1Asst. Professor, 2Professor, 3Student, R V College of Engineering, Bangalore-59, Karnataka, India

Abstract—Straight line detection in a digital image is one of the important preprocessing step in computer vision and pattern recognition. Varying lighting conditions and noise makes line detection challenging. In this paper randomized line detection algorithm is been modified to overcome the limitations such as manualisation of the inputs with respect to minimum line length and number of lines need to be detected.

The algorithm described in this paper instantiates after selecting a first non-zero pixel by row scanning and draws a strip from the reference pixel to any other pixel at the boundary of the image. Then the strip is rotated by 360° with predefined step angle. After each step, length of the strip is noted and compared with the previous strip length. Finally strip with the maximum length is considered. This process continues till the last non-zero pixel in defined image reaches to obtain a line detected image.

Index Terms — computer vision, randomized line detection, strip

I. INTRODUCTION

Line detection plays an important role in applications involving face detection using line edge map, eye detection, lane departure warning system, analysis of objects for video understanding etc. However it is challenging task due to the presence of various problems like varying lighting conditions and noise. Several approaches have been proposed in the literature that includes, Hough Transform (HT) proposed by Hough et al [1] for line detection. Even though it is extensively used, the major obstructions with this method are its computational entanglement and storage concern. Along with B-spline algorithm Hough transform is also used for identifying mid-line [8] in a curved road to assist the drivers to stay at the centre of the road. Duda et al [2] modified the HT approach for detecting curve in a particular direction. Philip et al [3] extended the work by Duda and Hart to reduce the computational complexity and implemented the modified algorithm in parallel machine to reduce the execution time. Ravi et al [4] implemented the HT in mesh connected parallel computers using single input multiple data (SIMD) architecture. Charles Dixon et al [5] put forward a line detection algorithm for scene modeling. Here the main spotlight is on horizontal and vertical lines in binary images. To reduce the storage space and computational time Guoquan Jiang et al [6] proposed a randomized line detection method to detect the lines in noisy images. To increase the accuracy of line detection and to reduce the parameters used, Andres Solis Montero et al [7] proposed a method for line detection which is based on repeated segment directions on image contours.

In this paper a modified randomized line detection algorithm is described. The proposed algorithm is based on randomized method. Rest of the paper is organized as; section II describes the straight line detection based on randomized method along with its drawbacks. In section III, a modified randomized line detection algorithm is given along with its advantages over randomized method. Sections IV and V give the results and conclusion, respectively.

II. LINE DETECTION BASED ON RANDOMIZED METHOD

In digital images not all the edge pixels lie on a straight line [6] since edge orientation and its occurrence may vary from image to image. However, edge pixels can be approximated as a straight line if they lie roughly on a line. This can be shown in figure 1. In randomized line detection first the strip is drawn to make out the actual line. Strip contains two straight lines at a distance ‘d’ from the middle line one at the top and other at the bottom. Based on the threshold used whether the line is desired line or false line has been resolved. Various steps involved in randomized line detection method are briefly explained next.

1. Coordinates of all the pixels are stored in a set V. Minimum length of the line is determined to set the threshold. And Note the number of lines to be determined.
2. Select two edge points randomly and then calculate number of pixels and line length. If both are within the threshold then the line is desired one else it is false line. If the detected line is false then increment the failure counter by one, discard the edge points, select two new points, and repeat step two.
3. If the line is desired one then increment the line counter by one and continue the entire procedure until all the lines have been detected.

There are certain limitations associated with randomized line detection. They are
1. Minimum distance threshold used for one image may not be applicable for other images.
1. It is difficult to identify number of lines that are present in complex images.

These two can be easily observable from the figure 2. Where the minimum line length threshold differs from image ‘a’ to ‘c’. Also it is very difficult to identify the number of lines present in those images.

III. MODIFICATION TO RANDOMIZED LINE DETECTION ALGORITHM

Randomized line detection algorithm is modified to overcome the limitations stated in the previous section. Detailed steps involved in modified randomized line detection algorithm are as follows (figure 3 describes the flow of the algorithm)

1. Scan the rows for first non zero pixel value, say $P$ at $(y, x)$.
2. Take the point $P$ as a reference point and draw a strip to the pixel at the position $(N, 1)$.
3. Scan the columns for nonzero pixel values starting from $(y, x-1)$ position. Compute $d$ for each non zero pixels using the formula

$$d = \frac{|y_m - a * x_m - b|}{\sqrt{a^2 + 1}} \ldots (1) \text{ (courtesy [6])}$$

4. If $d$ is between 0.707 to -0.707 then corresponding nonzero pixel is on the line else it is not on the line.
5. If the pixel is on the line, check for the next column and repeat the process until 1st three consecutive zero pixels occur. Then compute the Euclidean distance between the reference point and last nonzero pixel value, say $D_1$.
6. Shift the strip end point position to $(N-2, 1)$ and repeat steps 3, 4, 5 also calculate $D_2$. If $D_2$ is greater than $D_1$ then discard $D_1$ retain $D_2$ else retain $D_1$.
7. Continue shifting of the strip to the entire 360 degree, calculate $D_i$ in each shift, compare it with $D_{i-1}$ and retain the maximum.
8. When entire 360 degree is covered, the line with maximum $D$ is the desired line. Clear all the nonzero pixels along the line (pixels with $d$ between 0.707 to -0.707 ). And draw the line between reference pixel and the last nonzero pixel.
9. Now scan the rows for the next nonzero pixel and repeat steps 2 to 8.
10. Continue scanning of rows and repeating steps 2 to 8 until the last pixel in the last row is reached.

IV. RESULTS

Proposed algorithm was tested for different real world digital images. As a preprocessing step edge detected image is obtained by employing a canny edge detector (because of its high noise reduction capability canny edge detector is preferred over others). Figure 4(a), figure 5(a), figure 6(a) and figure 7(a) shows the RGB images used for testing the proposed algorithm. Figure 4(b), figure 5(b), figure 6(b) and figure 7(b) represent the output of canny edge detector. Next the modified algorithm for line detection is adopted to obtain the line detected image. Figure 4(c), figure 5(c), figure 6(c) and figure 6(c) shows the output of the proposed algorithm. From the results it is clear that modified randomized line detection method provides good results.
Fig 4 (a) Original RGB image, (b) output of canny edge detector after inversion, (c) output of proposed algorithm.

Fig 5 (a) Original RGB image, (b) output of canny edge detector after inversion, (c) output of proposed algorithm.
Fig 6 (a) Original RGB image, (b) output of canny edge detector after inversion, (c) output of proposed algorithm.

Fig 7 (a) Original RGB image, (b) output of canny edge detector after inversion, (c) output of proposed algorithm.
V. CONCLUSION

The algorithm for line detection was successfully implemented using Matlab. Proposed algorithm is based on randomized line detection method. Difficulty involved in defining the line threshold for images and identifying the number of lines present can be overcome by using the modified randomized line detection method.

REFERENCES