A Test Framework for the Accuracy of Line Detection by Hough Transforms

Thuy Tuong Nguyen  
School of Information and Communication Engineering  
Sungkyunkwan University  
Suwon, Korea  
Email: nthuy@skku.edu

Xuan Dai Pham  
School of Information and Communication Engineering  
Sungkyunkwan University  
Suwon, Korea  
Email: pdxuan@ece.skku.ac.kr

Dongkyun Kim  
School of Information and Communication Engineering  
Sungkyunkwan University  
Suwon, Korea  
Email: blusterkdg@skku.edu

Jae Wook Jeon  
School of Information and Communication Engineering  
Sungkyunkwan University  
Suwon, Korea  
Email: jwjeon@yurim.skku.ac.kr

Abstract—The detection of lines in an image is an important task. In the fact that many works have been done on line extraction, there is a lack of a comprehensive comparison of the so far proposed algorithms. The design and implementation of a framework to test line detection algorithms on intensity images is described in this paper. Our test framework is applied to compare the correctness and precision of lines extracted by Standard Hough Transform, Progressive Probabilistic Hough Transform and a proposed method based on Standard Hough Transform. The correctness of the extracted lines relates to global accuracy whereas the precision concerns accuracy at a local level.

The well-known Standard Hough Transform (SHT) and Progressive Probabilistic Hough Transform (PPHT) are two of the most efficient algorithms for line detection. SHT can detect almost straight lines in the image, and it is highly resistant to noise. Line segments are effectively found by PPHT. However, this algorithm has lower accuracy than SHT. The proposed method based on SHT overcomes this. It contains three extensions: the technique of accumulation, the application of a local maxima rule, and the detection of line segments.

The test framework enables us to evaluate the advantages and disadvantages of the three Hough Transform algorithms by analyzing the results of line extraction.

I. INTRODUCTION

Selecting the best method to extract lines from input data is a difficult task for anyone building a line-based computer vision system. Many algorithms have been proposed to extract line features from 2D images. Two popular approaches are: the Split-and-Merge algorithm [1] and the Hough Transform (HT) [2], [3]. The Split-and-Merge algorithm has several desirable properties. It exploits the local structure, whereas HT only considers global structure [4]. The Split-and-Merge algorithm is mainly used in mobile robots navigation systems such as localization, dynamic map building, and path planning [4]. This local method is characterized by a lower computational cost than global method but it is not robust to noise. Other methods are also referred in this paper for their benefits.

Line regression algorithm is presented in [5] for map-based localization. Random Sample Consensus (RANSAC) [6] is an algorithm for robust fitting of models in the presence of data outliers. This algorithm is popular in computer vision to extract image features. The Expectation-Maximization (EM) is a probabilistic method and has been used as a line extraction tool in computer vision. It has some shortcomings. It can be trapped in local maxima and it is difficult to choose good initial values [7].

There are many algorithms that may be selectively applied to different particular purposes. We need to find out which algorithms are most appropriate for different environments. It is necessary to give comparisons of line extraction algorithms. The methods may be selected as the best candidates with respect to speed, correctness or precision. The work described in [4] gives an evaluation of six line extraction algorithms on range scans. The selected algorithms included Split-and-Merge, Line Regression, Incremental, RANSAC, Hough Transform, Expectation-Maximization. These are most commonly used in mobile robotics and computer vision. The paper is considered to be adequate to the comparison of line detection methods. Comparison criteria are speed, complexity, correctness relating to global observation and precision relating to local observation. The comparison is only applied to algorithms used in 2D laser scans for indoor environments. The Split-and-Merge and Incremental predominate in the comparison of [4] because of their sequential order property in scanning. There is no experimental evaluation presented for general-purpose problems. Hence, the algorithms which are successfully applied to line detection on intensity images are not robust in this comparison such as Hough Transform algorithms. In addition, [4] does not provide much in the way of comparison criteria to evaluate the line extraction results.

Comparing Hough Transform methods, [8], [9] and [10] show the output of line extraction with respect to computation time, memory usage and number of detected lines. Hough Transform methods in [8] are divided into two categories: probabilistic and non-probabilistic methods. The overview and comparison of these eleven algorithms are given but only the number of detected lines is mentioned, along with speed, and size of the parameter space. This information is insufficient to draw conclusions as to the methods’ accuracy. The Standard Hough Transform (SHT) and the Randomized Hough Transform (RHT) variants are described in [9]. They are also compared with respect to the number of detected lines. In [10], the Standard Hough Transform, the Counter-Based Hough Transform, the Fast Incremental Hough Transform and the Randomized Hough Transform and its variant are...
described. The scheme implemented in this paper does not seek line end-point information. These three papers judge that SHT is the preferred method, but its computation speed is very low.

In this paper, a test framework is designed to evaluate the results of different line extraction algorithms that are mainly used on intensity images. It gives more details on the correctness and the precision of the extracted lines. These data relate to local as well as global accuracy. Moreover, an extension based on the well-known line detection algorithm, Standard Hough Transform, is compared to other Hough Transform algorithms. The line detection algorithms used in our work are Standard Hough Transform (SHT), Progressive Probabilistic Hough Transform (PPHT) and the proposed method. The Hough Transform tends to be most successfully applied to line detection on intensity images. Therefore, this class of algorithm is evaluated in the general circumstances. We are interested in finding out the most appropriate way to evaluate the line extraction results and how well the proposed extension works without considering the computational burden imposed by the algorithms. Experiments are performed on two sets of synthetic images generated randomly and two real-world images. These synthetic images are divided into two categories: line segments impacted by noises and ones not impacted by noises. They are sufficiently complicated to simulate typical input. Real-world images are then given to finalize the tests of the evaluation method. They are also used to determine if the proposed extension can overcome the disadvantages of SHT and PPHT.

This paper is organized as follows. In the next section, the Hough Transform and related problems will be described. In Section III, the argument for the proposed extension of SHT will be revealed. Section IV presents the test framework method for SHT, PPHT and our proposed method. The experimental results will be presented in Section V. This paper is drawn to a conclusion in Section VI.

II. THE HOUGH TRANSFORM

This part introduces the notion of the Hough Transform and its two variants Standard Hough Transform and Progressive Probabilistic Hough Transform. The Hough Transform [2], [3] has been the most popular algorithm for extracting global features such as straight lines, circles, ellipses, from an image. This algorithm uses an array called parameter space to detect the existence of a line. For each pixel in the edge image, the Hough transform algorithm determines if there is enough evidence of a line at that pixel. It calculates the parameter values and accumulates the cells in the parameter space according to all the pixels. Then, by finding the cells with the highest values, the most likely lines can be extracted. The simplest way to find these peaks is to apply some form of threshold. Finally, the segments are extracted using the information obtained from the maximum positions.

The basic Hough Transform, also called the Standard Hough Transform (SHT) [2], [3], has established itself as the default technique for straight line Hough transform evaluation. Its popularity comes from its robustness to noise and the very simple algorithmic implementation of SHT. In SHT all the points \((x, y)\) are calculated using the equation expressed in polar coordinates

\[
\rho = x \cos \theta + y \sin \theta
\]

to obtain \(\rho\) with a change in the succession of \(\theta\) in the parameter space. Having the values \((\rho, \theta)\), the SHT algorithm then looks for the accumulator’s cells that the parameters fall into, and increases the values of those cells. After some edge pixels considered to constitute a straight line have been accumulated, the distribution of the parameter space is shaped like a butterfly [11]. The most likely straight line in the image domain is represented by the highest value in the parameter space. The peak lies at the center of the butterfly. When all the accumulated points form a chaotic parameter space, several overlapping butterflies are obtained. The present accumulator has peaks that may have values greater than the specified threshold. The simplest way to find these peaks is to compare the values of the peaks with the threshold value. Then the peaks with the highest values are compared with other peaks in the neighborhood using predefined windows. We can draw straight lines from these peaks.

The SHT and Progressive Probabilistic Hough Transform (PPHT) are two of the most efficient algorithms for line detection. Although robust properties, SHT has drawbacks in its high computational and storage requirements, and it provides only the \(\rho\) and \(\theta\) parameters of the straight lines. This simple transform fails to determine any more information about the length or the start and end points. PPHT, an advanced Hough transform algorithm, can solve the problems that occur in SHT. PPHT originated from the Probabilistic Hough Transform [13]; it is also a randomized version of the Hough Transform. Its objective is to minimize the proportion of points that are used in voting while trying to maintain false negative and false positive detection rates at the level achieved by SHT. The use of the same one-to-many voting pattern with probabilistic methods can decrease the requirements of computation and storage while detecting the line segments [14]. These benefits are obtained by the way of accumulating and finding supporting points. Line segments are obtained by PPHT. Problems that affect the accuracy of this algorithm make it less accurate than SHT. The analysis will be revealed in the remainder of this paper.

III. THE PROPOSED EXTENSION OF SHT

An extension of the Standard Hough Transform (SHT) is presented in this section to accurately detect line segments. This method is based on the robust features of SHT and then widened by three extensions: the “weighted” accumulation, the local maxima rule and the elimination of the agglomerate lines in the same area. It is simple but can yield effective results. It can overcome the shortcomings of SHT and PPHT. An experimental result of the proposal is shown in this paper in order to compare the SHT extension to the PPHT. From these comparisons, we can understand the effect of the extension
of the standard algorithm. In PPHT, we ignore the step of choosing the threshold automatically, which was approximated with a probability model. Three extensions discussed in this section are the technique of accumulating, the local maxima rule, and the determination of the line segments.

A. Accumulation technique

The first step of line extraction using the Hough Transform is to accumulate pixels of the image in the parameter space. This step seems to be easy, but it has contributed much to the fact of choosing lines. The manner of managing the cell values gives a special importance. The coordinates of the pixel are input to the line equation which is expressed in polar coordinates, and then the \( \rho \) value termed \( \rho_{\text{exact}} \) is calculated. It is unclear whether \( \rho_{\text{exact}} \) belong to \( \rho_n \) or \( \rho_{n+1} \). This doubt can be resolve in different ways. The most straightforward approach can be elucidated as follows: for each \((x_i, y_i)\) and for each \(\theta\) along the \(\theta\)-axis of the parameter space, compute

\[
\rho_{\text{exact}} = \frac{x_i \cos \theta + y_i \sin \theta}{\sin \theta},
\]

and let \(\rho_{\text{exact}} \approx \rho\) (Shapiro 1978; Thrift and Dunn 1983; Galkowski and Galkowski 1986; Ballard 1987) [14]. Niblack and Petkovic [1] have also presented two more techniques of accumulating, but only the first method is selected because of its simplicity and efficiency: let \((x_i, y_i)\) be one of the points, and let \(\rho_{\text{exact}} = x_i \cos \theta + y_i \sin \theta\); then, if \(\rho_{\text{low}}\) and \(\rho_{\text{high}}\) are consecutive values along the \(\rho\)-axis such that \(\rho_{\text{low}} \leq \rho_{\text{exact}} < \rho_{\text{high}}\), then calculate

\[
H(\rho_{\text{low}}, \theta) = H(\rho_{\text{low}}, \theta) + (\rho_{\text{high}} - \rho_{\text{exact}})
\]

\[
H(\rho_{\text{high}}, \theta) = H(\rho_{\text{high}}, \theta) + (\rho_{\text{exact}} - \rho_{\text{low}})
\]

where \(H(\rho_n, \theta)\) represents the value at the cell \((\rho_n, \theta)\).

B. Local maxima

In SHT, after accumulating all pixels in the image to the Hough space, the lines are selected only if their values at the corresponding cells are greater than a certain threshold. In a density image, it is easy to acquire several bins having high values that are greater than a certain threshold. In particular, these bins are located in the same small local areas. However, in many cases, the extracted lines should not be very close to each other so that they give a good condition to detect more lines in the rest of the image. We can avoid a case in which many lines are located in the same straight area whereas there are no lines that satisfy the condition in the list of "longest" lines. The size of a local window to eliminate unsatisfied lines can be easily specified depending on the usage as well as the content of images. Then, the maximum value in the local area is selected to represent a line.

Although the local maxima rule has been generally used, it is necessary to refer to the PPHT algorithm in order to verify whether the rule has been applied [13]. After checking if the highest peak in the accumulator that was modified by the new pixel is higher than the threshold; if this condition is satisfied, there is only one highest value in the Hough space at this time. Contrary to SHT, it might not be necessary to use local maxima to choose which line is the most reasonable. PPHT shows that it does not take interest in the position of lines.

C. Finding line segments

In the PPHT technique, a cell that satisfies the threshold is chosen, after which contiguous points are found based on the last pixel accumulated [13]. These related pixels have to be on the line \((\rho, \theta)\) that currently has the highest value in the parameter space. The extension of SHT also applies the method of finding neighboring pixels after finishing the step picking out \((\rho, \theta)\) from the local maxima pace. This part can be summarized as follows: look along a corridor specified by the peak in the accumulator, and find the longest segment either continuous or exhibiting a gap that does not exceed a given threshold.

Finally, accurate line segments are extracted. They are not only the straight lines detected by SHT but also they are the line segments having accurate positions in the image. Furthermore, these proposed extensions might reduce the agglomeration of high-valued lines in the same area. This problem is usually encountered in PPHT due to its algorithm which includes selecting pixels and finding the supporting points.

IV. METHOD OF TEST FRAMEWORK

This method is proposed to evaluate the results of line extraction algorithms. In our work, we apply the method to three Hough Transform based algorithms that are most popular used on intensity images. Standard statistical methods will be applied on the correctness and the precision of detected lines. These data include the number of possibly detected lines, correct lines, duplicate lines, superfluous lines, false negative lines, the false positive, and the errors of angle \(\theta\), distance \(\rho\) and lengths. We divide this method into two parts: testing with synthetic images and testing in case of real-world images. A synthetic image contains lines that are drawn pseudo-randomly, according to parameters defined at runtime. Line detection algorithms are applied to each simulated edge-image, and then the parameters of the detected lines are compared with those of the known inputs. Performance statistics are accumulated automatically over many images. Similarly, real-world images are evaluated using their own ground-truths that are constructed by hand. It is difficult to find appropriate input parameters to get the best lines from real-world images, thus the tests are necessary for the trial-and-error process.

Evaluation of synthetic images is resulted from trials of different algorithms using sequences of 20 Random Number Generator (RNG) generated 240 \(\times\) 320 pixel images. Each image contains 10 line segments, of length from 100 to 400 pixels, distributed in any orientation. To enhance testing complexity, noises are added. They are randomly located around line segments using Gaussian RNG with a mean of 0 and standard deviation of 20 pixels. The number of noises depends on the length of line segments; we can imagine that 0 to 20 random pixels are placed near each pixel positioned on each
line segment. Although, these parameters are fixed in the test framework, they can be easily modified for more complicated or sparser cases. Afterward, trials are made on these images to get the measurement of correctness and precision.

The correctness measures are defined as follows:

- Number of possibly detected lines by the algorithm ($N_{\text{detected}}$).
- Number of correct lines ($N_{\text{correct}}$), an important parameter of correctness.
- Number of duplicate lines ($N_{\text{duplicate}}$): in case of evaluating line segments, we define an area surrounding each line segment in order to limit agglomerate ones. All segments locating in this line’s buffer are called duplicate lines with respect to that line except the correct one. We set up this parameter to evaluate the algorithm in detecting adjacent line segments. Thus, if an algorithm can detect most of lines, but it has mistakes over small regions, it cannot be judged as a robust method.
- Number of superfluous lines:

$$N_{\text{superfluous}} = N_{\text{detected}} - N_{\text{correct}} - N_{\text{duplicate}}$$

- Number of false negative lines:

$$N_{\text{false negative}} = N_{\text{true}} - N_{\text{correct}}$$

where $N_{\text{true}}$ is the number of lines generated in the ground-truth image.

- False positive:

$$FP = \frac{N_{\text{duplicate}} + N_{\text{superfluous}}}{N_{\text{detected}}}$$

To determine the precision, we define the following three sets of errors on line parameters:

$$\Delta \Theta = \{\Delta \theta_i = \theta_i - \theta_i^T, i = 1..n\}$$

$$\Delta P = \{\Delta \rho_i = \rho_i - \rho_i^T, i = 1..n\}$$

$$\Delta L = \{\Delta l_i = l_i - l_i^T, i = 1..n\}$$

where $n$ is the number of correct lines; $\theta_i^T$, $\rho_i^T$, $l_i^T$ are line parameters of a true line; $\theta_i$, $\rho_i$, $l_i$ are of the corresponding detected line. The error distributions are assumed to be Gaussian distributions. The standard deviations of the three distributions are calculated as follows:

$$\sigma_{\Delta \theta} = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\Delta \theta_i - \overline{\Delta \theta})^2}$$

$$\sigma_{\Delta \rho} = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\Delta \rho_i - \overline{\Delta \rho})^2}$$

$$\sigma_{\Delta l} = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\Delta l_i - \overline{\Delta l})^2}$$

V. EXPERIMENTAL RESULTS

Experiments are conducted on a list of Random Number Generator generated images with the objective of evaluating the quality of the output of two Hough Transform algorithms and the proposed extension. We use the same sequence of 20 images for all algorithms. The algorithms were run with an accumulator resolution of 1 pixel for $\rho$, 1.15 degrees for $\theta$, 100 for threshold, 20 pixels for minimum length of the detected line, 3 pixels for the gap in finding adjacent points, and maximum in detecting lines. The gap and length do not relate to a straight-line extraction problem. The local maxima window has size of $5 \times 5$ pixels. The parameters of the line buffer, allowed difference of $\rho$, of $\theta$, and line length for benchmarking are respectively 5 pixels, 5 pixels, 11.5 degrees and 5 pixels. The duplicate lines are taken if their errors of $\rho$s and $\theta$s are lower than the allowed difference.

The benchmark results of algorithms without noise are shown in Fig. 1. There are nine charts corresponding to the evaluation values considered. The vertical axes are noted with the corresponding units of the measurements. Twenty images are represented from number 1 to 20 in the horizontal axis of each chart. We intentionally omitted the ordinal number for a clearer view. With the total number of true lines is 200, we see that PPHT performs well in the measures of $N_{\text{detected}}$, $N_{\text{duplicate}}$, $N_{\text{superfluous}}$, $FP$. However, in one of the most important evaluation parameters, $N_{\text{correct}}$ or $\%_{\text{correct}}$, PPHT performs worst. We also collate the charts in Fig. 1 with the data shown in Tab. I. The three sets of errors do not significantly differ between the algorithms. They might be fine for accuracy. In this test case, the proposed extension of SHT, SHT extension, is the best method judged by correctness in extracting line segments, although its precision is weaker than for PPHT.

In Fig. 2, the number of correct lines, false positive and errors of lengths are shown to be the most important parameters in case of testing with noise. SHT is impressive with nearly perfect correctness result. PPHT and SHT perform similarly. The errors are small, so they do not impact overall accuracy. Fig. 2 shows that the SHT extension is not very robust in a noisy situation; therefore, it is evaluated not to be the best in this test.

Fig. 3 and Fig. 4 show the real-world images that are used for a similar test to that of the pseudo-randomly synthetic images. The resolution of the shelf image shown in Fig. 3-(a) is $300 \times 320$ pixels, and the following parameter set is used: 30 pixels for minimum line length, 6 pixels for maximum gap, 100 for threshold, and the maximum detected line segments. With the $180 \times 180$ pixels street image shown in Fig. 4-(a), we use 30 pixels for minimum line length, 3 pixels for maximum gap, 60 for threshold, and also maximum detected line segments. Fig. 3-(c), Fig. 3-(d) and Fig. 4-(c) show the results with number of true lines are 31 and 9, respectively. These tests only consider PPHT and SHT extension as the testing objects.

Fig. 3 and Tab. II show that PPHT omitted many line
segments, while the SHT extension can yield good results. We admit that the threshold can be the same for both algorithms, but there is a problem of inconsistency. With regard to the detected line segments near to the left of the shelf, it is easy to see only four line segments are found whereas there are five line segments remain that have equivalent length. Similar situations are found on the right. Choosing lines whose lengths are greater than 30 pixels, we considered the elimination of vertically parallel lines at the left and right of the image. PPHT shows its inconsistency in detecting separate lines of one long line. This line might have a high accumulated value, but it cannot preserve the advantage of a long line. After determining a supporting pixel whose cell value is greater than the threshold, PPHT decreases the value at the cell that it had already accumulated [8], [9]. Using the PPHT algorithm, it is possible that this long line may not be detected. In addition, only small line segments instead of correctly long lines are detected at the top, center and bottom of the edge image although a clear edge image was provided.

The last experimental image, Fig. 4, is a general image used to test the robustness of two methods. Although this street image is not simple, most of main line segments were obtained correctly. We can see the detailed result of correctness and precision in Tab. II. These data are presented the same way as the first experiment.

Expanding the idea of testing real-world images, we can apply the evaluation method to large sets of street images,

**Fig. 1.** Benchmark results without noise of SHT, PPHT and SHT extension.

**Fig. 2.** Benchmark results with noise of SHT, PPHT and SHT extension.

**TABLE I**

RESULTS OF THREE ALGORITHMS IN CASE OF TESTING WITH SYNTHETIC IMAGES

<table>
<thead>
<tr>
<th></th>
<th>Without Noise</th>
<th></th>
<th>With Noise</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SHT</td>
<td>PPHT</td>
<td>SHT ext.</td>
<td>SHT</td>
</tr>
<tr>
<td>N_detected</td>
<td>334</td>
<td>203</td>
<td>265</td>
<td>545</td>
</tr>
<tr>
<td>N_duplicate</td>
<td>106</td>
<td>13</td>
<td>42</td>
<td>27</td>
</tr>
<tr>
<td>N_superfluous</td>
<td>38</td>
<td>31</td>
<td>54</td>
<td>75</td>
</tr>
<tr>
<td>N_falsenegative</td>
<td>10</td>
<td>41</td>
<td>31</td>
<td>7</td>
</tr>
<tr>
<td>N_found</td>
<td>42.55</td>
<td>20.79</td>
<td>33.24</td>
<td>63.81</td>
</tr>
<tr>
<td>N_correct</td>
<td>95.00</td>
<td>79.50</td>
<td>84.50</td>
<td>96.50</td>
</tr>
<tr>
<td>Δθ (degree)</td>
<td>0.34</td>
<td>0.45</td>
<td>0.68</td>
<td>0.86</td>
</tr>
<tr>
<td>Δρ (pixel)</td>
<td>0.53</td>
<td>0.71</td>
<td>0.87</td>
<td>1.05</td>
</tr>
<tr>
<td>Δl (pixel)</td>
<td>n/a</td>
<td>78.62</td>
<td>71.71</td>
<td>n/a</td>
</tr>
<tr>
<td>σΔθ (degree)</td>
<td>0.16</td>
<td>0.32</td>
<td>0.49</td>
<td>0.38</td>
</tr>
<tr>
<td>σΔρ (pixel)</td>
<td>0.18</td>
<td>0.61</td>
<td>0.46</td>
<td>0.27</td>
</tr>
<tr>
<td>σΔl (pixel)</td>
<td>n/a</td>
<td>33.22</td>
<td>29.73</td>
<td>n/a</td>
</tr>
</tbody>
</table>
or room images, or even furniture images in order to find out which algorithm is the best to extract lines in a particular environment. The experiment on the shelf and the street image judges that SHT extension might be better than PPHT, but in other situations PPHT may predominate. These two images are given in order to analyze the pros and cons of the different line extraction methods; we need to conduct more experiments to determine the optimum algorithm for each environment.

VI. CONCLUSION AND FUTURE WORK

This paper introduces a test framework for evaluating the results produced by line detection algorithms. The framework is used to compare the correctness and precision of line extracted by Standard Hough Transform (SHT), Progressive Probabilistic Hough Transform (PPHT) and proposed extension of SHT. Although our proposed method yields good results in benchmarking with synthetic and real-world images, it is much slower than PPHT or other fast Hough Transforms because it inherits the characteristics of the SHT algorithm. That is after completing the SHT phase; it continues to find line segments. This might be the biggest disadvantage of this proposed extension.

For future work, we plan to manipulate the test framework to investigate the results under different environmental conditions such as in rooms, and streets. This results will enable users to select the most appropriate algorithm for their situation.

**ACKNOWLEDGMENT**

This research was performed as part of the Intelligent Robotics Development Program, one of the 21st Century Frontier Research and Development Programs funded by the Ministry of Commerce, Industry and Energy of Korea.

**REFERENCES**


