

Line Detection Techniques for Automatic Content Conversion Systems

Costin-Anton Boiangiu, Bogdan Raducanu
Computer Science Department
“Politehnica” University of Bucharest
Splaiul Independentei 313, Bucharest
ROMANIA

Costin@cs.pub.ro, BRaducanu@student.cs.pub.ro

Abstract: In an image processing application there is often need to identify and extract different morphological elements such as characters or lines. This paper studies one general method of identifying vertical or horizontal lines. However, the techniques described here can be used to detect other analytical objects like circles, or even ellipses. Line Detection is an important add-on to an automatic content conversion system which builds digital documents from scanned papers. After identifying lines, other layout elements can be extracted: columns, paragraphs, tables, headers. The present paper is a study of the Hough Transform for which several new enhancements are introduced.

Key-Words: - automatic content conversion, line detection, edge detection, Hough transform, feature extraction

1 Introduction

In developing our automatic content conversion system, we found the need for a Line Detection subsystem implemented at application level. Ideally, this system will be able to identify straight lines in a black and white image.

1.1 Automatic Content Conversion Systems

Such a system is employed to construct digital documents from hard copies of papers, books, newspapers, etc. In today's ever growing digital networks, systems like these can bring the final boost in implementing everything through computers. Old newspaper archives, full digital libraries, can all be achieved with an increasing accuracy.

The job of an Automatic Content Conversion System (ACCS) is to first provide an efficient scanning of the documents. This includes optimizations for correct alignment, exposure, noise elimination, color balance, etc. Then comes the Layout Analysis part which tries to construct the logic in which the input document was intended [2,3], i.e. identifying the original columns, paragraphs, images, tables, titles, etc. and including them in the digital document obtained.

An OCR engine is used to extract text from the image. The purpose is to constantly improve the OCR results and perform fewer corrections.

Statistical reports are generated once the document is ready. These include the characteristics of the fonts used, the colors, type of images, alignment [9, 10, 20].

Finally, the document is exported to a preferred format (such as the popular PDF) and the job is reiterated for another paper.

The processes mentioned above are all located at different layers of the ACCS (Fig. 1).

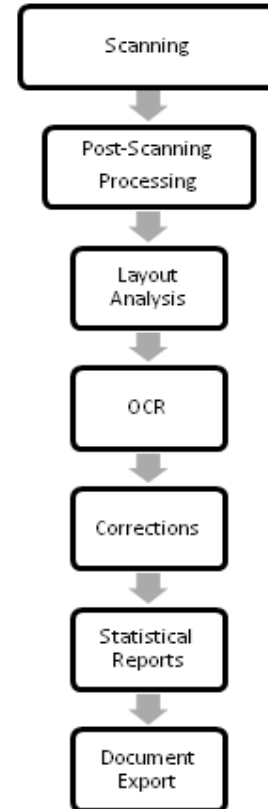


Fig. 1 – ACCS Processing

1.2 Line Detection

Within the Layout Analysis layer the identification of the original form of the document is attempted. To achieve this, different mark elements have to be detected first, such as characters and lines. Hence our Line Detection problem is located at the Layout Analysis layer and can be performed using a software application [1].

Solving this task will bring us closer to our goal of successfully detecting page layout elements, such as paragraphs, columns, images, borders or tables [7, 8].

This Line Detection system will accept as input a binary (black and white) image of a scanned newspaper page and will return a collection of pixel groups, each group representing a vertical or horizontal line from that image. Given that most of the times the input image will contain noise and imperfect lines (erased, dotted, interrupted, and/or crooked), the Line Detection module should also accept a set of parameters which will calibrate the method and filter out undesired lines.

The methods we present here rely on two different mathematical models, but they share the same logic, that of the Hough-transform technique.

The Standard Hough Transform is a well known approach for solving such tasks like detecting analytical shapes in an image space [11]. It is mostly used in edge detection applications as a final processing step. However, for the particular purpose of a digitization system, this method has a number of disadvantages, thus an improved method is sought and found – the Discrete Line Parameterization Method.

2 The Hough Transform

The Hough transform is often used in computer vision or image processing field to find imperfect instances of geometrical shapes (lines, circles, etc) [15, 16, 21]. These shapes are found by carrying out a voting process on a parameter space. In this space, each point corresponds to a distinct object. At the end, local maxima identify the most probable shapes. For example, a straight line $y = mx + b$ can be parameterized by the point (m, b) . We iterate over all pixels in the input image and through all allowed values for one of the parameters. Having assigned these values, we can solve for the second parameter and then cast a vote for this line in the parameter space (also called the accumulator).

The drawback to this approach is that vertical lines give rise to unbounded values for the parameters. For computational reasons, it is more

practical to parameterize the line in terms of its polar coordinates ρ and θ [16,17,18].

The parameter ρ represents the distance between the line and the origin while θ represents the angle of the vector from the origin to the closest point on the line (Fig. 2). Using these parameters the equation of a straight line becomes:

$$\rho = y \sin \theta + x \cos \theta$$

The idea of the algorithm is to iterate over all pixels in the input image. For a pixel located at (x, y) iterate over all values of the θ parameter (i.e. look for all allowed orientations of a line). Obviously, the iteration can only be performed on a discrete set of angles, thus a careful examination is needed to establish which angles are required and what the discrete step should be. Typical implementation sets a window of 20° around the normal (horizontal or vertical) orientation.

Once the three parameters (x, y, θ) have been established, the remaining parameter ρ can be computed.

The final step is to cast a vote into the appropriate parameter space bin. At this step there is again a matter of choosing the right number of bins so as to bring enough accuracy, but not too much computing overhead.

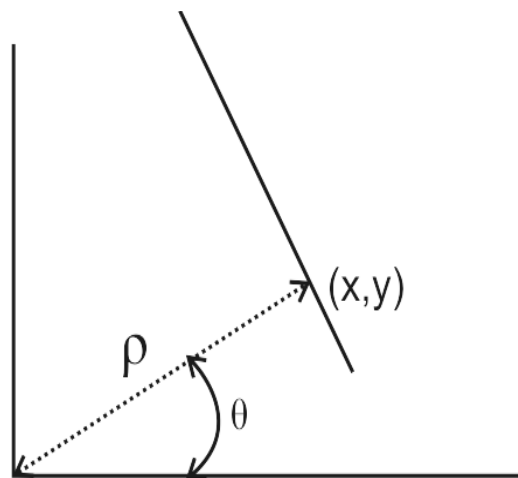


Fig. 2 – Standard Hough Parameterization

We can keep an accumulator to count the number of points that have been reported for the line (ρ, θ) and then look at local maxima to find the best fitting lines [11,12,13,14]. After the voting process is complete, the resulting parameter space can be viewed as a grayscale image where each pixel is brighter if its number of votes is higher (Fig. 4).

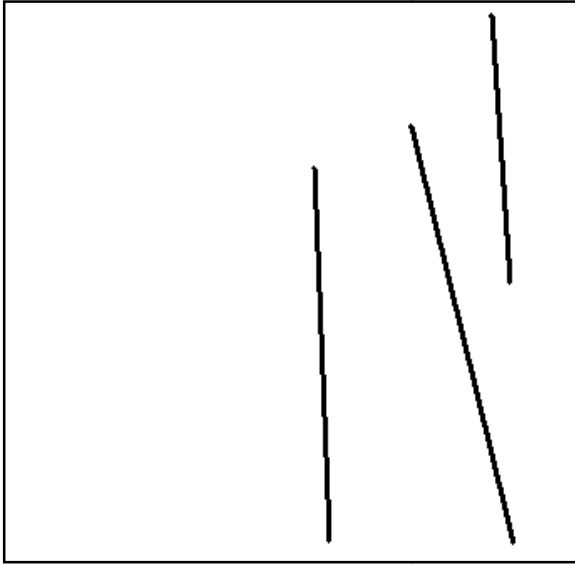


Fig. 3 – Input Image



Fig. 4 – Parameter Space and local maxima

The Hough transform is efficient on random noised images and is able to detect even heavily damaged lines. This is because the mathematical model used is not strictly followed, but rather applied to generate statistical indicators which are interpreted relative to each other and the most probable objects are identified [17,18,19].

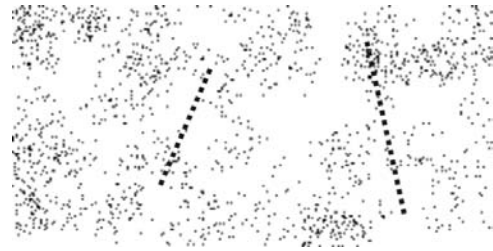


Fig. 5 – Input image



Fig. 6 – Output using standard (ρ, θ) Hough transform

3 Improving the Hough Transform

Several techniques have been invented to improve the Standard Hough Transform and achieve more quality results.

3.1 Randomized Hough Transform

Instead of selecting just one point from the input image and computing the parameters for it, the algorithm can randomly select 2 points and cast a vote for the line that passes through them. This way computing time is reduced and more accuracy is obtained since there is no need to iterate over parameter values because a line is uniquely identified by the two points [4, 22].

3.2 Probabilistic Hough Transform

It can be proved in terms of probability that it is sufficient to select just a fraction of the points from the input image and still achieve comparable results. This greatly increases performance and stability against noise. The fraction of examined points should be determined by a prior scan of the image, but typical values are 10%-15%.

3.3 Generalized Hough Transform

This is a more advanced method which can be used to detect any object in an image, even if that object is not an analytical shape (i.e. it can

be described by a finite number of parameters). It uses an approach called template matching to identify the position of the object in the input image.

4 Problems with the Standard Hough Transform

Most problems occur when processing images with many textual regions which contain several lines of small characters aligned vertically. This causes the Standard Hough Transform to report groups of such aligned characters as dotted lines. Measures which can be taken to prevent this include prior elimination of the characters [6], or statistical examination of the line segment width variation (good lines often have the same thickness, while false lines composed of characters do not).

5 Discrete Line Parameterization

To obtain better results for images with high density of textual content we have devised a new parameterization to be used instead of the standard Hough parameterization. Within text regions, characters are often vertically aligned and resemble a dotted line. The Hough transform detects many false lines this way. The new parameterization takes into account the fact that in a digital image there can be only a restricted set of possible lines. Each line can be represented by two parameters x_0 and dx (Fig.7). It is a close analogy to the previous (ρ, θ) parameterization: x_0 and ρ both reflect offsets while dx and θ represent angles.

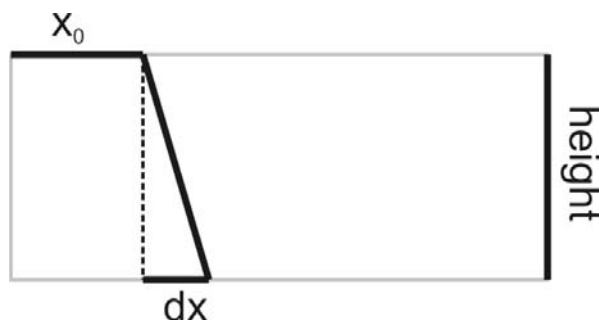


Fig. 7 – A discrete line representation

Based on this we can proceed similarly to Hough transform and iterate over all points in the image and vote the discrete lines that pass through them.

The advantage of this method comes from the

fact that it uses only integer numbers, thus the unknown parameter can be coined more accurately, discarding some of the false lines that were being picked by the more loosely Hough transform method. Also, the new parameters can only account for a small subset of lines. Pixels belonging to characters can rarely pass this restriction and fall into the vote bins.

For this parameterization, the processing stage operates the same way as with the standard one. All pixels are processed iteratively. For each pixel the dx parameter is iterated. Its window of allowed values is determined based on the image resolution. Since dx represents an angle, the bigger the image resolution, the bigger its window has to be, to allow the same set of angles to be iterated for every image type.

After all parameters have been established, the final one is computed (1).

$$x_0 = (x * height - y * dx) / height \quad (1)$$



Fig. 8 – Discrete Parameter Space

Similarly, the parameter space can be represented by a grayscale image where bright pixels symbolize high vote counts and possible lines (Fig. 8).

Through this similarity it can be observed that the two parameter spaces can be processed alike. Furthermore, the possibility of running both parameterizations and comparing the results to achieve greater quality exists.

6 Processing the Parameter Space

For these two methods to achieve best results, a stable method is needed to detect maximum points in the parameter spaces. There are several approaches discussed. Each of them can be applied to both parameterizations. Visual results here are shown only for the Discrete Parameterization.

6.1 Dynamic threshold

A threshold is given an initial value equal to the maximum number of votes present in a point from the parameter space. Then iteratively select the points which pass the threshold and include them in the results. After an iteration the threshold is scaled down by a percentage (70% was used) and the algorithm continues. Also, to prevent duplicates a decimating convolution kernel is applied in the vicinity of the currently selected maxima (Fig. 9).

The problem with this approach is that the stop condition is not stable. The algorithm can be configured to run while the threshold is above a certain absolute value. However, there is not direct correlation between the length of a line and the number of votes it generates in the parameter space. Short and thick lines can yield a large number of votes while long and thin generate few votes. Thus, the algorithm is not stable enough for major practical applications, but it is efficient for small, non-noised images.

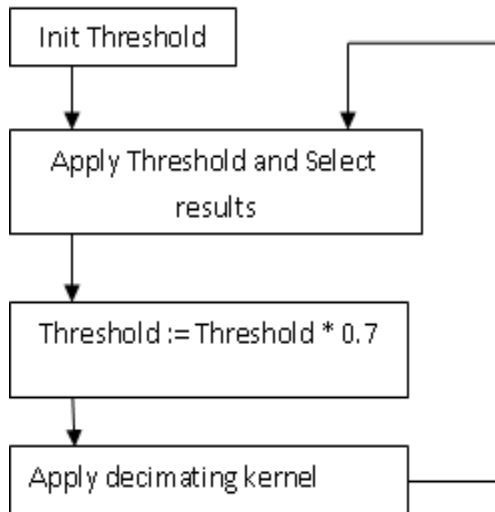


Fig. 9 – Dynamic Threshold

The decimating kernel is very important. Without it, the algorithm would simply be a static threshold and large areas of pixels would get selected instead of singular points like in Fig. 10.



Fig. 10 – Dynamic Threshold Results

6.2 Connected Components

Instead of applying the before-mentioned decimating kernel, a more stable method can be employed. The points that pass the threshold are processed and grouped into clusters. To group them a flood fill algorithm can be used with a small error to account for isolated points. After these connected clusters are formed a line result is extracted from each one. The line parameters inside the clusters are averaged either uniformly or weighted center-wise.

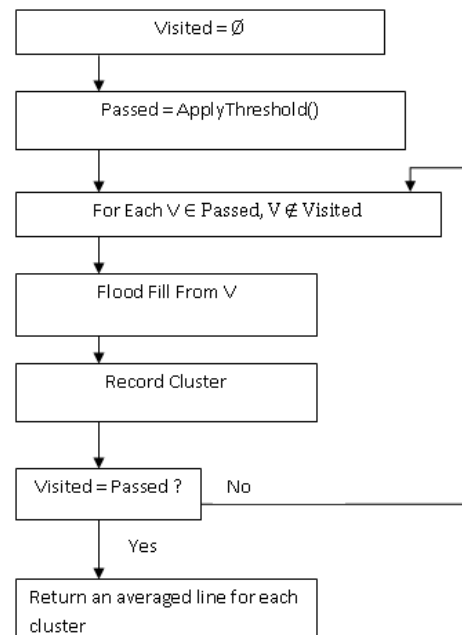


Fig. 11 – Connected Components Algorithm

In the vote space, the maxima are surrounded by areas with higher vote counts corresponding to lines with approximately equal parameter pairs. The Connected Components method identifies these regions, thus preventing duplicates and the necessity for applying a decimating function on neighbor points.

As an improvement for this method, the threshold can be eliminated altogether. The flood fill algorithm can be run to connect points with similar vote magnitudes. From the resulting set of regions, those with small areas are with high probability related to maxima and lines.



Fig. 12 – Connected Components

6.3 Filtering

If the input image contains noise, smoothing the parameter space might provide some extra accuracy. To smooth the voting field a number of data filters can be applied. Some of them are resumed in the following.

6.3.1 Triangle Filter

The triangle filter is good for enhancing local maxima as it gives higher weights to center points (2).

$$T(t) = \begin{cases} 1-|t|, & |t| < 1 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

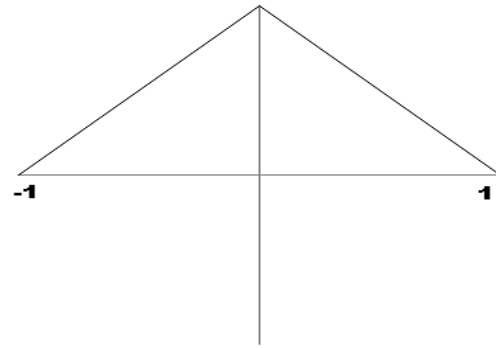


Fig. 13 – Triangle Filter

6.3.2 Gaussian Filter

The discrete Gaussian filter is a smoother version of the triangle filter. It's more time consuming to apply but yields more quality. The kernel below can be used.

$$\frac{1}{273} \begin{pmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{pmatrix}$$

6.3.3 Lanczos Filter

This filter is typically used to scale images as it propagates high frequency values, thus the resulting image will have more quality.

$$L(t) = \begin{cases} \sin c(t) * \sin c\left(\frac{t}{a}\right), & -a < t < a \\ 0, & \text{otherwise} \end{cases}$$

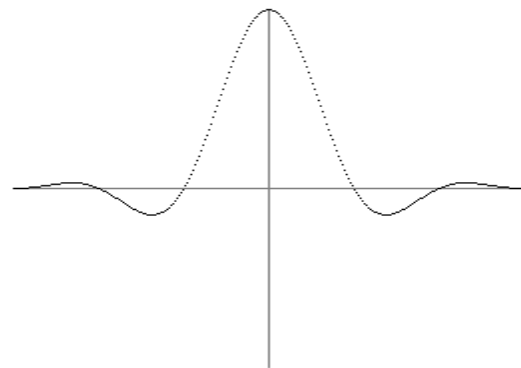


Fig. 14 – Lanczos Filter

6.3.4 B-Spline Filter

This is a general smoothing filter which also gives higher weights to center elements thus accentuating peaks and useful for maxima sweep (Fig. 15).

$$B(t) = \begin{cases} \frac{1}{2}t^3 - t^2 + \frac{2}{3} & , |t| < 1 \\ \frac{1}{6} * (2-t)^3 & , 1 < |t| < 2 \end{cases}$$

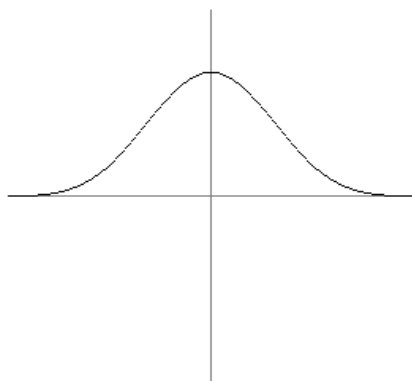


Fig. 15 – B-Spline Filter

6.3.5 Other Filters

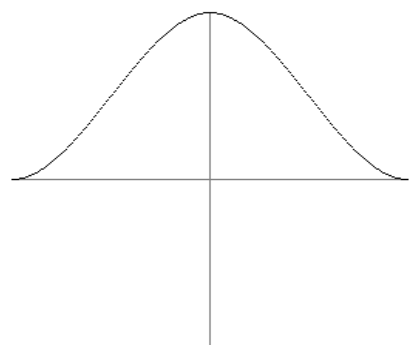


Fig. 16 – Hermite filter

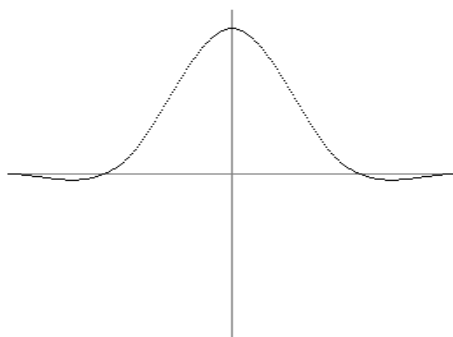


Fig. 17 – Mitchell Filter

The filters can be applied as a preprocessing step after which the Connected Components method can be used, for example.

Smoothing is good for eliminating noise from the parameter space as well as for balancing regions with alternating vote peaks.

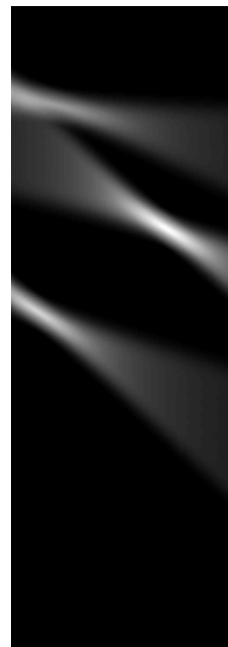


Fig. 18 – Triangle Filtering

7 Results

Both methods were tested on our collection of scanned newspaper pages.

To accommodate a large variety of line types, our implementations of the two methods also accept a set of parameters. The parameters cover targeted line segment properties.

A maximum allowed deviation factor adjusts the window of the iterating parameter (θ for the Hough Transform, dx for the Discrete Line method) and helps the detection of slightly tilted lines. The imperfection can have any cause, but most likely is due to a misalignment from the scanning process.

Another calibrating parameter specifies a maximum distance in pixels which should be allowed as an interruption in the line segment. Line segments that are closer than this number of pixels and have the same orientation are grouped together to form a complete line. This is mostly helpful when detecting dotted or white noised lines.

Other parameters are used to filter out undesired and degenerated lines.

For each text, the parameters were adjusted to

return the best possible results for the tested method.

The results were visually interpreted and a comparison was made for different types of images between the two methods.

The implemented method consists of a filtered Connected Components method combined with the Probabilistic Hough Transform improvement.



Fig. 19 - Original Image

The image shown in Fig. 19 is of a typical paragraph, often encountered as input. The text is left aligned and the font is normal. These cause the Hough transform method to form a line entirely of characters, an issue that rarely affects the Discrete Line method.



Fig. 20 – Standard Hough Results



Fig. 21 – Discrete Parameterization Results

The Discrete Parameterization has successfully extracted only the correct lines, leaving the characters out.

Vorhersage der Temperaturen in °C und

	heute	morgen
Amsterdam	23 ne 0	23 ne 0
Bangkok	37 bw 0	35 re 12
Barcelona	26 he 0	25 bw 0
Beograd	18 re 1	21 bw 0
Brussel	24 he 0	24 he 0
Budapest	15 rs 2	19 ge 4
Caracas	25 he 0	26 bw 0
Casablanca	22 re 5	21 re 3
Danzig	18 bw 0	18 he 0
Havana	33 he 0	34 sd 0
Helsinki	22 bw 0	20 bw 0
Hongkong	30 re 13	29 ge 24
Kapstadt	22 re 28	18 he 0
Kopenhagen	21 he 0	18 bw 0
Kuala Lumpur	32 rs 3	32 rs 6
Lissabon	24 he 0	22 bw 0
Los Angeles	20 wf 0	19 bd 0
Luxemburg	23 wf 0	23 he 0
Mailand	24 bw 0	26 bw 0
Malta	24 wf 0	19 wf 0
Mauritius	27 re 5	26 wf 0
Miami	31 wf 0	31 w 0

Montag, 18. Mai 1998 -


Tierkreis:  SA: 5.27
Stier SU: 21.11

Fig. 22 - Original Image

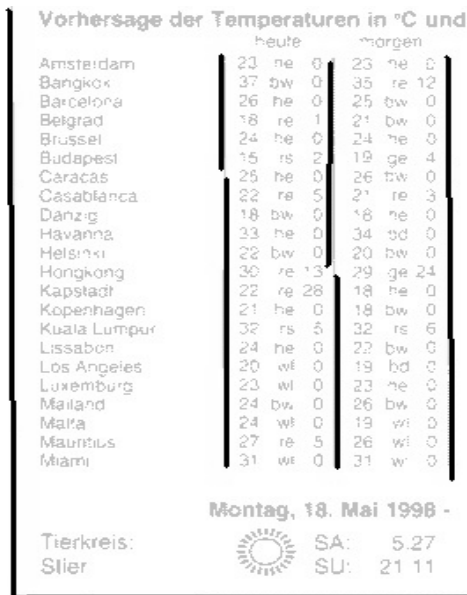


Fig. 23 - Standard Hough Transform Results

This second image underlines another disadvantage of the Hough transform method.

Since it works with real valued parameters and covers the real set of lines, the Hough transform can't tolerate the imperfections of a digital line, which is actually formed by small parallel line segments (similar to how Bresenham drawing algorithm works). It can be seen in Fig.23 that this causes the method to split lines in multiple segments.

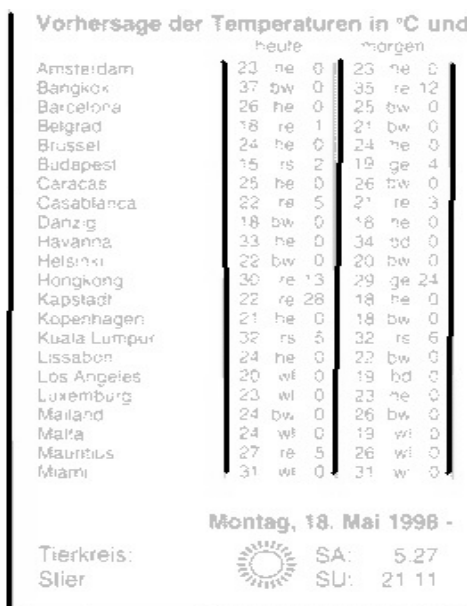


Fig. 24 - Discrete Line Method Results

One of the purposes of the Line Detection module is seen in Fig. 24 where successfully detected lines are helpful in separating table columns, otherwise impossible to identify.

8 Performance

Both methods operate in optimal linear time, proportional to the resolution of the scanned image. The memory consumption is also linear. Different improvements can be made to achieve even more efficient computation times for large images (e.g. Probabilistic Hough Transform or Randomized Hough Transformed), [5].

9 Conclusions and Future Work

This paper attempted to cover one general approach to Line Detection, i.e. the Hough Transform. In its generality, the Hough Transform leaves room for further improvements, which were also discussed. A new method based on voting spaces was illustrated – The Discrete Line Parameterization. Also, new ways of processing the parameter space were introduced – Filters, Connected Components.

Together, these can be successfully integrated into a Line Detection subsystem which can be adjusted to take into account different types of lines, images, noise levels, etc.

The alternative parameterization presented here can be used with the same logic adopted for the standard Hough transform technique. This new parameterization has the advantage that works with whole numbers, thus making calculations simpler and restricting the parameters to digital allowed values. Our tests show that this method is better suited for processing binary images which also contain large quantities of text.

Future enhancements of these methods are possible and needed. A Line Detection module will take as input a set of calibrating parameters. One enhancement could be to allow fast processing after a parameter has changed its value. This could be useful in correlation with a human operator to achieve real-time visual results and enhance its own performance when correcting documents.

The data structure currently used to store a parameter space is a typical 2D array. A study of other data structures which would make the algorithm more efficient can be carried out. For example, due to the fact that the voting space is discrete, it causes a lot of precision loss. Sparse matrices might be able to solve this by concentrating

the data into critical regions, i.e. regions with high vote counts.

A learning system could be sought. Such a system should be able to give an approximated set of parameters and make its own adjustments as the processing is carried out.

Lastly, the methods presented here can be studied with other tasks, like character or image detection.

Line Detection is vital to a complete Automatic Content Conversion System. Its role is perfectly integrated in the Layout Analysis stage, where, by joining information with other modules the complete layout of the page may be obtained.

The methods presented in this paper including the newly introduced Discrete Line Parameterization can be easily implemented into an application and used. After a proper calibration of the algorithms, the Line Detection module is ready to be assembled and initialized.

References:

- [1] Costin-Anton Boiangiu, Bogdan Raducanu, Robust line detection methods, *Proc. 9th WSEAS International Conference on Automation and Information*, 2008
- [2] Costin-Anton Boiangiu, Andrei Spataru, Dan-Cristian Cananau, Automatic text clustering and classification based on font geometrical characteristics, *Proc. 9th WSEAS International Conference on Automation and Information*, 2008
- [3] Costin-Anton Boiangiu, Bogdan Raducanu, Entity clustering using 3D mesh simplification, *Proc. 9th WSEAS International Conference On Automation and Information*, 2008
- [4] Kwangsoo Hahn, Youngjoon Hahn And Hernsoo Hahn, Ellipse detection using a randomized Hough transform based on edge segment merging scheme, *Proc. 6th WSEAS International Conference on Signal Processing, Robotics and Automation*, 2007
- [5] M. Fikret Ercan and Yu-Fai Fung, Computation of the Hough Transform using SIMD extensions, *WSEAS Int. Conf. On Electronics, Control & Signal Processing*, 2002
- [6] S. Kahlouche, K. Achour and M. Benkhelif, A new approach to image segmentation using genetic algorithm with mathematical morphology, *WSEAS Int. Conf. on Electronics, Control & Signal Processing*, 2002
- [7] Costin-Anton Boiangiu, *Multimedia Techniques*, Macarie, 2002
- [8] Costin-Anton Boiangiu, *Elements Of Virtual Reality*, Macarie, 2002
- [9] Costin-Anton Boiangiu, The “Beta-Shape” Algorithm for polygonal contour reconstruction, *The 14th International Conference on Control System and Computer Science*, C.6, Vol. II, 2003
- [10] Serban Petrescu, Zoea Racovita, Florica Moldoveanu, Costin-Anton Boiangiu, Alin Moldoveanu, Gabriel Hera, Neuron GIS solutions for the optimal path selection, *The 11th International Conference on Control System and Computer Science*, 11.10, Vol. II, 1997
- [11] R. O. Duda and P. E. Hart, Use of the Hough transformation to detect lines and curves in pictures, *Comm. ACM*, Vol. 15, 1972, pp. 11–15
- [12] W. E. L. Grimson And D. P. Huttenlocher, On the sensitivity of the Hough transform for object recognition, *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 12, 1990, pp. 255–274
- [13] J. Illingworth And J. Kittler, The adaptive Hough transform, *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 9, 1987, pp. 690–698
- [14] J. Illingworth And J. Kittler, A survey of the Hough transform, *Comput. Vision Graphics Image Process.*, 44, 1988, pp. 87–116
- [15] W. Niblack And D. Petkovic, On improving the accuracy of the Hough Transform, *Mach. Vision Appl.*, Vol. 3, 1990, pp. 87–106
- [16] J. Princen, J. Illingworth, And J. Kittler, A formal definition of the Hough Transform: Properties and relationships, *J. Math. Imaging Vision 1*, 1992, pp. 153–168
- [17] T. Risse, Hough Transform for line recognition: complexity of evidence accumulation and cluster detection, *Comput. Vision Graphics Image Process.*, 46, 1989, pp. 327–345
- [18] P.V.C. Hough, *Method and means for recognizing complex patterns*, 1962
- [19] J. Sklansky, On the Hough technique for curve detection, *TC Journal*, Vol. 27, 1978
- [20] B. Chen, and L. He, “Fuzzy template matching for printing character inspection”, *WSEAS Transactions on Circuits and Systems*, Issue 3, Vol. 3, 2004
- [21] Thouraya Merazi Meksén, Malika Boudraa, Redouane Draï, Automatic Detection of Cracks using the Hough Transform on Sparse Matrix, *WSEAS Transactions on Signal Processing*, Issue 8, Volume 2, August 2006
- [22] M.I. Rajab., “Feature Extraction of Epiluminescence Microscopic Images by Iterative Segmentation Algorithm”, *WSEAS Transactions on Information Science and Applications*, Issue 8, Vol. 2, 2005