A LOCAL APPROACH FOR FAST LINE DETECTION

S. Lefèvre, C. Dixon, C. Jeusse, N. Vincent*

Laboratoire d'Informatique
E3i / Université de Tours
64 avenue Portalis
37200 Tours - FRANCE
{lefevre,vincent}@univ-tours.fr

Abstract: In this paper we are concerned with line detection for scene modelling. So we focus on horizontal and vertical lines and we propose a fast local approach to detect lines in binary images. The approach described here can easily be extended to lines characterized by other directions. Pixels are analysed using an accumulator on a block-based basis in order to obtain possible line segments for each block. As the results obtained rely only on a local analysis, their coherence can be further improved using a postprocessing step. The proposed method has been compared to classical approaches in order to show its efficiency.

1. INTRODUCTION

Video understanding relies on the analysis of the different objects composing the scene and of their relative position at a given time. These moving objects can be tracked using some adequate methods. In order to understand a video sequence, it is necessary to reposition these objects within the scene or background model. So we are looking for generical tools for background structure extraction. As we are dealing with video processing, we have also to take into account the real time constraint. Among features we can use for background structure modelling, we choose to focus on discontinuities and more precisely on lines rather than points or edges. Indeed vertical and horizontal lines are of great interest when dealing with artificial (or human built) environments.

The line detection process can be performed using local approaches or global ones, as the well-known Hough Transform [1] and its derivated [2]. As the computation time is a strong constraint to respect in our case, we propose to perform line detection on a local basis, instead of a global one. Only few local approaches (as [3]) were proposed in litterature with regards to global ones.

We consider that edge detection has already been performed and resulting binary image I has been obtained. Image I is then divided into disjoint blocks which are analysed to determine the presence or not of a line segment. This analysis relies on the use of a local accumulator. We consider here only horizontal and vertical directions for line detection but the proposed method can be easily adapted to other directions. As the proposed method is based on a local analysis, we have included a second step to ensure coherence between neighbouring line segments as a postprocessing step.

We will first recall some classical methods that we will compare our method with. We then present the block-based approach for line segment detection we proposed here. Both steps will be presented, first how an accumulator can classify line segments according to their directions, and secondly the postprocessing step for coherence enhancement. Finally we will compare results obtained with the classical approaches from first section and proposed approach from following sections in order to show the quality of our contribution.

2. CLASSICAL APPROACHES FOR LINE DETECTION

Line detection is a classical image processing problem and has been studied for many years. In this section we detail several methods presented in [4]. These methods will be compared with the approach we proposed here in section 5 of this paper. It will help us to show the efficiency of our contribution.

In order to detect lines present in a image, it is possible to process directly the input greylevel image using some convolution masks. The first approach called "Edge linking" relies on gradient computation using some adequate mask (*e.g.* Sobel operator). Once the gradient has been computed, every pixel is further analysed as follows: a pixel is kept if its gradient magnitude is higher than a given threshold (*i.e.* it is an edge point) but also if there exists a pixel in its neighbourhood which is characterized by similar gradient magnitude and direction values.

Another approach consists in the use of orthogonal masks introduced by Frei and Chen in [5]. Depending on the result of the convolution, every pixel can be classified into one of the three following classes: edge, line, or average. Lines are then obtained by keeping only pixels belonging to the class "line".

When focusing on horizontal and vertical lines, it is possible to use some dedicated unidirectional masks and then to combine results obtained with these masks.

Instead of processing directly greylevel images, some approaches are taken as input binary edge images. Among them, Hough Transform is probably the most well-known [1]. In this method, every edge pixel votes for a set of lines represented in a parametric space (*e.g.* the polar representation space using ρ and θ). Votes are stored in an accumulator array where each value is linked with a given physical line. Finally lines linked with maximum values are kept.

In this section we have presented a few line detection methods we will compare our proposed method with. This method will be explained in the following section.

3. BLOCK-BASED LINE SEGMENT DETECTION

Let us recall we are concerned with line detection for scene modelling in video sequences. In this case, execution time is a constraint which has to be taken into account. So we propose to analyse the input image on a local basis instead of a global one. Local processing is often characterized by lower computational cost due to its easier possible parallel implementation. We have chosen to process binary edge input images instead of original greylevel images. So edge detection has to be performed as a preprocessing of our method.

The image is then viewed as a set of blocks composed of $W \times H$ pixels. On each of the blocks our method will be applied. It is based on a local accumulator used to detect and classify line segments into horizontal and vertical directions. Decision relies on a vote procedure. In this section, we will describe the different steps of this approach, which are accumulator representation, computation, and analysis.

For every block, a local accumulator is created and then analysed to determine the presence and the orientation of a possible line segment. We propose to divide the blocks themselves into several regions R_k to detect horizontal and vertical lines. More precisely, M (respectively N) regions are created for detection of horizontal (respectively vertical) lines. Regions considered can either be disjoint or overlapping. This is illustrated in figure 1 where M and N are both equal to 3, resulting in 3 overlapping regions in both directions. So in the block we consider $M + N$ regions R_k with $k \in [1, M + N]$. The accumulator A is linked with these regions and contains one counter per region. So it is also composed of $M + N$ counters $\{A_1, \ldots, A_{M+N}\}$.

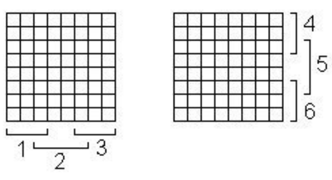


Fig. 1. Spatial representation of regions R_k corresponding to (a) vertical lines and (b) horizontal lines.

The accumulator will be used to store information from image data. Similarly to Hough Transform, each pixel (x, y) of the block B characterized by a non zero value (*i.e.* representing an edge point) votes for a particular feature (here at least two regions R_k of the block B, one among horizontal regions and one among vertical ones) associated with a counter in the accumulator. This vote simply depends on the spatial position of the pixel and results in an increment of the associated counter in the accumulator, as shown by:

$$A_k \leftarrow A_k + 1 \text{ if } (x, y) \in R_k \tag{1}$$

The values in the accumulator are normalised with respect to each region area.

The last step is the analysis of the accumulator counters once every pixel of the block B has been scanned. The highest counter A_{max} of the accumulator is computed. If this value is lower than a threshold T, we consider the number of votes for the associated region as unsufficient and assume no line has been deteced in this block. Otherwise the line associated with A_{max} is kept.

We are here considering only horizontal and vertical lines which are of most importance for scene modelling, particularly when the nature of background is artifical (*i.e.* human built). However the method can easily be extended to other line directions by defining some other local areas R_k for every block.

The approach described in this section relies on a local analysis of pixel values using a set of parameters $\Omega = \{W, H, M, N, T\}$. So the coherence of results obtained in connected blocks is not ensured. This problem can be solved using a filtering considered as a postprocessing step. This step will be detailled in next section.

4. FILTERING

The local approach proposed in previous section is relying on analysis of blocks of $W \times H$ pixels where W and H are small regarding the size of the input image. So it does not ensure coherence of line segments detected in neighbouring blocks. The filtering step described here helps to increase robustness of the proposed approach.

Every block containing a line segment is compared to its neighbours to determine if the direction of the detected line is coherent with the neighbouring blocks. More precisely, we analyse pairs of blocks B_n built by taken two neighbouring blocks of B which are symmetrical with respect to B. Figure 2 shows an example of pairs of neighbouring blocks B_1B_5, B_2B_6 and B_3B_7 in the case of an horizontal line segment has been detected in B. More advanced filters can be designed using this procedure in an extended neighbourhood of B.

B_1	B_8	B_7
B_2	B	B_6
B_3	B_4	B_5

Fig. 2. Pairs of neighbouring blocks B_1B_5, B_2B_6, and B_3B_7 used in filtering of a block containing horizontal line.

The filtering step described here allows to increase coherence of line segments detected by our previous approach. We will now see the efficiency of the proposed approach compared to classical ones.

5. RESULTS

We have tested the local approach proposed in this paper on binary images obtained by applying an edge detection operator (currently the Sobel operator with thresholding) on greylevel images, as shown in figure 3.

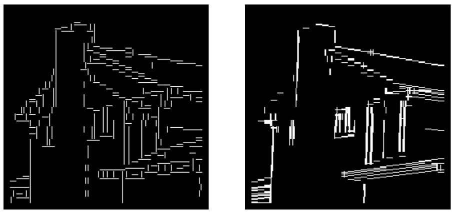
Fig. 3. Original greylevel image (left) and binary edge image after thresholding result of Sobel operator (right).

In the results presented below, we will use two sets of parameters precised in table 1. These parameters do not depend on the kind of image analysed but rather on the kind of result required. Parameters (W, M) and (H, N) are choosen depending on the required position precision and directional tolerance of the line segments. T is a tolerance threshold linked to the noise accepted in the line segment. When no tolerance is assumed, the Ω_2 set has to be choosen. Indeed it is particularly adapted in situations where the result should be as precise as possible and the line segments present in the input image have to be complete (*i.e.* no pixel is missing). In this case there is no overlappings among horizontal or vertical regions. The M, N, T parameters can be decreased in order the process to be more tolerant, resulting in the set Ω_1. The results presented use Ω_1 and Ω_2 sets of parameters. Ω_1 is associated with the covering of the blocks shown in figure 1. In this case we consider overlapping regions.

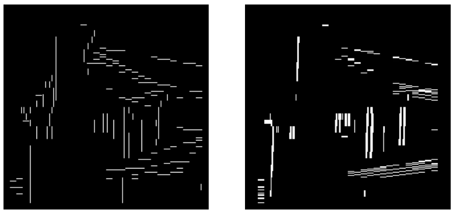
Parameter	Description	Ω_1	Ω_2
W	Block width	8	8
H	Block height	8	8
M	Number of horizontal areas	3	8
N	Number of vertical areas	3	8
T	Number of votes required	5	8

Table 1. Description of the two sets of parameters used to obtain results presented in this paper.

Figure 4 shows results of the proposed approach using the two different sets of parameters Ω_1 and Ω_2 without performing the coherence verification step. We can verify that using Ω_1 will return more line segments but the spatial position of these segments is less precise than using the other set of parameters. On the contrary, when processing images with Ω_2, precision of the results is increased but the number of line segments is limited. So depending on the context of the application, Ω_1 or Ω_2 will be more relevant. We can see on top line of figure 4 that coherence of line segments detected in neighbouring blocks is not ensured. So it can be necessary to apply the filtering step for coherence verification we have described in section 4. Bottom line of figure 4 shows results of the proposed method using the two sets of parameters Ω_1 and Ω_2 when considering the filtering step described in figure 2 for coherence verification.



Ω_1 without filtering Ω_2 without filtering



Ω_1 with filtering Ω_2 with filtering

Fig. 4. Results of proposed method without and with coherence verification and using Ω_1 and Ω_2.

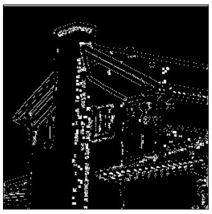
In order to evaluate the quality of our contribution, we have compared the proposed approach with some classical methods described in section 2. Of course our approach could be compared with a more complete panel of methods from the litterature, either local [3] or global [2] ones. Figure 5 shows results of line detection using edge linking method, Frei and Chen orthogonal masks, and finally combination of horizontal and vertical masks. As illustrated in figure 5, the compression of information is not great enough to achieve an easy understanding step of the scene. Indeed a lot of isolated edge pixels are kept and resulting image do not consists only on a set of line segments, which is the necessary result for correct scene modelling based on line detection.

We have also compared our contribution with Hough Transform. It is of great importance as the proposed approach can be seen as a fast and local adaptation of the Hough Transform. Figure 6 shows the binary edge image obtained with sobel operator and used as input in the line detection process. Result of Hough Transform using XHoughTool is also given in this figure as presented in [6]. Lines are correctly detected but we can see that two vertical lines are missing.

Figure 7 shows results obtained using Ω_1 and Ω_2. Contrary to results obtained with Hough Transform, it is possible to keep all line segments present in the input image when performing the line detection with a certain tolerance (using Ω_1). Ω_2 is particularly useful when dealing with modelling of artificial scenes which are most of the time composed of vertical lines.

6. CONCLUSION

In this paper, we are concerned with fast line detection for scene modelling. Our contribution consists in a fast

(a) (b)

(c)

Fig. 5. Results of edge linking method (a), use of orthogonal masks from Frei and Chen (b), and combination of horizontal and vertical masks (c).

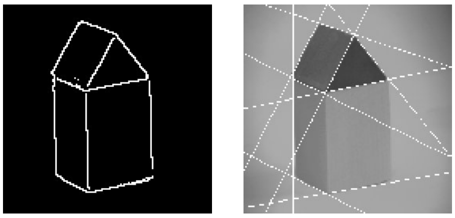
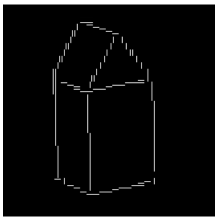
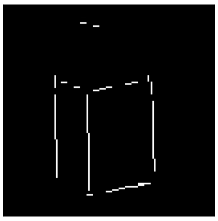
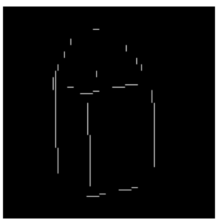
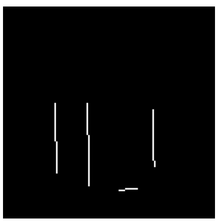


Fig. 6. Original edge image (left) and result of Hough Transform (right) using XHoughTool as presented in [6].

Ω_1 without filtering Ω_2 without filtering

Ω_1 with filtering Ω_2 with filtering

Fig. 7. Results of proposed method without and with coherence verification and using Ω_1 and Ω_2.

and local approach for line detection in binary edge images. Detection processing is performed on a block-based basis. Several sets of parameters can be used depending not on the kind of image processed but rather on the kind of result required. Indeed, sometimes it is necessary to obtain results of great precision whereas in other cases it could be interesting to process input image with a certain tolerance (when line segments present in the input image are incomplete). To avoid the consequences of a too local approach we have introduced a more global step that allows to increase the coherence of results obtained for neighbouring blocks. Indeed we propose a filtering used as a postprocessing step for coherence verification. In order to qualify our contribution, we have compared the proposed approach with some classical ones, as edge linking method, classification based on orthogonal Frei and Chen masks, combination of horizontal and vertical masks and finally Hough transform. Results obtained show our method gives results more adapted to further scene modelling. Due to its low computation time, the proposed method can be more easily integrated in a video analysis system than classical approaches.

Future work will include extension of the proposed method to detect other kinds of local shapes by defining some different areas. We also consider implementation of the described approach on a multiprocessor workstation in order to limit computation time, which will allows this method to be included in a real time video analysis tool. We are now developping an approach based on greylevel or colour images without performing edge detection.

REFERENCES

[1] P. V. C. Hough, “Methods and means for recognizing complex patterns,” *U.S. Patent*, vol. 3069654, December 1962.

[2] A. A. Kassim, T. Tan, and K. H. Tan, “A comparative study of efficient generalised hough transform techniques,” *Image and Vision Computing*, vol. 17, no. 10, pp. 737–748, August 1999.

[3] R. Shpilman and V. Brailovsky, “Fast and robust techniques for detecting straight line segments using local models,” *Pattern Recognition Letters*, vol. 20, pp. 865–877, 1999.

[4] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, 1992.

[5] W. Frei and C. Chen, “Fast boundary detection: A generalization and a new algorithm,” *IEEE Transactions on Computer*, vol. 26, no. 10, pp. 988–998, October 1977.

[6] H. Kalviainen, P. Hirvonen, and E. Oja, “Houghtool – a software package for the use of the hough transform,” *Pattern Recognition Letters*, vol. 17, no. 8, pp. 889–897, 1996.