NETLines: Recovering Line-Networks via Gradient-Based Line Segments Refinement

Xiaohu Lu, Jian Yao†, Kai Li, Li Li, Kao Zhang, and Jinge Tu
School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, Hubei, P.R. China
†Email: jian.yao@whu.edu.cn Web: http://cvrs.whu.edu.cn/

***Abstract*— In this paper, we propose a novel line segment detector, named as NETLines, which can produce a set of accurate line segments and a set of node-connected line-networks formed by connection of the line segments and the image boundary. Based on the line segments generated by other line segment detectors (e.g., EDLines [1]) on an edge map, the proposed algorithm efficiently makes use of the gradient map of the original image instead of its edge map to extend and refine the line segments. The line-networks are constructed with the extended and refined line segments and a set of nodes generated by connecting of the line segments. Furthermore, the line segments and line-networks are optimally supplemented and refined by linking and merging the line segments. Experimental results on a set of natural images illustrate the proposed NETLines produces more accurate and complete line segments compared with state-of-the-art line segments detectors.**

***Index Terms*— Line Segments Detection, Edge Detection, Line-Networks, Line Segments Refinement**

I. Introduction

Line segments detection and line-networks recovering are both well-known computer vision issues. The well-established way is to first detect the line segments and then recover the line-networks of an image. However, these two stages can be integrated into a single one. On the one hand, the intersected line segments can determine the locations of the nodes in the line-networks; on the other hand, the nodes can provide constraints on line segments detection, the result of which is certainly more robust. The line-networks comprised of line segments and connected nodes, contain important geometric information of an image, especially when the scene of the image is consisted of many man-made objects. Besides, line-networks can be used as low-level features to assist to solve problems such as stereo matching [2], [3], indoor scene layout recovering [4], road extraction [5], crack detection in materials, image compression, and so on.

In general, line segments detection can be divided into two categories: gradient-magnitudes-based and gradient-orientations-based. The most widely used gradient-magnitudes-based edge detection approach is the Canny edge detector [6], which calculates the gradients of pixels with the Sobel operator followed by the non-maximum suppression and edge tracking. Line segments can then be extracted from the edge detection result. Hough transform (HT) [7] is a traditional line detector based on an edge map, which extracts all lines containing a number of edge points exceeding a threshold. A lot of variants of the Hough transform have been proposed, e.g., the elliptical Gaussian kernel-based Hough transform [8], but they all require a binary edge map as input and extract infinitely long lines instead of line segments and easily cause many false detections in richly textured regions with strong edges. In order to overcome these shortcomings, Akinlar and Topal [1] propose a robust and efficient line detector, named as EDLines, which can be divided into three steps: (1) extracting the edge map by the Edge Drawing (ED) algorithm [9]; (2) extracting line segments from the edge chains based on Least-Squares line fitting method; (3) eliminating false line segments according to the Helmholtz principle [10], [11].

The idea of detecting line segments based on the gradient orientations was firstly proposed by Burns et al. [12] whose approach only depends on gradient orientations. In contrast to classic edge detectors, their proposed approach defines a line segment as a straight image region whose points share roughly the same image gradient orientation. A recently proposed line segment detector (LSD) [13], [14] which produces accurate line segments and controls the number of false detections in a low level by efficiently combing gradient orientations and the line validation according to the Helmholtz principle. Although LSD has a good performance on most types of images, it's a bit time-consuming which makes it unsuitable for real-time applications.

Both of these two kinds of approaches extract line segments directly from edge map without the consideration of additional information of the original image. If the gradient or the orientation of a pixel on a line segment is not salient enough to be extracted as an edge pixel, it is unlikely to be detected out using existing methods. To solve this problem, we propose a novel line segments detector, named as NETLines, which extends and refines the line segments based on the gradient map and builds the line-networks through linking and merging line segments. Given an image and the corresponding detected line segments, we generate line-networks through the following five steps. First, we calculate the gradient map of the image. Second, attempts are made to extend the line segments according to the gradient map,

whenever a line segment is extended it is refined based on the gradient map. The extension and refinement procedures are conducted iteratively until the terminal conditions are met. Third, we try to generate additional line segments by linking one terminal point of a line segment to another adjacent terminal point of other line segments around. Fourth, line segment pairs with close orientations but weak gradient connections neglected by the extension step are linked together to form a single line segment to complete the line-networks. Fifth, the line segments with close orientations connected by nodes are merged to obtain longer line segments. The result of our method is a set of line-networks with lines and nodes connecting with each other. Fig. 1 is a demonstration of these five steps.

II. Algorithm

A. Preparation

Before all the operations, a preparation process is conducted to get some necessary informations about the input line segments, including the gradient, the stability and the rough trace of each line segment.

The gradient of a line segment is the sum of the gradient' parallel component of all the points on it to the direction of the line segment, which is calculated as follow:

$$G(\mathbf{l}_i) = \begin{cases} G(\mathbf{l}_i) + Sig(g(\mathbf{x})) \times \sin\theta, & \text{if } |\theta| < th_\theta \\ & \text{or } |\theta| > 2\pi - th_\theta \\ G(\mathbf{l}_i), & \text{otherwise} \end{cases} \tag{1}$$

where $G(\mathbf{l}_i)$ denotes the gradient of a line segment, Sig is a Sigmoid function which normalizes the gradient of point $\mathbf{x}$, g($\mathbf{x}$), into $[0, 1]$, $\theta = \theta(\mathbf{x}) - \theta(\mathbf{l}_i)$ is the deviation between the orientation of point $\mathbf{x}$, $\theta(\mathbf{x})$, and the orientation of line $\mathbf{l}_i$, $\theta(\mathbf{l}_i)$, th_θ denotes the deviation threshold between $\theta(\mathbf{x})$ and $\theta(\mathbf{l}_i)$.

The stability of a line segment is the normalization value of the line segment's gradient. Based on the observation that the greater the gradient of a line segment is the more stable it is, we consider the gradient of a line segment can reflects its stability in a way. However, the gradient values of line segments range from zero to thousands. Thus a normalization of the gradient is followed by applying a Sigmoid function. The more stable a line segment is, the harder it can be changed.

The trace of a line segment is recorded on an integer pixel binary map which writes down the number of each line segment on a pixel when the line segment is crossing this pixel. Whenever a line segment is changed, the binary map is rewrote.

B. Extending

Given line segments and the gradient map of the image, the extending is conducting in three steps. First, we sort all the line segments into three clusters: zero-node line segments, one-node line segments and two-nodes line segments. The extension begins with the one-node line segments from the longest to the shortest, and then is the zero-node line segments which extend in two directions. Second, for each line segment to be extended, we search a front area to judge if the line segment can be extended. Third, the line segment is refined based on the gradient map if it can be extended.

Fig. 2. Searching step and span: The searching steps of the line segment $\mathbf{l}_1$ is set as 1/10 of its length, and the searching direction of $\mathbf{l}_1$ is converted into the vertical.

1) Searching: The extension begins with searching an area in front of the line segment that is the searching process. The length of this area, that is the searching steps, is set as 1/10 of the line segment's length; the width, that is the searching span, which is orthogonal to the direction of the line segment, is set as fixed value for all the line segments. We convert the searching direction into the vertical when the absolute value of the line segment's slope is less than 1 and horizontal directions otherwise for convenience. Fig 2 is an example of searching steps and span of a line segment.

Searching is conducted in three steps. First, we search for other line segments in the area based on the binary map, if the current line segment is extended to the edge of the image, the searching is stopped and a node is added here. Second, for each detected line segment, if the orientation deviation between it and the current line segment is smaller than a certain limitation, we try to merge these two and calculate the gradient of the merged line segment, if the gradient of the merged line segment is greater than 80% of the sum gradient of these two line segments, the detected line segment is accepted as an extending hypothesis. If the orientation deviation is so greater, we calculate the intersection point of these two line segments, if the intersection point is within the searching steps, we added an extending hypothesis. Third, we choose the closest extending-hypothesis as the best-extending-hypothesis, if the detected line segment of the best-extending-hypothesis has more than one node, we then judge if the current line segment can be added into the closer node according to the stability of the node. The stability of a node is defined as follow:

$$S(\mathbf{n}_i) = F(S(\mathbf{l}_i), S(\mathbf{l}_j), \theta) \tag{2}$$

where $S(\mathbf{n}_i)$ is the stability of node $\mathbf{n}_i$, $S(\mathbf{l}_i), S(\mathbf{l}_j)$ are the

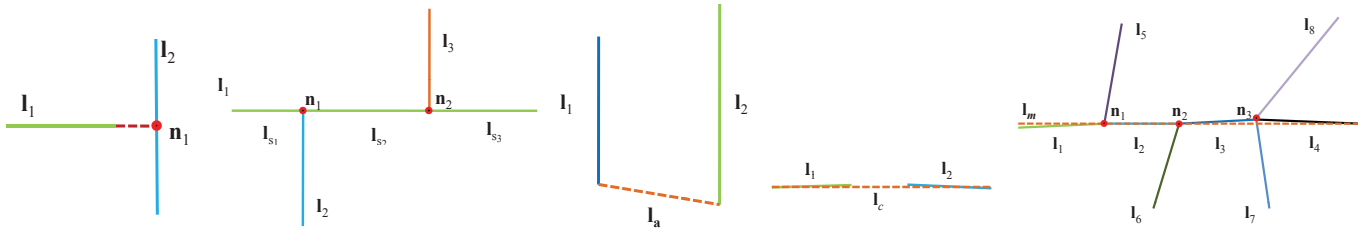

Fig. 1. Five steps of our method, from left to right are: (1) extend line segment $\mathbf{l}_1$ to $\mathbf{l}_2$; (2) split line segments $\mathbf{l}_1$ on nodes $\mathbf{n}_1$ and $\mathbf{n}_1$; (3) add line segment $\mathbf{l}_a$ between $\mathbf{l}_1$ and $\mathbf{l}_2$; (4) link line segment $\mathbf{l}_1$ to $\mathbf{l}_2$; (5) merge line segment $\mathbf{l}_1$, $\mathbf{l}_2$, $\mathbf{l}_3$ and $\mathbf{l}_4$ to form a longer line segment $\mathbf{l}_m$.

stabilities of the two line segments $\mathbf{l}_i$ and $\mathbf{l}_j$ which form the node, and θ denotes the orientation deviation of $\mathbf{l}_i$ and $\mathbf{l}_j$. The greater the stability, the harder the current line segment can be added into the node. If the current line segment is added to a node, a net-refinement process is conducted to adjust the node to the best position on which the sum gradient of all the line segments connected with the node is the greatest.

To get the gradient informations in this area, we search this area with a smaller span than the searching-span, that is the gradient-span. The more stable the line segment is, the smaller the gradient-span should be. The gradient informations include the parallel-gradient and the total-gradient in this area. The total-gradient is the sum of point's gradient when its orientation deviation to the current line segment is smaller than th_θ, and the parallel-gradient is the sum of the parallel component of the point's gradient.

2) One Direction Extension: At the beginning, it is the searching process. If an extending hypothesis is detected, we judge whether it can be accepted or not according to the following principles, which we call them the EPs (extending principles). (1) The parallel gradient in the searching area should be great enough in quantity. The actual parallel-gradient of the searching area should be no less than the supposed parallel-gradient of the area, which is a linear interpolation of the gradient of the two closest portions of the current line segment to the current searching terminal point. (2) The parallel gradient in the searching area should be great enough in ratio. The portion of the actual parallel-gradient in this area should be no less than a limitation. (3) There should be enough edge points. Considering the influence of noise and the fact that the orientation of a point is not stable when its gradient is too small, we introduce an edge map of the image by applying a standard Canny operator on the image, the edge points in the searching area should be no less than a certain limitation. Both those three principles are weak constraints, but their combination is strong and robust. Also, an extending hypothesis is accepted if it is close enough. If the searching process detects nothing or the detected hypothesis is rejected, then it comes to the extension process. The extension is an iterative process which starts with the whole length of the searching steps and reduce by a pixel on every iteration. On each iteration, first, we search the front in three directions with $-45°$, $45°$ and $90°$ deviation to the current line segment to see if there is any potential line segment that blocks the extension; then the EPs is applied to determinate whether the line segment can be extended.

3) Line Segment Refinement: The line segments refinement is based on the gradient map, which has two steps. First, we choose 5 hypotheses for each terminal point of the line segment on the orthogonal direction, thus 25 hypotheses of the new line segment are obtained. Then, we find out the hypothesis with the greatest gradient and judge if it can be accepted by comparing the gradient increment and the offset between the position of the hypothesis line segment and the original line segment. The principle is that the bigger the offset is the greater the gradient increment should be.

4) Two Directions Extension: The two-directions-extension for zero-node line segments is conducted after the one-directions-extension, which is the same with the one-direction-extension in the overlook process, but differs in the extension process for that it is conducted in two direction: the original direction and the opposite direction. Searching process is also conducted in two directions, once a hypothesis in any direction is searched and can be accepted, we halt the two-direction-extension and turn to one-direction-extension. If the overlook process returns no hypothesis or the hypothesis is rejected, we start the extension in two direction which differs from the one direction extension only in two aspects: (1) the searching is in two directions; (2) we choose the direction with a bigger gradient to extend.

C. Splitting

After extension, some line segments are intersected with one or more line segments, thus should be split into short ones. For each modified or new added node, we find out all line segments connected with it and try to split the line segment at this node. After splitting, the line segments can be used as input for next extension. Extending and splitting are conducted iteratively, the iteration does not stop until iterative times reach the upper limitation, or the line segments length deviation of two adjacent iterations is smaller than a certain limitation.

D. Adding

The extension of a line segment ends when there is not sufficient parallel gradient or there is a block in front, which means it is very likely to be a potential line segment there, thus we introduce the adding process to add line segments that have not been detected by the former line segment detectors. The adding process has five steps. First, a grid is built to store all the terminal points of line segments whose length is greater than a certain limitation. Second, we sort these line segments according to their length and start the searching from the longest line segment. Third, for each line segment to be searched, we find out its terminal point that is not a node and search for other terminal points around this point according to the grid. Fourth, for the current searching point we find 5 closest points around as the potential start point, for each non-node-point around we also find 5 points as the potential end points and for node-points we set only itself as the potential end point. Fifth, we link each potential start point with end points to form a line segment hypothesis and judge if it can be accepted according to the parallel gradient ratio and the edge points' quantity. For each point around we set the accepted line segment hypothesis with the greatest edge points as the best line segment hypothesis, and if it is longer than a certain limitation, it is added as a new line segment.

E. Linking

After the extension, there are still some line segments supposed to be connected are separated. Thus we bring in the linking process which differs from extending in two aspects: a more stricter angle threshold and a more looser gradient threshold. There are two steps.

First we search for line segments that can be connected. Line searching is similar with the searching process of extension, it has a same searching span but a longer searching steps, we set it 100% of the original line segment's length. Whenever we find a line segment or a node in this area, searching is halted which means there is very likely to be a block nearby. A detected line segment is accepted as a hypothesis if its orientation and orthogonal distance are close enough to current line segment.

Then we try to link these two line segments. If line searching return a hypothesis, we firstly try to link it with the current line segment to form a merged line segment. Then we refine the merged line segment and all the other line segments connected with it if it has more than one nodes, otherwise only the merged line segment itself is refined. If the gradient of these line segments is greater than a certain percent of their original gradient, the hypothesis is accepted. After that, the detected line segment is deleted and the new line is rewrote to the binary map.

F. Merging

Given a line-networks with line segments and nodes connected with each other, we apply the merging process to merge connected close orientation line segments to form a neater line-networks with longer and less line segments. There are three steps.

First, we sort the line segments that have at least one node according to their length, the merging starts from the longest one.

Second, we search for close orientation line segments node by node. Node searching is to find out the line segments that are connected with each other with close orientations. It starts from the nodes of the longest line segment with more than one node. If the orientation of a detected line segment is close to current line orientation, we set the merged line segments as the current line segment and the other node of the detected line segment as the current node. Node searching is iterative, which stops when there is no line segment connected with the current node having a close orientation, or the detected line segment has only one node that is the current node. If the current line segment has only one node, we conduct node searching in one direction. Otherwise, the process runs in the normal direction and the opposite direction of the line segment simultaneously, the results of each direction are then combined together.

Finally, we try to merge these searched line segments to form a single line segment. There are three steps. (1) We try to merge the detected line segments as more as possible to obtain a longer line segment. It is achieved by applying an iterative process which starts from merging all the detected line segments and then removes the farthest one if the gradient of the merged line segment is less than a certain percent of the sum gradient of all line segments that forms the merged line segment. (2) We calculate the new positions of the nodes detected in node searching process, which are now supposed to be on the new merged line segment, and update the line segments connected on these nodes. (3) The detected line segments are all removed.

III. Experimental Results

In this section we tested our proposed method (NETLines) on images of a wide variety of scenes and compared the result with the state-of-the-art approach EDlines. To verify the robustness of our method, we test it on three kinds of scenes: the indoor scene, the outdoor scene and the clutter scene. For each kind of scene, 10 pictures are tested. What should be mentioned is that for all those tested pictures, to get the best result only one parameter is adjusted that is the length threshold on the adding process. Fig. 4 is the statistical result of NETLines on three different scenes, we can see from the left figure that the proposed method improves the total length of detected line segments most on the indoor scene in which there are more regular objects and less line segments that are

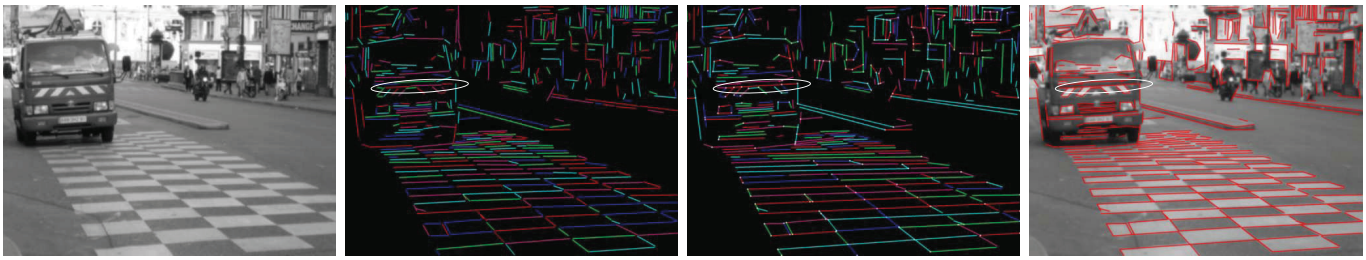

Fig. 3. Example on line segment splitting and merging: from left to right are the original image, the line segments detected by EDLines, the line segments refined by the proposed method, and the overlapped image of the refined line segments on the original image. The line segments in the white ellipse is connected and split into a clear network.

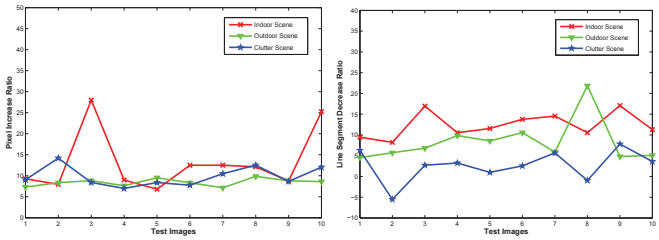

Fig. 4. Test result on three kinds of scenes: the left one is the pixel increase ratio on different scenes, the right one is the line segment decrease ratio on these scenes.

salient enough to be detected out by EDLines. Meanwhile, the NETLines can also perform well on both the outdoor and clutter scenes, which shows the robustness of the proposed method. In the right one of Fig. 4, we can see that on most cases the NETLines can reduce the number of detected line segments in a certain degree, but there are still some situations in which the proposed method detects more line segments than that of EDLines, the second test image of the clutter scene for example. The reason is that the splitting procedure can break a long line segment into several short ones on the nodes. Fig. 3 is a demonstration of this condition, we can see that the horizontal line segments in the white ellipse is first merged into a long line segment and then split by the oblique line segments below into several short ones, which may leads to the increasing of line segments' number.

Table I shows the average ratio of line segments (#LSegments) decreased and that of edge pixels belong to line segments (#LPixels) increased on these three kinds of scenes. We can see from the table that the proposed method NETLines can perform well on a wide variety of scenes. Fig. 6 is a demonstration of detection result on three kinds of scenes, we can get from these images the following conclusions: (1) not only line segments are extended to be longer but also several line segments are added on the right place, especially on the indoor scene; (2) line segments detected by NETLines fit to the edge of objects better, especially when an edge is detected as several short line segments by EDLines but merged by NETLines; (3) the line-networks of image are built. However there is still a problem: line segments may

Scene	↓#LSegments (%)	↑#LPixels (%)
Indoor	13.2%	12.4%
Outdoor	8.2%	8.4%
Clutter	9.8%	2.6%

TABLE I

TEST RESULT ON THREE KINDS OF SCENES.

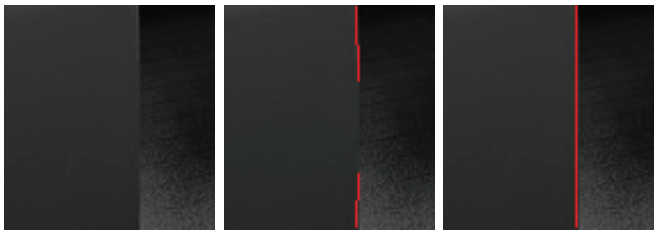

Fig. 5. Precision comparing between the result of EDLines and NETLines: from left to right are the original image, the overlapped image of the line segments detected by EDLines on the original image, and the overlapped image of the refined line segments on the original image.

be added between two terminal points where exists no edge exactly. This situation often occurs when the texture around the edge is too clutter.

To evaluate the precision of our method, we also use 10 artificial images which contain several simple edges with some noise on them to compare the detection results of EDLines and our method. The EDLines turns out to break on these noises while NETLines can detect them as whole line segments on most cases. The statistical result shows that line segments detected by NETLines is 1° to 2° more accurate than that of the EDLines on general. Fig. 5 is a comparing of the line segment precisions between the result of EDLines and that of NETLines, we can see that the proposed NETLines detects a single vertical line segment, while the EDLines extracts two short line segments with a small deviation to the vertical direction.

IV. CONCLUSION

In this paper, we propose a method to refine the line segments based on the gradient of the original image and recover the line-networks by a series of line segments operations. We test the robustness and validity of our method on

Fig. 6. Experimental result on three different kinds of scenes. The four columns from left to right are: the line segments detected by EDLines, the overlapped image of EDLines, the line-networks recovered by NETLines, the overlapped image of NETLines. In (a) the number of line segments detected by EDLines is 518 with a total line length of 22105.5 in pixels, these of NETLines are 469 and 24157.7. In (b) these figures are 1821 and 66192.2 for EDLines, 1697 and 72027.4 for NETLines. In (c) these figures are 305 and 6425.0 for EDLines, 302 and 6962.5 for NETLines.

different kinds of scenes, experiments indicate our method has a fairly good performance. The detected line segments have less line numbers and more pixels, which means the result is more efficient than current line detection approaches. In the future, we will develop the multi-images-based line detection method, and apply it to the recovering of the 3D layout in indoor scene.

REFERENCES

[1] Cuneyt Akinlar and Cihan Topal, "EDLines: A real-time line segment detector with a false detection control," *Pattern Recognition Letters*, vol. 32, no. 13, pp. 1633–1642, 2011.

[2] Herbert Bay, Vittorio Ferraris, and Luc Van Gool, "Wide-baseline stereo matching with line segments," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005.

[3] Hyunwoo Kim and Sukhan Lee, "Simultaneous line matching and epipolar geometry estimation based on the intersection context of coplanar line pairs," *Pattern Recognition Letters*, vol. 33, pp. 1349–1363, 2012.

[4] Jun Chu, Anzheng GuoLu, Lingfeng Wang, Chunhong Pan, and Shiming Xiang, "Indoor frame recovering via line segments refinement and voting," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 2013.

[5] Radu Stoica, Xavier Descombes, and Josiane Zerubia, "A gibbs point process for road extraction from remotely sensed images," *International Journal of Computer Vision*, vol. 57, pp. 121–136, 2004.

[6] John Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[7] John Illingworth and Josef Kittler, "A survey of the hough transform," *Computer vision, graphics, and image processing*, vol. 44, pp. 87–116, 1988.

[8] Leandro AF Fernandes and Manuel M Oliveira, "Real-time line detection through an improved hough transform voting scheme," *Pattern Recognition*, vol. 41, pp. 299–314, 2008.

[9] Cihan Topal and Cuneyt Akinlar, "Edge Drawing: A combined real-time edge and segment detector," *Journal of Visual Communication and Image Representation*, vol. 23, no. 6, pp. 862C–872, 2012.

[10] Agnes Desolneux, Lionel Moisan, and Jean-Michel Morel, "Edge detection by Helmholtz principle," *Journal of Mathematical Imaging and Vision*, vol. 14, no. 3, pp. 271–284, 2001.

[11] Agnès Desolneux, Lionel Moisan, and Jean-Michel Morel, *From gestalt theory to image analysis: a probabilistic approach*, Springer, 2008.

[12] J. Brian Burns, Allen R. Hanson, and Edward M. Riseman, "Extracting straight lines," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 4, pp. 425–455, 1986.

[13] R. Grompone von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, 2010.

[14] R Grompone Von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall, "Lsd: A line segment detector," *Image Processing On Line*, 2012.