www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

Detection of prediction-correction lines

[1]Mr. Ch Surya Prakash, [2]Anchuri Padmavathi,

[1]Associate Professor, Department of MCA, Rajamahendri Institute of Engineering & Technology. Bhoopalapatnam, Near Pidimgoyyi,Rajahmundry,E.G.Dist.A.P. 533107.

[2]Student,Department of MCA, RajamahendriInstitute of Engineering & Technology. Bhoopalapatnam, Near Pidimgoyyi,Rajahmundry,E.G.Dist.A.P. 533107.

ABSTRACT

Because of their remarkable efficiency, edge-drawing approaches are becoming more and more common in line segment identification. However, in order to reduce false positives, most current algorithms use a threshold that is already set on the input image's gradient magnitude. This might cause line segments that aren't completely complete to be detected. The prediction-correction line segment detector (PCLSD) is our new approach to fixing this underlying issue. A Canny-based technique is used to create line segment predictions in the PCLSD's first prediction step. Each anticipated line segment is fine-tuned in the next corrective step. In order to improve the line segment's orientation, location, and completeness, a directional routing approach is used for its extension and refit. Finally, trust is ensured by validating the rectified line segment. The suggested PCLSD outperforms the state-of-the-art approaches according to the experimental data.

INTRODUCTION

Line segments are widely used in many computer vision tasks because of the wealth of geometric and topological information they convey in real-world scenes. For example, they are useful for 3D reconstruction, SLAM, pose estimation, vanishing point detection, and power line extraction from UAV images. There have been a plethora of line segment detectors developed in the last few decades. These current approaches may seem different, but they really use the same processing procedures to identify line segments. The process begins with feature extraction from the input picture at a basic level and continues with line segment identification using these characteristics. The most common low-level picture characteristics used are line-support areas [13, 14], edge points [6-12], and linelets [15]. These basic attributes are all obtained from the same picture cues—gradient directions and magnitudes—from a more foundational viewpoint. As a result, the two visual gradient cues indicated earlier are crucial to the successful recognition of line segments. Pixels that are aligned in this way show very large gradient magnitudes, suggesting strong contrast, and a line segment is deemed salient if its locally calculated gradient directions agree well with its normal vector. When this is true, it's possible to extract enough low-level picture characteristics to recognize the line segment with high completeness. Regrettably, in order to limit false positives, most current methods—particularly those that rely on edge drawing—impose a fixed threshold on the amplitude of the gradient. Nevertheless, the completeness of the discovered line segments is compromised by this method. As an example, consider the following:

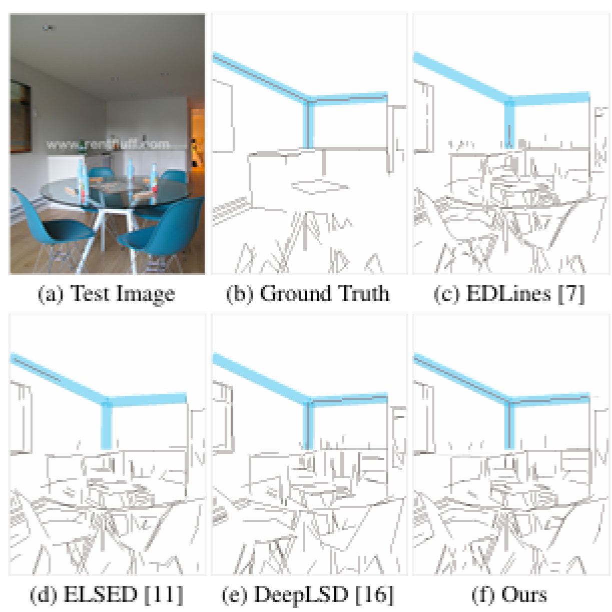
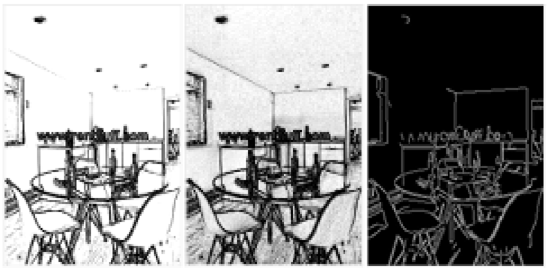


Fig. 1 for demonstration. Fig. 1(a)

captures an interior scene with three crossing line segments—one each between the walls and the ceiling, and also between the walls themselves.



(a) (b) (c)

shown in Fig. 1(b) as the ground truth, as indicated in blue. Low gradient magnitudes make these line segments seem low-contrast, which is a result of the roof and walls being the same color. According to Figure 1(c) and (d), the gradient magnitudes of these line segments usually fall below the imposed pre-determined threshold, making them hard to identify with a high degree of completeness. They are also commonly accompanied by notable orientation and position problems. Figure 1(e) shows that the aforementioned problems have not been adequately solved, even though various deep learning-based line segment detectors have been proposed [16–18]. The prediction-correction line segment detector (PCLSD) is a new approach to line segment detection that this work suggests as a solution to the problems mentioned before. A line segment predictor (LSP) that makes use of an adaptive Canny edge detector [8] is what makes it unique. After that, line segments are corrected to improve their quality using a directed routing approach. Prediction and correction may identify low-contrast line segments, as seen in Figure 1(f). This will lead to a significant improvement in line segment detecting performance. The following is the outline for the remainder of the paper. Section 2 explains our suggested PCLSD in detail. Section 3 presents and discusses extensive experimental data. Section 4 concludes the whole thing. 2. The Approach Suggested Outline There are primarily two processes involved in line segment detection using edge drawing methods: 1) finding the pixels with the highest gradient magnitudes; these are called anchor points. 2) fitting line segments by linking these anchor points using a routing approach. The candidate pixel set, shown in Fig. 2(a), is derived from the two phases mentioned earlier by applying a threshold to the gradient magnitude. As is evident, the potential pixel

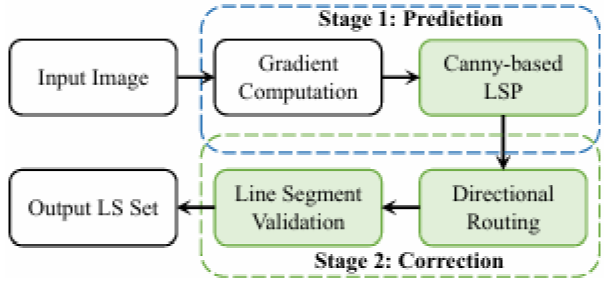


Fig. 3: The pipeline of the proposed method,

where the green highlights the contribution or uniqueness of our work. The set of anchor points grows in tandem with the set as the preset threshold shrinks. Although more full line segments may be recognized, this leads to an increase in false positives. As a rule, line segments tend to crop up toward the borders of images, where gradients tend to trend. Hence, instead of simply using the set of pixels where a threshold has been applied to the grain magnitude, our PCLSD retrieves chor points based on picture edges. Our extracted anchor points can be thought of as line segment predictions, comparable to drawing a solid line with a dotted line. Then, we use a directional routing method to fill in the predicted line segments, which is like drawing a full solid line matching the dotted line. Figure 3 displays the PCLSD pipeline. 2.2. Calculating Gradients The input picture is represented as I(x,y). The picture I(x,y) is first subjected to a modest degree of Gaussian smoothing in order to reduce noise. Subsequently, the image gradient is constructed using the Sobel operator: $\nabla I = (gx,gy)T$, where gx and gy denote the derivatives with respect to x and y, respectively. As a result, we can determine the gradient direction by

$$G_\theta = \arctan(g_y/g_x), \tag{1}$$

and the gradient magnitude is given by

$$G_m = |g_x| + |g_y|, \tag{2}$$

in that order. It should be noted that the gradient magnitude is computed using the L1 norm due to its cheap computational expenditure. The gradient direction Gθ is binarized in the following way to make the follow-up directional routing step easier: it is vertically oriented if and only if |gx| is less than or equal to |gy|, and it is horizontally oriented otherwise. 2.3. Line Segment Prediction Based on Canny When we implemented the line segment predictor, we employed an adaptive Canny edge detector [8]. In addition to producing edge segments made up of edge pixels that are almost perfectly linear, this edge detector also gives you a variable lower bound Tm on the gradient magnitude. Be mindful that while these perimeter sections are

Table 1: Comparison of different line segment detection methods in terms of APR, IoU and F-score. The best score is in bold and the second best is underlined.

Methods	YorkUrban-LineSegment dataset [15]						ShanghaiTech Wireframe dataset [19]					
	APR (%)		IoU (%)		F-score (%)		APR (%)		IoU (%)		F-score (%)	
	Con_1	Con_2	Con_1	Con_2	Con_1	Con_2	Con_1	Con_2	Con_1	Con_2	Con_1	Con_2
LSD [13]	43.30	32.75	27.46	19.61	42.33	32.05	26.97	20.04	14.46	10.38	24.80	18.42
EDLines [7]	44.84	32.97	28.74	19.83	43.90	32.30	28.26	20.89	15.23	10.89	25.95	19.23
CannyLines [8]	44.51	35.13	28.37	21.27	43.47	34.35	28.21	22.01	14.96	11.33	25.57	19.96
MCMLSD [9]	46.91	37.54	30.45	23.11	46.28	37.03	<u>33.23</u>	**26.85**	15.51	12.21	26.43	21.37
Linelet [15]	43.79	32.78	28.30	19.88	43.42	32.50	28.27	21.58	14.60	10.82	25.03	19.13
AG3line [10]	<u>49.38</u>	<u>38.26</u>	<u>32.92</u>	<u>23.87</u>	<u>48.93</u>	<u>37.93</u>	32.27	24.76	15.89	11.79	26.99	20.74
SOLD^2 [17]	25.20	13.13	11.43	5.66	20.19	10.55	20.89	11.10	11.02	5.60	19.41	10.33
ELSED [11]	47.12	38.00	30.51	23.42	46.11	37.23	29.71	23.63	<u>16.29</u>	<u>12.60</u>	<u>27.59</u>	<u>21.97</u>
DeepLSD [16]	45.43	34.74	29.25	21.09	44.77	34.25	29.61	22.33	15.87	11.55	26.93	20.30
PCLSD(Ours)	**50.70**	**40.45**	**34.03**	**25.56**	**50.10**	**39.99**	**33.93**	<u>26.84</u>	**18.43**	**14.08**	**30.62**	**24.23**

could save more detailed information about the line segments in the image, but they may only store a portion of the data about the low-contrast line segments (see to Fig. 2(c)). While Lu et al. [8] limit their line segment recognition to these edge segments, we take into account all the relevant pixels for line segment extraction. It is easy to implement the line segment predictor using the edge segments: Ei is used to indicate a certain edge segment. If every pixel (x,y) in Ei is a local maximum in the gradient magnitude (in the 8 neighborhood), then it is considered an anchor point. Named Li, the collection of anchor points that make up this edge segment is a prediction of a line segment (shown as a dotted line to make it easier for the routing algorithm to construct a full line segment later on). Lp = {Li}N i=1, where N is the number of edge segments, is the set of all line segment predictions. Method for Directional Routing (2.4) Full line segments are drawn using a method similar to that of Su'arez et al. [11]. Furthermore, to ensure that pixels from low-contrast line segments are not filtered out, we choose Tm as the threshold applied to the gradient magnitude (see Fig. 2(b)). We carry out the following steps sequentially for every set of anchor points Li, where |Li| ≥ 2. that reparameterization is required for each line segment extension in order to update orientation and position data. 2.5. Validation of Line Segments False positives may be common in the directed routing output line segments, especially in areas with a dense concentration of edge pixels. A post validation is used to guarantee the reliability of the last portions of the output line: To get the angle error, ME, for every pixel x that is used to create a line segment, we begin by projecting it onto the segment. Then, we use bilinear interpolation to determine the gradient direction of this projected point, and we compare it to the normal direction of the segment. Here is how we assign a score, S, to every line segment: The line segment is deemed to have a high level of confidence and proceeds to pass the validation process because S = 1 |L| x∈L 1{θE(x) 0.5. • The anchor points should be used to parameterize a line segment. To lengthen the line segment, use the following set of characteristics to the pixels: 1) exhibiting the highest possible gradient magnitude, with Gm being equal to or more than Tm; 2) being sufficiently close to the line segment; and 3) sharing the same orientation (horizontal or vertical) as the line segment. • Always use a depth-first technique to expand, meaning add pixels in the same direction as the current one as much as feasible. The search tree gains a new sister branch whenever an opposing orientation is challenged. Once there is only a single extendable pixel, the technique reverts to the one used by Su'arez et al. [11]. Important to mention

$$S = \frac{1}{|L|} \sum_{x \in L} \mathbb{1}_{\{\theta_E(x) < T_\theta\}}. \qquad (3)$$

EXPERIMENTS

A pair of reference datasets, the YorkUrban-LineSegment dataset [15] and the ShanghaiTech Wireframe dataset [19], are used for the experiments. Line segment identification tasks make extensive use of the first dataset, which comprises 102 street photos. The second collection of data includes 462 pictures of man-made items, both inside and out, together with line segments that represent the scene's spatial geometry. For comparison, we have included nine state-of-the-art methods: LSD [13], EDLines [7], CannyLines [8], MCMLSD [9], AG3line [10], ELSED [11], Linelet [15], and SOLD2 [17]. and

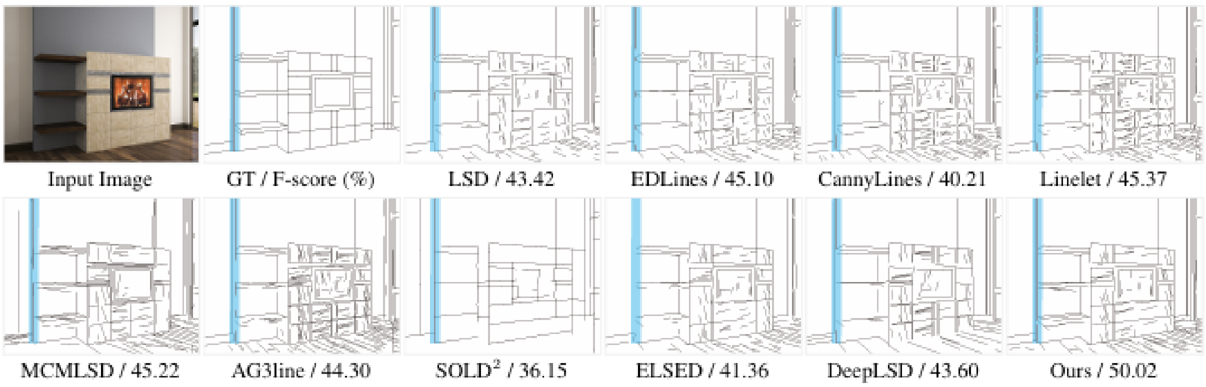


Fig. 4: Subjective comparison of different methods on the image #330 taken from the ShanghaiTech Wireframe dataset [19].

DepthLSD [16]. Two of these current approaches, SOLD2 and DeepLSD, use deep learning to detect line segments. For our trials, we're using the default settings and the publicly accessible code that each method's developers have made available. The three most used metrics for assessment are the APR, the F-score, and the intersection over union. Together, they represent the harmonic mean of recall and accuracy. These metrics are determined by comparing the detected line segments to the ground truth and measuring the spatial distance λd, angular difference λθ, and intersection λa. This allows for the identification of true positives within a set of allowable error tolerances [15]. We have two evaluation conditions in our trials. One is a lenient one, with parameters {λθ = 5◦,λd = 1,λa = 0.75}, and the other is a strict one, with parameters {λθ = 1◦,λd = 1,λa = 0.75}, marked as Con2. In Con1, as proposed in [15], λd = 1 and λa = 0.75 are very stringent requirements; nevertheless, λθ = 5◦ might lead to the acceptance of low-quality true positives, since the orientation divergence from the ground truth could exceed 5◦. So, in Con2, we set λθ to 1◦ to apply a stronger limitation. The results of the several methods on the two benchmark datasets are summarized in Table 1. Under both test scenarios, our suggested approach outperforms the state-of-the-art in every measure, with the exception of Con2's APR score on the ShanghaiTech Wireframe dataset [19]. Even though we didn't get first place in the previous scenario, we performed second best, which is almost the same as first place. Different approaches identify line segments, as shown qualitatively in Fig. 4. By enhancing the completeness of identified line segments without compromising detection accuracy, our technique achieves the optimum performance. In general, PCLSD significantly surpasses the current approaches. We examined the various line segment recognition algorithms to get a clearer picture of how well they worked.

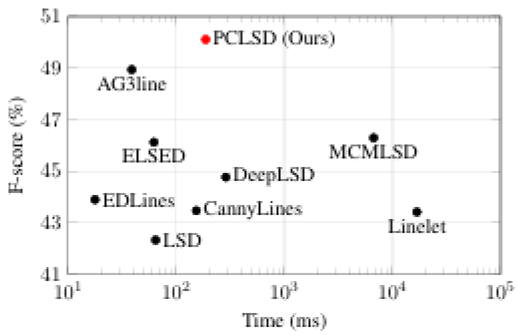


Fig. 5: Average run time comparison of our PCLSD and other comparable methods using the images from the YorkUrbanLineSegment dataset [15].

duration of each procedure on a single computer with a 2.70 GHz Intel Core i5-11400H central processing unit and an NVIDIA Titan Xp graphics processing unit. Figure 5 displays the performance points that were derived by comparing the average run times to the average F-score; points below the lower limit are not shown. When compared to the detection performance, our method's slower operation is tolerable, especially considering how long the Canny based line segment prediction step takes.

Conclusion

In summary, Unfortunately, current line segment detectors often produce incomplete line segments, frequently with apparent orientation and position problems, since they use a pre-set threshold on the gradient magnitude of the input picture to handle false positives. This study introduces PCLSD, a new approach to line segment detection that achieves remarkable precision in terms of orientation, location, and completeness. Two benchmark datasets' worth of experimental findings show that PCLSD outperforms the current state-of-the-art approaches.

REFERENCES

[1]. [1] Dong Wei, Yi Wan, Yongjun Zhang, Xinyi Liu, Bin Zhang, and Xiqi Wang, "ELSR: Efficient line seg ment reconstruction with planes and points guidance," in IEEE Conference on Computer Vision and Pattern Recognition, 2022, pp. 15807–15815.

[2]. [2] Xin Liu, Shuhuan Wen, and Hong Zhang, "A real-time stereo visual-inertial SLAM system based on point-and line features," IEEE Transactions on Vehicular Technol ogy, vol. 72, no. 5, pp. 5747–5758, 2023.

[3]. [3] Xinyu Lin, Yingjie Zhou, Xun Zhang, Yipeng Liu, and Ce Zhu, "Efficient and effective multi-camera pose es timation with weighted M-estimate sample consensus," in IEEE International Conference on Acoustics, Speech and Signal Processing, 2023, pp. 1–5.

[4]. [4] Xin Tong, Xianghua Ying, Yongjie Shi, Ruibin Wang, and Jinfa Yang, "Transformer based line segment classifier with image context for real-time vanishing point detection in Manhattan world," in IEEE Conference on Computer Vision and Pattern Recognition, 2022, pp. 6093–6102.

[5]. [5] Wenbo Zhao, Qing Dong, and Zhengli Zuo, "A method combining line detection and semantic segmentation for power line extraction from unmanned aerial vehicle im ages," Remote Sensing, vol. 14, no. 6, pp. 1367, 2022.

[6]. [6] Charles Galamhos, Jose Matas, and Josef Kittler, "Pro gressive probabilistic Hough transform for line detection," in IEEE Conference on Computer Vision and Pat tern Recognition, 1999, pp. 554–560.

[7]. [7] Cuneyt Akinlar and Cihan Topal, "EDLines: A real time line segment detector with a false detection con trol," Pattern Recognition Letters, vol. 32, no. 13, pp. 1633–1642, 2011.

[8]. [8] Xiaohu Lu, Jian Yao, Kai Li, and Li Li, "CannyLines: A parameter-free line segment detector," in IEEE In ternational Conference on Image Processing, 2015, pp. 507–511.

[9]. [9] Emilio J Almazan, Ron Tal, Yiming Qian, and James H Elder, "MCMLSD: A dynamic programming approach to line segment detection," in IEEE Conference on Com puter Vision and Pattern Recognition, 2017, pp. 2031 2039.

[10]. [10] YongjunZhang, DongWei,andYanshengLi, "AG3line: Active grouping and geometry-gradient combined vali dation for fast line segment extraction," Pattern Recognition, vol. 113, pp. 107834, 2021. 3519 "ELSED: Enhanced line segment drawing," Pattern Recognition, vol. 127, pp. 108619, 2022.

[11]. [12] Xinyu Lin, Yingjie Zhou, Yipeng Liu, and Ce Zhu, "Level-line guided edge drawing for robust line segment detection," in IEEE International Conference on Acous tics, Speech and Signal Processing, 2023, pp. 1–5.

[12]. [13] Rafael 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 Intelli gence, vol. 32, no. 4, pp. 722–732, 2008.

[13]. [14] Naila Hamid and Nazar Khan, "LSM: Perceptually ac curate line segment merging," Journal of Electronic Imaging, vol. 25, no. 6, pp. 061620, 2016.

[14]. [15] Nam-Gyu Cho, Alan Yuille, and Seong-Whan Lee, "A novel linelet-based representation for line segment de tection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 5, pp. 1195–1208, 2017.

[15]. [16] R´ emiPautrat, Daniel Barath, Viktor Larsson, Martin R. Oswald, and Marc Pollefeys, "DeepLSD: Line segment detection and refinement with deep image gradients," in IEEE Conference on Computer Vision and Pattern Recognition, 2023, pp. 17327–17336.