

Learning Regional Attraction for Line Segment Detection

Nan Xue, Song Bai, Fu-Dong Wang, Gui-Song Xia,
Tianfu Wu, Liangpei Zhang, Philip H.S. Torr

Abstract—This paper presents *regional attraction* of line segment maps, and hereby poses the problem of line segment detection (LSD) as a problem of region coloring. Given a line segment map, the proposed regional attraction first establishes the relationship between line segments and regions in the image lattice. Based on this, the line segment map is equivalently transformed to an attraction field map (AFM), which can be remapped to a set of line segments without loss of information. Accordingly, we develop an end-to-end framework to learn attraction field maps for raw input images, followed by a squeeze module to detect line segments. Apart from existing works, the proposed detector properly handles the local ambiguity and does not rely on the accurate identification of edge pixels. Comprehensive experiments on the Wireframe dataset and the YorkUrban dataset demonstrate the superiority of our method. In particular, we achieve an F-measure of 0.831 on the Wireframe dataset, advancing the state-of-the-art performance by 10.3 percent.

Index Terms—Line Segment Detection, Low-level Vision, Deep Learning

1 INTRODUCTION

LINE segment detection (LSD) is an important yet challenging low-level task in computer vision [1], [2], [3]. LSD aims to extract visible line segments in scene images (see Figure 1(a) and Figure 1(b)). The resulting line segments of an image provide a compact structural representation that facilitates many up-level vision tasks such as 3D reconstruction [4], [5], image partitioning [6], stereo matching [7], scene parsing [8], [9], camera pose estimation [10], and image stitching [11].

LSD is usually formulated as a heuristic search problem [2], [3] that groups or fits the edge pixels into several line segments. The classical Hough transform (HT) [13], as well as some HT-based variants [3], [14], [15], [16], [17], takes locally estimated edge maps as input to fit straight lines in the first step and then estimates the endpoints of the line segments according to the density of the edge pixels on these straight lines. These methods suffer from the incorrect edge pixel identification (see Figure 1(c) in the locally estimated edge maps, and often produce a number of false positive detections (see the result of MCMLSD [3] in Figure 1(d)).

In contrast to HT-based approaches, Burn *et al.* [1] attempted to locally group edge cues into line segments. Following this, LSD [2] and Linelet [18] grouped pixels with high gradient magnitude values (*i.e.*, edge pixels) into line

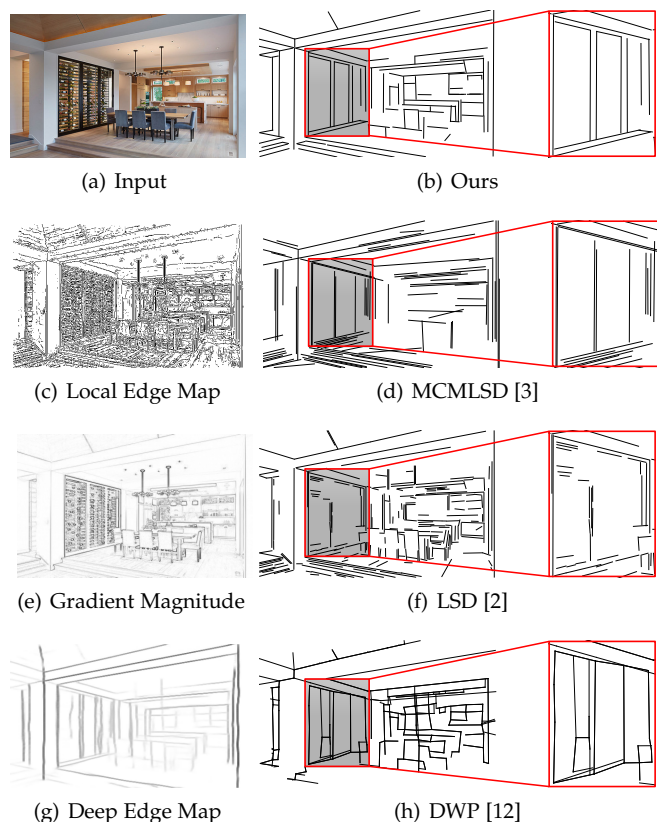


Fig. 1. Illustrative examples of different methods on an image (a). (b) shows our detected line segments. (c) and (d) present the locally estimated edge map and the result of MCMLSD [3]. (e) and (f) present the gradient magnitude and the result of LSD [2]. (g) and (h) display the deep edge map and the result of Deep Wireframe Parser (DWP) [12]. The rightmost column shows the close-up (in red) of detection results by different methods, which highlights the better accuracy of our proposed method. Best viewed in color.

- N. Xue, F.-D. Wang and L. Zhang are with State Key Lab. LIESMARS, Wuhan University, China.
E-mail: {xuenan, fudong-wang, zlp62}@whu.edu.cn
- G.-S. Xia is with School of Computer Science and the State Key Lab. LIESMARS, Wuhan University, China.
E-mail: guisong.xia@whu.edu.cn
- S. Bai and P. Torr are with the University of Oxford, United Kingdom.
E-mail: songbai.site@gmail.com, philip.torr@eng.ox.ac.uk
- T. Wu is with Dept. Electrical & Computer Engineering, NC State University, USA.
E-mail: tianfu_wu@ncsu.edu
Corresponding author: Gui-Song Xia (guisong.xia@whu.edu.cn)

segment proposals according to the gradient orientation. Once the line segment proposals were obtained, the validation processes based on the Helmholtz principle [19], [20] were applied to reject false positive detections. However, edge pixels in low-contrast regions were prone to being omitted, thereby breaking a long line segment into several short ones. An example of LSD [2] and the corresponding gradient magnitude are given in Figure 1(f) and Figure 1(e) respectively.

It is problematic for those methods to detect complete line segments while suppressing false alarms using traditional edge cues [2], [3], [18]. Furthermore, the edge pixels can only approximately characterize the line segment as a set of connected pixels, also suffering from unknown multiscale discretization nuisance factors (*e.g.*, the classic zig-zag artifacts of line segments in digital images).

In recent years, convolutional neural networks (ConvNets) have demonstrated a potential for going beyond the limitation of local approaches to detect edge pixels with global context. The Holistically-nested Edge Detector (HED) [21] used the fully convolutional network (FCN) architecture [22] for the first time to learn and detect edge maps for input images in an end-to-end manner. Later, many deep learning based edge detection systems were proposed [23], [24], [25] and significantly outperformed traditional edge detectors [26], [27], [28], [29]. Benefiting from the advances in deep edge detection, the deep wireframe parser (DWP) [12] transforms line segment detection into edge maps and junction detections with two ConvNets and then fuses detected junctions and edge pixels into line segments. As shown in Figure 1(g), the estimated edge maps can identify edge pixels better in regions with complicated appearances, thus pushing the performance bounds of LSD forward by a large margin. However, the over-smoothing effect of deep edge detection will lead to local ambiguity for accurate line segment detection. In Figure 1(h), some detected line segments are misaligned because of the blurred edge responses.

In summary, most previous work [2], [3], [12], [18] is built upon edge pixel identification and suffers from two main drawbacks: such work lacks elegant solutions to solve the issues caused by inaccurate or incorrect edge detection results (*e.g.*, local ambiguity, high false positive detection rates and incomplete line segments) and requires carefully designed heuristics or extra contextual information to infer line segments from identified edge pixels.

In this paper, we focus on a deep learning based LSD framework and propose a single-stage method that rigorously addresses the drawbacks of existing LSD approaches. Our method is motivated by the following observations:

- The duality between regions and the contour (or the surface) of an object is well-known in computer vision [30].
- All pixels in the image lattice should be involved in the formation of line segments in an image.
- The recent remarkable progress led by deep learning based methods (*e.g.*, U-Net [31] and DeepLab V3+ [32]) in semantic segmentation.

Thus, the intuitive idea of this paper is that when bridging a line segment map and its spatial proximate regions, we

can pose the problem of LSD as the problem of region coloring, and thus open the door to leveraging the best practices developed in state-of-the-art deep ConvNet based semantic segmentation methods to improve the performance of LSD.

1.1 Method Overview

Following this idea, we exploit the spatial relationship between pixels in the image lattice and line segments, and propose a new formulation termed *regional attraction* for line segment detection (as shown in Figure 2). Our proposed regional attraction establishes the relation between 1D line segments and 2D regions of image lattices, and an induced representation characterizes the geometry of line segments by using edge pixels and non-edge pixels together. Compared with the previous formulation of line segment detection, our proposed *regional attraction* can directly encode the geometric information of line segments without using the edge maps.

By learning the regional attraction, our proposed line segment detector eliminates the limitations of edge pixel identification. As shown in Figure 1, our method yields a much better result than several representative line segment detectors, especially in the gray region that has high-frequency textures.

We establish the relationship between pixels and line segments by seeking the most “attractive” line segment for every pixel in the image lattice. Suppose that there are n line segments on an image lattice Λ , where the most attractive line segment for every pixel $\mathbf{p} \in \Lambda$ is defined as the nearest line segment of pixel \mathbf{p} . By applying this criterion, the pixels in the image lattice Λ are partitioned into n regions $\{R_i\}_{i=1}^n$, which form a region-partition map. Consequently, non-edge pixels are also involved to depict the geometry of line segments. In detail, we use the shortest vector from every pixel $\mathbf{p} \in R_i$ to its most “attractive” line segment to characterize the geometric property of the line segment. As an example, if the pixel $\mathbf{p} \in R_i$ can reach a point inside the line segment, the vector will simultaneously depict the location and normal direction of the line segment. Otherwise, the vector indicates the endpoint of the line segment. We term such vectors as the attraction vectors. The attraction vectors of every pixel \mathbf{p} together form an attraction field map (AFM).

The format of attraction field maps is actually a two-dimensional feature map, which is compatible with convolutional neural networks. Therefore, regional attraction allows the problem of LSD to be transformed into a problem of region coloring. More importantly, thanks to recent advances of deep learning based semantic segmentation methods, it is feasible to learn the attraction field map in an end-to-end manner. Once the attraction field map of an image can be estimated accurately, regional attraction is capable of recovering the line segment map in a nearly perfect manner via a simple and efficient squeeze module. The regional attraction can also be viewed as an intuitive expansion-and-contraction operation between 1D line segments and 2D regions: the region-partition map jointly expands all line segments into partitioned regions, and the squeeze module degenerates regions into line segments.

Figure 2 illustrates the pipeline of the proposed LSD framework based on an encoder-decoder neural network.

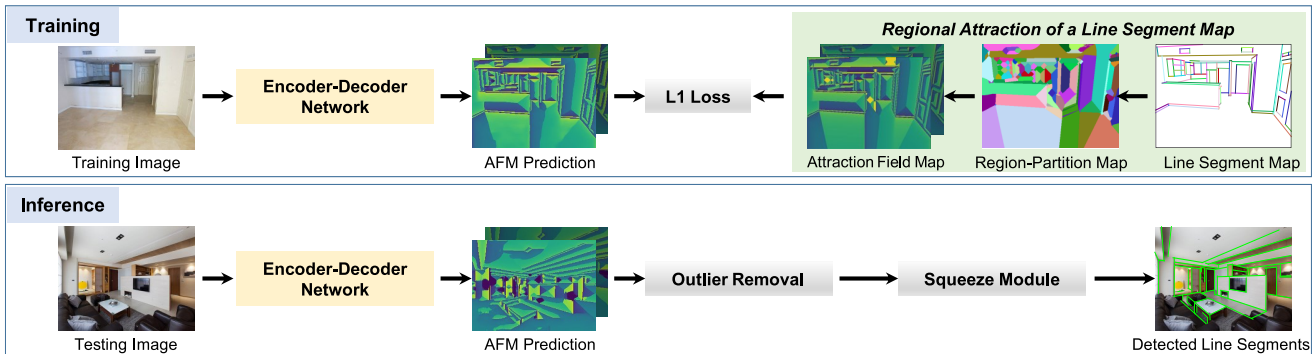


Fig. 2. An illustration of the proposed regional attraction and line segment detection system. In the training phase, the annotated line segments of an image are equivalently represented by an attraction field map (AFM). Then, the image and corresponding AFM are feed into the encoder-decoder network for learning. In the inference phase, a testing image is passed into the trained network to obtain the AFM prediction. After removing the outliers and squeezing the predictions, the system outputs a set of line segments.

Specifically, we utilize a modified network based on DeepLab V3+ [32] in our experiments to estimate the attraction field maps for line segment detection. In the training phase, the proposed regional attraction first forms a region-partition map and then generates ground truth of the attraction field map to supervise the training of the deep network. In the testing phase, the attraction field map computed by the network is squeezed to output line segments. Compared with the preliminary version of Attraction Field Map (AFM) [33], we further propose an outlier removal module based on the statistical priors of the training dataset, which significantly improves the performance of LSD. Besides, we find that better optimizer (*e.g.*, the Adam optimizer [34]) with adaptive learning rate decay can make ConvNets learn better attraction field maps. We name the enhanced version of the line segment detector as AFM++.

1.2 Contributions

Our work makes the following contributions to robust line segment detection, as

- A novel representation of line segments is proposed to bridge line segment maps and region-partition-based attraction field maps. To the best of our knowledge, it is the first work that utilizes this simple yet effective representation for LSD.
- With the proposed regional attraction, the problem of LSD is then solved by using a ConvNet without the necessity of identifying edge pixels.
- The proposed AFM++ obtains state-of-the-art performance on two widely used LSD benchmarks, including the Wireframe [12] and YorkUrban [4] datasets. In particular, on the Wireframe dataset, AFM++ beats the current best-performing algorithm by 10.3 percent.

The remainder of this paper is organized as follows. Existing research related to our work is briefly reviewed in Section 2. In Section 3, the details of the regional attraction for line segments are presented, followed by the definition of AFM++ in Section 4. The experimental results and comparisons are given in Section 5. Finally, we conclude our paper in Section 6.

2 RELATED WORK

2.1 Benchmark Datasets for Line Segment Detection

Like many other vision problems, benchmark datasets are important for evaluating the performance of a line segment detector. However, the ill-posed definition of line segment detection brings some difficulties to create a perfect benchmark dataset for line segment detection. Specifically, the perception ambiguity will lead to some inconsistency for annotating line segments from images. The well-known BSDS dataset [30] suffered from this issue in edge detection and they tried to use multi-source annotations to eliminate the ambiguity. For the problem of line segment detection, the existing benchmark datasets (*e.g.*, the Wireframe dataset [12] and the YorkUrban dataset [4]) tried to address this issue by using some priors of human perception or the scene geometry. Specifically, the line segments annotations of the Wireframe dataset [12] are obtained by associating the salient scene structures. For the YorkUrban dataset [12], they use the vanishing points as a criterion to annotate line segments and each line segment is associated with one of the vanishing points. In this paper, we use the Wireframe dataset [12] and the YorkUrban dataset [4] to evaluate our proposed line segment detector, and our proposed method consistently obtains the state-of-the-art performance on these two datasets that have different annotation rules.

2.2 Detection based on Local Edge Cues

For a long time, hand-crafted low-level edge cues were extensively used in line segment detection. The classical LSD baseline takes the output of an edge detector (*e.g.*, Canny detector [28]) and then applies Hough transform [13] (HT) to fit infinite-long straight lines. Then, line segments are obtained by cutting these straight lines according to the density of the edge pixels on the lines. Since the locally estimated edge maps suffer from a number of false positive edge pixels, it is challenging to detect line segments from input images robustly. The incorrectly identified edge pixels will produce many spurious peaks in the Hough space, which will produce a number of false positive and false negative detections. The progressive probabilistic Hough transform (PPHT) [14] proposed a false detection control to improve the detection results of the classical Hough transform. Desolneux *et al.* [19], [20] addressed the issue

of false detection by applying Helmholtz principle in line segment detection. In this method, the meaningful aligned line segments are retained as the final detections. Moreover, the distribution of peaks in the Hough space was studied in [35], [36], [37], [38] to improve the performance of LSD. Most recently, MCMLSD [3] proposed to control the false detections by exploiting the distribution of edge pixels on the voted straight lines. However, the performance of HT-based approaches still cannot achieve the satisfactory performance.

In contrast to fitting line segments from edge pixels, Burn *et al.* [1] found that the local gradient orientation is more robust to intensity variations than the gradient magnitude (and local edge maps). Based on this, a perception grouping approach [1] was proposed to detect line segments without using Hough transform. Given a gray-scale image, adjacent pixels with similar gradient orientations are grouped to yield a set of line segments. Similar to the HT-based approaches, this approach also suffers from false positive detections. Subsequently, a novel grouping approach based on Helmholtz principle [19], [20] was proposed in [39]. Afterward, LSD [2] was proposed to improve the performance of line segment detection in both speed and accuracy. Benefiting from the development of Helmholtz principle in the problem of LSD, the grouping approaches can suppress false detections by applying an *a-contrario* validation processes. Nevertheless, it is still a challenge to detect complete line segments in low-contrast regions. To this end, the ASJ detector [40] was proposed to detect long line segments starting from detected junctions [41]. However, that approach still suffers from the uncertainty caused by the image gradient. Recently, Cho *et al.* [18] proposed a linelet-based framework to address the problem of LSD. In this framework, pixels with large gradient magnitudes are grouped into linelets, and the line segment proposals are obtained by grouping the adjacent linelets. A probabilistic validation process is applied to reject false detections. To avoid incomplete results, line segment proposals passed the validation are fed into an aggregation process to detect complete line segments. Similar to the HT-based approaches, the performance of perception grouping approaches also rely on whether the image gradient can reflect the edge information in a precise way.

The performance of these line segment detectors depends on if the edge pixels can be correctly extracted. The edge maps (including image gradient maps) used for line segment detection are obtained from the local features, which are easily affected by the external imaging conditions (*e.g.*, noise and illumination). Therefore, the local nature of these approaches poses a challenge to accurately extract line segments from images even with powerful validation processes. Compared with the approaches based on local edge cues, our proposed method achieves robust line segment detection by learning more effective deep features. Moreover, our proposed detector only requires a simple criterion to reject false detections.

2.3 Deep Edge and Line Segment Detection

Recently, HED [21] opened up a new era for edge perception in images by using ConvNets. The learned multi-scale and

multi-level features effectively address the problem of false detection in the edge-like texture regions and approach human-level performance on the BSDS500 dataset [30]. From the perspective of binary classification, edge detection has been solved to some extent. It then inspires researchers to upgrade the existing edge-based line segment detectors to deep-edge based line segment detectors. Convolutional Oriented Boundaries (COB) [23], [42] detector was proposed to get multi-scale oriented contours and region hierarchies from a single ConvNet. Since the oriented contours are adaptive to the input format (*i.e.*, edge pixels and orientations) of fast LSD [2], they can be used to address the issue of incomplete detection in LSD effectively. However, the edge maps estimated by ConvNets are usually over-smoothed, which leads to local ambiguities for accurate localization. In comparison to edge detection, deep learning based line segment detection has not yet been well investigated and requires further exploration.

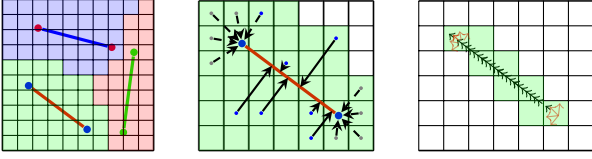
Most recently, Huang *et al.* [12] took an important step toward this goal by collecting a large-scale dataset with high-quality line segment annotations and approaching the problem of line segment detection as two parallel tasks, *i.e.*, edge map detection and junction detection. In the final step, the resulted edge map and junctions are merged to produce line segments. To the best of our knowledge, this is the first attempt to develop a deep learning based line segment detector. However, due to the sophisticated relation between edge map and junctions, inferring line segments from edge maps and junction cues in a precise way is still an open problem.

Compared with this approach, our proposed formulation enables us to detect line segments from the attraction field maps instead of using edge maps and additional junction cues. The richer geometric information encoded in the attraction field maps facilitates line segment detection without considering the blurring effect of deep edge detectors.

Furthermore, learning signed distance functions has been widely and successfully used for representing 2D closed-boundaries [43], [44], [45] and 3D object surfaces [46], [47], our proposed attraction field representation shares the similar spirit, but differs in two aspects: Our proposed method directly learns the attraction vectors instead of the distance maps, which can explicitly and accurately characterize the geometry of line segments, and thus eliminates the need of considering the approximation errors for numerical computation. And, our proposed formulation takes the pixels in the non-zero level sets (*i.e.*, non-edge pixels) into accounts for achieving robust line segment detection.

3 REGIONAL ATTRACTION

In this section, we provide the details of regional attraction to characterize the line segments. Concretely, we introduce a region-partition map to bridge the relationship between the line segments and regions in Section 3.1. In Section 3.2, we utilize the attraction field map (AFM) to depict the 1D geometry by using all pixels in the image lattice. In Section 3.3, we show that the attraction field map can be remapped into line segments by using a simple yet efficient squeeze module, which establishes the foundation of a deep learning based



(a) Support regions (b) Attraction vectors (c) Squeeze module

Fig. 3. A toy example illustrating a line segment map with 3 line segments, including (a) the region-partition map with 3 regions, (b) selected attraction vectors and (c) the squeeze module for obtaining line segments.

line segment detection system. Further analyses are given in Section 3.4.

3.1 Region-Partition Map

Let Λ be an image lattice (*e.g.*, 800×600). A line segment is denoted by $\mathbf{l}_i = (\mathbf{x}_i^s, \mathbf{x}_i^e)$ with the two endpoints being \mathbf{x}_i^s and \mathbf{x}_i^e (non-negative real-valued positions as sub-pixel precision is used in annotating line segments) respectively. The set of line segments in a 2D image lattice is denoted by $L = \{\mathbf{l}_1, \dots, \mathbf{l}_n\}$. For simplicity, we term the set L as a line segment map. Figure 3 illustrates a line segment map with 3 line segments in a 10×10 image lattice.

The region-partition map assigns each pixel $\mathbf{p} \in \Lambda$ to the nearest line segment in L . To this end, we use a point-to-line-segment distance function. Considering a pixel $\mathbf{p} \in \Lambda$ and a line segment $\mathbf{l}_i = (\mathbf{x}_i^s, \mathbf{x}_i^e) \in L$, we first project the pixel \mathbf{p} onto the straight line passing through \mathbf{l}_i in the continuous geometry space. If the projection point is not on the line segment, we use the closest endpoint of the line segment as the projection point. Then, we compute the Euclidean distance between the pixel and the projection point. Formally, we define the distance between \mathbf{p} and \mathbf{l}_i by

$$\begin{aligned} d(\mathbf{p}, \mathbf{l}_i) &= \min_{t \in [0,1]} d(\mathbf{p}, \mathbf{l}_i; t) \\ &= \min_{t \in [0,1]} \|\mathbf{x}_i^s + t \cdot (\mathbf{x}_i^e - \mathbf{x}_i^s) - \mathbf{p}\|_2^2, \\ t_p^* &= \arg \min_{t \in [0,1]} d(\mathbf{p}, \mathbf{l}_i; t), \end{aligned} \quad (1)$$

where the projection point is the original point-to-line projection point if $t_p^* \in (0, 1)$, and is the closest endpoint if $t_p^* = 0$ or 1 .

Then, the region in the image lattice for the line segment \mathbf{l}_i is defined by

$$R_i = \{\mathbf{p} \mid \mathbf{p} \in \Lambda; d(\mathbf{p}, \mathbf{l}_i) < d(\mathbf{p}, \mathbf{l}_j), \forall j \neq i, \mathbf{l}_j \in L\}. \quad (2)$$

It is straightforward to see that $R_i \cap R_j = \emptyset$ and $\cup_{i=1}^n R_i = \Lambda$, *i.e.*, all R_i 's form a partition of the image lattice. Figure 3(a) illustrates the region partition of line segments in a toy example. Denote by $R = \{R_1, \dots, R_n\}$ the region-partition map for a line segment map L .

3.2 Attraction Field Map for Line Segments

The region-partition map defines a region for each line segment. Consider the region R_i associated with the line segment \mathbf{l}_i . For each pixel $\mathbf{p} \in R_i$, its projection point \mathbf{p}' on \mathbf{l}_i is defined by

$$\mathbf{p}' = \mathbf{x}_i^s + t_p^* \cdot (\mathbf{x}_i^e - \mathbf{x}_i^s). \quad (3)$$

Then, we define the 2D attraction (or the projection vector) of the pixel \mathbf{p} in the support region R_i as

$$\mathbf{a}_i(\mathbf{p}) = \mathbf{p}' - \mathbf{p}, \quad (4)$$

where the attraction vector is perpendicular to the line segment if $t_p^* \in (0, 1)$ (see Figure 3(b)). The attraction mapping function in Equation (4) is applied over the image lattice as

$$\begin{aligned} \mathbf{a} : \Lambda &\rightarrow \mathbb{R}^2 \\ \mathbf{p} &\mapsto \mathbf{a}_i(\mathbf{p}), \text{ if } \mathbf{p} \in R_i. \end{aligned} \quad (5)$$

We term the mapping defined in Equation (5) as the attraction field of the line segment map L . For simplicity, we denote the attraction field map (AFM) of L as $A = \{\mathbf{a}(p) \mid p \in \Lambda\}$ by enumerating all the pixels in Λ .

Figure 2 shows examples of the x - and y -component of an attraction field map. It should be mentioned here that the attraction field map can be regarded as a variant of distance transform [48]. Generally, the distance transform is applied to binary images, where a pixel inside foreground regions is changed to measure its minimal distance to the boundary. Specially in our scenario, we use AFM to explicitly encode the geometric relationship between pixels and line segments.

Compared with the edge map (or image gradient map) used in previous work (*e.g.*, DWP [12], LSD [2] and Linelet [18]), the advantages of attraction field map can be summarized as follows:

- The edge map only approximately characterizes the line segments with very few pixels, which results in zig-zag effects. In contrast, our proposed AFM depicts the geometry of line segments in a more precise way with redundantly sampling over the line segments.
- Because each line segment is associated with a well-defined support region, our proposed representation does not need to consider the blurring effects for the nearly distributed parallel line segments.

Next, we will show how to remap the attraction field map into a set of line segments.

3.3 Squeeze Module

The squeeze module groups the attraction vectors that are adjacent to a set and the non-perpendicular vectors are used as a condition of terminating the grouping process for resulting line segments. Given an attraction field map A , we can compute the real-valued projection point for each pixel \mathbf{p} in the lattice as

$$\mathbf{v}(\mathbf{p}) = \mathbf{p} + \mathbf{a}(\mathbf{p}), \quad (6)$$

and its corresponding discretized point in the image lattice as

$$\mathbf{v}_\Lambda(\mathbf{p}) = \lfloor \mathbf{v}(\mathbf{p}) + 0.5 \rfloor, \quad (7)$$

where $\lfloor \cdot \rfloor$ represents the floor operation, and $\mathbf{v}_\Lambda(\mathbf{p}) \in \Lambda$. In addition, the attraction field map provides the normal direction (if the projected point $\mathbf{v}(\mathbf{p})$ is inside) of the line segment going through the point $\mathbf{v}(\mathbf{p})$ by

$$\phi(\mathbf{p}) = \arctan2(\mathbf{a}_y(\mathbf{p}), \mathbf{a}_x(\mathbf{p})), \quad (8)$$

where $\mathbf{a}_x(\mathbf{p})$ and $\mathbf{a}_y(\mathbf{p})$ are the x- and y- components of the vector $\mathbf{a}(\mathbf{p})$ respectively.

Then, the attraction vectors are rearranged according to their discretized projecting points, which results in a sparse map for recording the locations of possible line segments. For notation simplicity, such a sparse map is termed a line proposal map in which each pixel $\mathbf{q} \in \Lambda$ collects the attraction field vectors whose discretized projection points are \mathbf{q} . The candidate set of attraction field vectors collected by a pixel \mathbf{q} is then defined by

$$\mathcal{C}(\mathbf{q}) = \{\mathbf{a}(\mathbf{p}) \mid \mathbf{p} \in \Lambda, \mathbf{v}_\Lambda(\mathbf{p}) = \mathbf{q}\}, \quad (9)$$

where $\mathcal{C}(\mathbf{q})$'s are usually non-empty for a sparse set of pixels \mathbf{q} 's which correspond to points on the line segments. An example of the line proposal map is shown in Figure 3(c), which projects the pixels of the support region for a line segment into pixels near the line segment.

With the line proposal map, our squeeze module utilizes an iterative and greedy grouping strategy to fit line segments in the spirit of the region growing algorithm used in [2]. The pseudocode of the squeeze module is given in Algorithm 1.

- Given the current set of active pixels each of which having a non-empty candidate set of attraction field vectors, we randomly select a pixel q and one of its attraction field vector $\mathbf{a}(\mathbf{p}) \in \mathcal{C}(\mathbf{q})$. The tangent direction of the selected attraction field vectors $\mathbf{a}(\mathbf{p})$ is used as the initial direction of the line segment passing the pixel \mathbf{q} .
- Then, we search the local observation window centered at \mathbf{q} (e.g., a 3×3 window is used in this paper) to find the attraction field vectors that are aligned with $\mathbf{a}(\mathbf{p})$ with an angular distance less than a threshold τ (e.g., $\tau = 10^\circ$ used in this paper).
 - If the search fails, we discard $\mathbf{a}(\mathbf{p})$ from $\mathcal{C}(\mathbf{q})$, and further discard the pixel \mathbf{q} if $\mathcal{C}(\mathbf{q})$ becomes empty.
 - Otherwise, we grow \mathbf{q} into a set and update its direction by averaging the aligned attraction vectors. The aligned attraction vectors will be marked as used (and thus made inactive for the next round of search). For the two endpoints of the set, we recursively apply the greedy search algorithm to grow the line segment.
- Once terminated, we obtain a candidate line segment $\mathbf{l}_q = (\mathbf{x}_q^s, \mathbf{x}_q^e)$ with the support set of real-valued projection points. We fit the minimum outer rectangle using the support set. We verify the candidate line segment by checking the aspect ratio between the width and length of the approximated rectangle with respect to a predefined threshold to ensure the approximated rectangle is "thin enough". If the checking fails, we mark the pixel \mathbf{q} inactive and release the support set to be active again.

3.4 Verifying Duality and Scale Invariance

So far, we have established a dual representation to depict the geometry of line segment maps in the image lattice.

Algorithm 1 Squeeze Module

Input: The attraction field map A

- 1: Generating the line proposal map

$$\mathcal{Q} = \{\mathbf{q} \mid \mathcal{C}(\mathbf{q}) \neq \emptyset, \forall \mathbf{q} \in \Lambda\}.$$

- 2: Initialize the status $\mathcal{S}(\mathbf{p})$ for every pixel $\mathbf{p} \in \Lambda$ by,

$$\mathcal{S}(\mathbf{p}) \leftarrow \begin{cases} 0 & \text{if } \mathbf{v}_\Lambda(\mathbf{p}) \notin \Lambda \\ 1 & \text{otherwise} \end{cases}.$$

- 3: $L \leftarrow \emptyset$

4: **for** $\mathbf{p} \in \Lambda$ with $\mathcal{S}(\mathbf{p}) = 1$ **do**

5: $\theta_0 \leftarrow (\phi(\mathbf{p}) + \frac{\pi}{2}) \bmod \pi$

6: **procedure** $R \leftarrow \text{REGION_GROW}(\mathbf{q})$

7: **if** $\mathcal{S}(\mathbf{p}') = 0 \forall \mathbf{a}(\mathbf{p}') \in \mathcal{C}(\mathbf{q})$ **then**

8: Exit

9: $R \leftarrow \{\mathbf{v}(\mathbf{p})\}$

10: Initialize θ , R and \mathcal{S} from $\mathbf{a}(\mathbf{q}') \in \mathcal{C}(\mathbf{p})$ and θ_0

11: **if** Initialization failed **then** Return \emptyset

12: **for** $\mathbf{q}' \in R_\Lambda$ **do** $\triangleright R_\Lambda$ is the set of discretized points in R

13: **for** $\mathbf{a}(\mathbf{p}') \in \mathcal{C}(\mathbf{q}') \forall \mathbf{q}'' \in \mathcal{N}(\mathbf{q}')$ **do**

14: $\theta' \leftarrow (\phi(\mathbf{p}') + \frac{\pi}{2}) \bmod \pi$

15: $R' \leftarrow \emptyset$

16: **if** $\text{dis}(\theta, \theta') < \tau$ **then**

17: Average θ with θ'

18: $\mathcal{S}(\mathbf{q}') \leftarrow 0$

19: $R' \leftarrow R' \cup \{\mathbf{v}(\mathbf{p}')\}$

20: $R \leftarrow R \cup R'$

21: Fitting a rectangle $(\mathbf{x}_1, \mathbf{x}_2, w)$ from the point set R

22: **if** $r = w / \|\mathbf{x}_1 - \mathbf{x}_2\| < \epsilon$ **then**

23: $L \leftarrow L \cup \{\mathbf{l}_i = (\mathbf{x}_1, \mathbf{x}_2)\}$

24: Return L

25: **else**

26: $\mathcal{S}(\mathbf{p}') \leftarrow 1, \forall \mathbf{v}(\mathbf{p}') \in R$

27: Return L

Output: A set of line segments $L = \{(\mathbf{x}_i^s, \mathbf{x}_i^e)\}_{i=1}^N$

Given a line segment map L defined over the image lattice Λ , we are able to compute the corresponding attraction field map and then squeeze the AFM back to a set of line segments. In this section, we are going to verify the duality between line segments and the corresponding attraction field map. Furthermore, the scale invariance of regional attraction representations is verified.

We test the proposed regional attraction on the training split of the Wireframe dataset [12]. We first compute the attraction field map for each annotated line segment map and then compute the estimated line segment map by using the squeeze module. The verification is executed across multiple scales, varying from 0.5 to 2.0 with a step size of 0.1. The scale factor is used to control the size of attraction field maps. The estimated line segment maps are evaluated by measuring the precision and recall following the protocol provided along with the dataset. Figure 4 shows the precision-recall curves. The average precision and recall rates are above 0.99 and 0.93 respectively, thus verifying the duality between line segment maps and corresponding region-partition based attractive field maps, as well as the scale invariance of the duality. It is noteworthy that the precision will drop with the scale increases, which is prob-

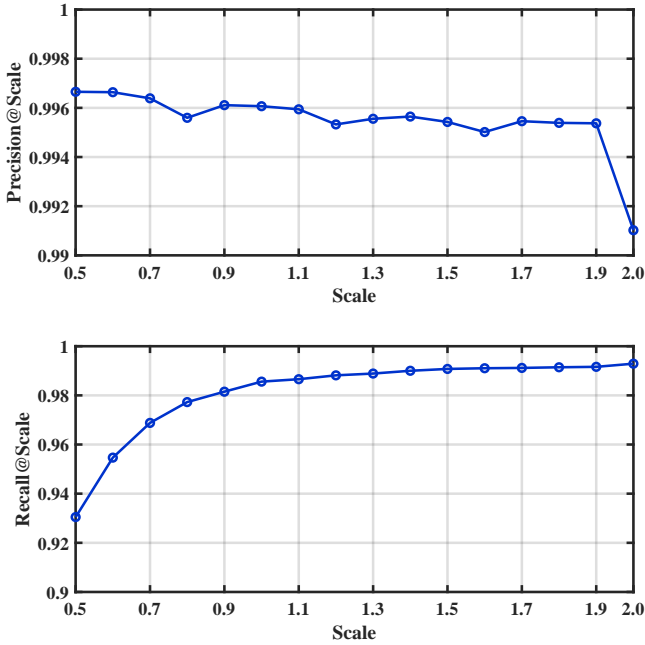


Fig. 4. Verification of the duality between line segment maps and attraction field maps, and its scale invariance.

ably caused by the fixed window size of 3×3 . When the scale increases, it is possible to induce more noisy attraction vectors, which will increase the probability to produce a bit more false positives. Despite this, the precision can be kept high as long as the attraction vectors are accurate.

Therefore, the problem of LSD can be posed as a problem of region coloring without sacrificing the performance too much (the gap is negligible). With the formulation of regional attraction, our goal is to learn ConvNets to infer the attraction field maps for input images, which we will expand in the next section.

4 DEEP LINE SEGMENT DETECTOR

In this section, we present the details of learning ConvNets for line segment detection. The proposed system takes the image I as input and outputs M line segments $L = \{l_j\}_{j=1}^M$.

4.1 AFM Parameterization

Denote by $D_{raw} = \{(I_i, L_i)\}_{i=1}^N$ the provided training dataset consisting of N pairs of raw images and annotated line segment maps. We first compute the AFM for each training image, then obtain the dual training dataset $D = \{(I_i, \mathbf{a}_i); i = 1, \dots, N\}$.

Numerical Stability and Scale Invariant Normalization.

To make the AFMs insensitive to the sizes of raw images, we adopt a simple normalization scheme. For an AFM \mathbf{a} with the height H and the width W , the size-normalization is done by

$$\mathbf{a}_x := \mathbf{a}_x / W, \quad \mathbf{a}_y := \mathbf{a}_y / H, \quad (10)$$

where \mathbf{a}_x and \mathbf{a}_y are the components of \mathbf{a} along x and y axes respectively. However, the size-normalization will make the values in \mathbf{a} quite small, which leads to numerically unstable

training. We apply a point-wise invertible value stretching transformation for the size-normalized AFM as

$$z' := S(z) = -\text{sign}(z) \cdot \log(|z| + \varepsilon), \quad (11)$$

where ε is set to $1e-6$ to avoid $\log(0)$. The inverse function $S^{-1}(\cdot)$ is defined as

$$z := S^{-1}(z') = \text{sign}(z') e^{(-|z'|)}. \quad (12)$$

For notation simplicity, denote by $\mathcal{R}(\cdot)$ the composite reverse function comprised of Equation (10) and Equation (11). We still denote by $D = \{(I_i, \mathbf{a}_i); i = 1, \dots, N\}$ the final training dataset.

4.2 Inference

Denote by $f_{\Theta}(\cdot)$ a ConvNet with the parameters Θ . As illustrated in Figure 2, for an input image I_{Λ} , the inference process of the proposed system is defined by

$$\hat{\mathbf{a}} = f_{\Theta}(I_{\Lambda}) \quad (13)$$

$$\hat{L} = \text{Squeeze}(\text{Inlier}(\mathcal{R}(\hat{\mathbf{a}}))), \quad (14)$$

where $\hat{\mathbf{a}}$ is the predicted attraction field map for the input image (the size-normalized and value-stretched one). The $\text{Inlier}(\cdot)$ operator is designed to filter out inaccurate attraction vectors. $\text{Squeeze}(\cdot)$ denotes the squeeze module and \hat{L} is the inferred line segment map.

Distribution of Regional Attraction and Outlier Removal.

Since not all the pixel predictions are accurate enough in practice, it is reasonable to remove potential outliers and only feed inliers to the squeeze module for better line segment detection. Meanwhile, our proposed regional attraction can depict every line segment with a relatively large region, the line segments can be precisely characterized even if we throw away some of attraction vectors. For the sake of computational efficiency, we analyze the magnitude of size-normalized attraction vectors on the training split of the Wireframe dataset [12] in Figure 5. The magnitude of most attraction vectors are small than $0.02 \times \min\{H, W\}$. Moreover, the networks should learn the vectors with small magnitude more accurately since a large penalty will be implicitly induced by using Equation (11).

Observing this fact, we can filter out the outliers by using the magnitude of vectors without incurring any extra computational cost. Specifically, the $\text{Inlier}(\cdot)$ operator in Equation (14) only retains the attraction vectors by

$$\text{Inlier}(\mathcal{R}(\hat{\mathbf{a}})) = \{\tilde{\mathbf{a}} \mid \tilde{\mathbf{a}} \in \mathcal{R}(\hat{\mathbf{a}}) \|\tilde{\mathbf{a}}\| \leq \gamma\}, \quad (15)$$

where γ is set to $0.02 \times \min\{H, W\}$ according to the above discussion.

4.3 An *a-trous* Residual U-Net

Benefiting from our novel formulation, the problem of LSD can be addressed with the state-of-the-art encoder-decoder networks that are widely used in dense prediction tasks. However, the existing encoder-decoder architectures are usually designed to predict a down-sampled dense map due to the characteristics of tasks. For the problem of LSD, we expect to learn high-resolution attraction field maps to preserve the geometric information as much as possible. We

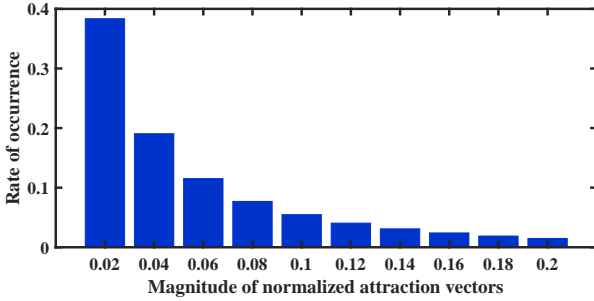


Fig. 5. Distribution of magnitudes for the size-normalized attraction vectors in the training split of the Wireframe dataset.

TABLE 1

Network architectures we use for attraction field learning. $\{\cdot\}$ and $[\cdot]$ represent the double conv in U-Net and the residual block, respectively. Inside the brackets are the shape of convolution kernels. The suffix $*$ represents the bi-linear up-sampling operator with a scaling factor of 2. The number outside the brackets is the number of stacked blocks on a stage.

stage	U-Net	<i>a-trous</i> Residual U-Net
c1	$\left\{ \begin{array}{l} 3 \times 3, 64 \\ 3 \times 3, 64 \end{array} \right\}$	$3 \times 3, 64, \text{ stride } 1$
	$2 \times 2 \text{ max pool, stride } 2$	$3 \times 3 \text{ max pool, stride } 2$
c2	$\left\{ \begin{array}{l} 3 \times 3, 128 \\ 3 \times 3, 128 \end{array} \right\}$	$\begin{array}{l} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{array} \times 3$
c3	$2 \times 2 \text{ max pool, stride } 2$	$\begin{array}{l} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{array} \times 4$
	$\left\{ \begin{array}{l} 3 \times 3, 256 \\ 3 \times 3, 256 \end{array} \right\}$	
c4	$2 \times 2 \text{ max pool, stride } 2$	$\begin{array}{l} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{array} \times 6$
	$\left\{ \begin{array}{l} 3 \times 3, 512 \\ 3 \times 3, 512 \end{array} \right\}$	
c5	$2 \times 2 \text{ max pool, stride } 2$	$\begin{array}{l} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{array} \times 3$
	$\left\{ \begin{array}{l} 3 \times 3, 512 \\ 3 \times 3, 512 \end{array} \right\}$	
d4	$\left\{ \begin{array}{l} 3 \times 3, 256 \\ 3 \times 3, 256 \end{array} \right\} *$	ASPP $\begin{array}{l} 1 \times 1, 256; 1 \times 1, 256 \\ 3 \times 3, 512 \\ 1 \times 1, 512 \end{array} *$
d3	$\left\{ \begin{array}{l} 3 \times 3, 128 \\ 3 \times 3, 128 \end{array} \right\} *$	$\begin{array}{l} 1 \times 1, 128; 1 \times 1, 128 \\ 3 \times 3, 256 \\ 1 \times 1, 256 \end{array} *$
d2	$\left\{ \begin{array}{l} 3 \times 3, 64 \\ 3 \times 3, 64 \end{array} \right\} *$	$\begin{array}{l} 1 \times 1, 64; 1 \times 1, 64 \\ 3 \times 3, 128 \\ 1 \times 1, 128 \end{array} *$
d1	$\left\{ \begin{array}{l} 3 \times 3, 64 \\ 3 \times 3, 64 \end{array} \right\} *$	$\begin{array}{l} 1 \times 1, 32; 1 \times 1, 32 \\ 3 \times 3, 64 \\ 1 \times 1, 64 \end{array} *$
output	$1 \times 1, \text{ stride } 1, \text{ w.o. BN and ReLU}$	

achieve this by changing the stride of the conv1 layer to 1 in U-Net architecture to ensure that the output feature map has the same size as the input image. Based on this, we further adopt the ResBlock [49] and ASPP [50] modules to improve the learning ability of U-Net, which is termed as *a-trous* Residual U-Net.

Table 1 shows the configurations of U-Net and *a-trous* Residual U-Net. The network consists of 5 encoder and 4 decoder stages indexed by $c1, \dots, c5$ and $d1, \dots, d4$ respectively.

- For U-Net, the double conv operator, which contains two convolution layers, is applied and denoted as $\{\cdot\}$. The $\{\cdot\}*$ operator of d_i stage upscales the output feature map of its last stage, then we concatenate it with the feature map of c_i stage before applying the double conv operator.
- For the *a-trous* Residual U-Net, we replace the double

conv operator with the Residual block, denoted as $[\cdot]$. In contrast to ResNet, we use the plain convolution layer with a 3×3 kernel and a stride of 1. Similar to $\{\cdot\}*$, the operator $[\cdot]*$ also takes the input from two sources and upscales the feature of the first input source. The first layer of $[\cdot]*$ contains two parallel convolution operators to reduce the depth of feature maps, then we concatenate them for the subsequent computations. In d_4 stage, we use 4 ASPP operators with the output channel size equal to 256 and a dilation rate of 1, 6, 12, 18, then concatenate their outputs. The output stage is a 1×1 convolution with a stride of 1 without batch normalization [51] and ReLU [52] for the AFM prediction.

4.4 Training and Testing

We follow the standard deep learning protocol to estimate the parameters Θ . We adopt the l_1 loss function in training, defined as

$$\ell(\hat{\mathbf{a}}, \mathbf{a}) = \sum_{(x,y) \in \Lambda} \|\mathbf{a}(x, y) - \hat{\mathbf{a}}(x, y)\|_1. \quad (16)$$

Baseline Implementation. We train the networks from scratch on the training set of the Wireframe dataset [12]. To make a fair comparison, we follow the standard data augmentation strategy from [12] to enrich the training samples with image domain operations including mirroring and flipping upside-down. The Adam optimizer is used here for training with the default settings in PyTorch ($\beta_1 = 0.9$ and $\beta_2 = 0.99$) and the initial learning rate is set to 0.001. We train all of the networks with 200 epochs and the learning rate is decayed with the factor of 0.1 after 180 epochs. In the training phase, we resize the images to 320×320 and then generate the attraction field maps from the resized line segment annotations to form the mini batches. As discussed in Section 3, the rescaling step with reasonable factors will not affect the results. The mini-batch sizes for the two networks are 16 and 4 respectively due to GPU memory limitations.

In the inference stage, a test image is also resized to 320×320 as the input of the network. Then, we use the squeeze module to convert the learned regional attraction into line segments. Since the line segments are insensitive to scale, we can directly resize them to original image size without sacrificing accuracy. The squeeze module is implemented with C++ on CPU.

5 EXPERIMENTS

In this section, we evaluate the proposed line segment detector and compare with existing state-of-the-art line segment detectors [2], [3], [12], [18], [33] on the Wireframe dataset [12] and YorkUrban dataset [4]. The source code of this paper will be released at <https://cherubicxn.github.io/afmplusplus/>.

5.1 Datasets and Evaluation Metrics

Wireframe Dataset. The Wireframe dataset [12] was proposed for line segment detection and junction detection. The images in this dataset are all taken in indoor scenes

(e.g., kitchens and bedrooms) and outdoor man-made environments (e.g., yards and houses). To the best of our knowledge, this dataset is the largest dataset (containing 5000 training samples and 462 testing samples) with high-quality line segment annotations to date. The average resolution of the images in this dataset is 480×405 . Since this dataset focuses on scene structures, the line segments on the boundary of irregular or curved objects (e.g., pillows and sofa) are not annotated. In this paper, we train our line segment detector on the training split of this dataset and evaluate the performance on the testing split for comparison.

YorkUrban Dataset. The YorkUrban dataset [4] was initially proposed for edge-based Manhattan frame estimation and consists of 102 images (45 indoor and 57 outdoor) with a size of 640×480 . The dataset is randomly split into a training set and a testing set with 51 images each. For each image in this dataset, the ground truth line segments are annotated with sub-pixel precision. Since this dataset was designed for Manhattan world estimation, some of the line segments that are not associated with any vanishing point are not annotated. In this paper, we only use the testing split of this dataset for evaluation and performance comparison. We do not train or fine tune the model on this dataset.

Evaluation Protocol. We follow the evaluation protocol from the DWP [12] to make a comparison. First, the proposed method is evaluated on the testing split of Wireframe dataset [12]. To validate the ability of generalization, we also evaluate it on the YorkUrban dataset [4]. All the methods are evaluated quantitatively using precision and recall following [12], [30]. The precision rate indicates the proportion of positive detections among all of the detected line segments while recall reflects the fraction of detected line segments among all in the scene. The detected and ground-truth line segments are digitized into the image domain and we define the “positive detection” pixel-wised. The line segment pixels within 0.01 of the image diagonal are regarded as positive. After obtaining the precision (P) and recall (R), we compare the performance of algorithms using the F-measure $F = 2 \cdot \frac{P \cdot R}{P + R}$.

5.2 Main Results for Comparison

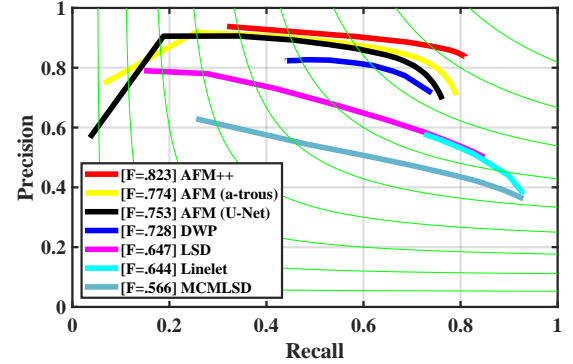
We compare our proposed method with AFM [33], DWP¹ [12], Linelet² [18], the Markov Chain Marginal Line Segment Detector³ (MCMLSD) [3] and the Line Segment Detector (LSD)⁴ [2]. The source codes of those methods are obtained from the links provided by the respective authors.

Threshold Configuration. In our proposed method, we use the aspect ratio to filter out false detections. Here, we vary the threshold of the aspect ratio in the range (0, 1] with the step size $\Delta\tau = 0.02$. For comparison, the LSD [2] is evaluated with the $-\log(\text{NFA})$ in $0.01 \times \{1.75^0, \dots, 1.75^{19}\}$ for *a-contrario* validation where NFA is the number of false alarms. In addition, Linelet [18] uses the same thresholds as

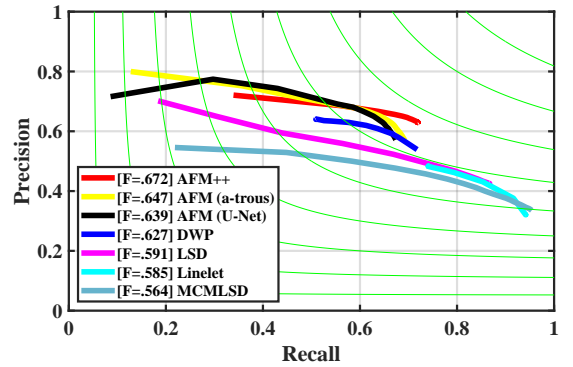
1. <https://github.com/huangkuns/wireframe>
 2. <https://github.com/NamgyuCho/Linelet-code-and-YorkUrban-LineSegment-DB>
 3. <http://www.elderlab.yorku.ca/resources/>
 4. <http://www.ipol.im/pub/art/2012/gjmr-lsd/>

TABLE 2
Comparison of the F-measure with the state-of-the-art methods on the Wireframe and YorkUrban datasets. The last column reports the average inference speed (*frames-per-second*, FPS) on the Wireframe dataset.

Methods	Wireframe dataset	YorkUrban dataset	FPS
LSD [2]	0.647	0.591	19.6
MCMLSD [3]	0.566	0.564	0.2
Linelet [18]	0.644	0.585	0.14
DWP [12]	0.728	0.627	2.24
AFM (U-Net) [33]	0.753	0.639	10.3
AFM (<i>a-trous</i>) [33]	0.774	0.647	6.6
AFM++ (<i>a-trous</i>)	0.823	0.672	8.0



(a) PR curves on the Wireframe dataset



(b) PR curves on the YorkUrban dataset

Fig. 6. The PR curves of different line segment detection methods on the Wireframe dataset [12] and YorkUrban dataset [4].

the LSD to filter out false detections. For MCMLSD [3], we use the top- K detected line segments for evaluation. With regard to the evaluation of DWP [12], we follow the default threshold setting for junction detection and line heat map binarization. In detail, the confidence threshold for both the junction localization and the junction orientation are set to 0.5. The thresholds for line heat map binarization are set to [2, 6, 10, 20, 30, 50, 80, 100, 150, 200, 250, 255] to detect line segments.

Precision & Recall. We first evaluate the proposed method on the Wireframe dataset [12]. The precision-recall curves and the F-measure are presented in Figure 6(a) and Table 2, respectively. As is shown, the proposed AFM++ sets a new state-of-the-art performance, that is the F-measure of 0.823.



Fig. 7. Some results of line segment detection of different approaches on the Wireframe [12] dataset. From top to bottom: LSD [2], MCMLSD [3], Linelet [18], DWP [12], AFM [33] with the *a-trous* Residual U-Net and AFM++ proposed in this paper.

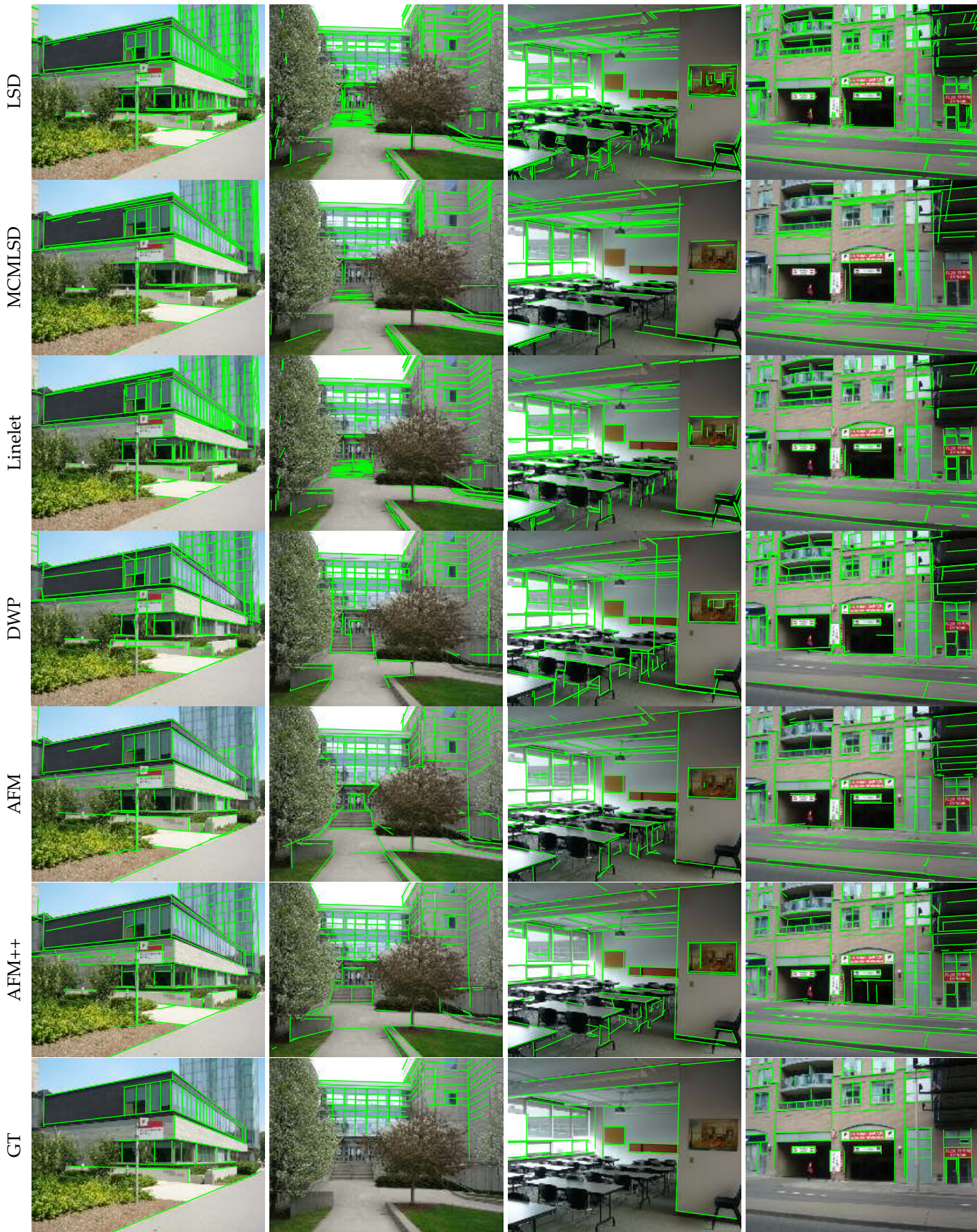


Fig. 8. Some results of line segment detection of different approaches on the YorkUrban [4] datasets. From top to bottom: LSD [2], MCMLSD [3], Linelet [18], DWP [12], AFM [33] with the *a-trous* Residual U-Net and AFM++ proposed in this paper.

This achievement is dramatically better than DWP [12], with a performance improvement of approximately 10 percent. Compared with the previous version AFM [33], AFM++ improves the F-measure by 5 percent on this dataset. This demonstrates the usefulness of outlier removal module and better optimizer, which will be further discussed below.

Furthermore, we also evaluate our proposed approach on the YorkUrban dataset [4] and the performance comparison is given in Table 2 and Figure 6(b). Consistent with the results on the Wireframe dataset, our work (AFM and AFM++) beats those representative algorithms by a large margin. In particular, AFM++ achieves an F-measure of 0.672, advancing the state-of-the-art performance by 4.5 percent (over 0.627 reported by DWP [12]). Note that the YorkUrban dataset only focuses on the Manhattan frame estimation, which results in that some line segments in the images are not labeled. Therefore, one may observe that the performance on the YorkUrban dataset is generally lower than that on the Wireframe dataset.

Visualization and Discussion. We visualize the line segments detected by different methods in Figure 7 for the Wireframe dataset and Figure 8 for the YorkUrban dataset, respectively. The threshold configurations for visualization are as follows:

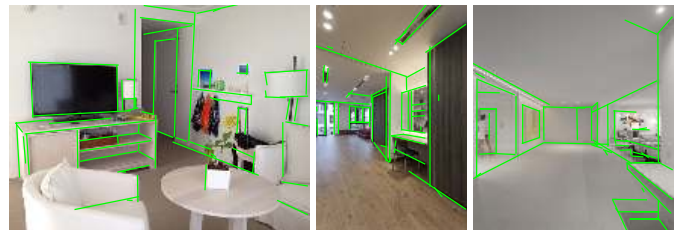
- 1) The *a-contrario* validation of LSD and Linelet are set to $-\log \epsilon = 0.01 \cdot 1.75^8$;
- 2) The top 90 line segments detected by MCMLSD are visualized;
- 3) The threshold of the line heat map is 10 for DWP;
- 4) The aspect ratio is set to 0.2 for AFM [33] and AFM++.

As we can see from Figure 7 and Figure 8, the deep learning based approaches, including AFM++, AFM [33] and DWP [12], generally perform better on the two datasets than the other approaches, including LSD [2], MCMLSD [3] and Linelet [18], since they utilize the global information to capture the low-contrast regions while suppressing the false detections in the edge-like texture regions. The approaches [2], [3], [18] only infer line segments from local features, thus causing incomplete detection results and a number of false detections even with powerful validation processes. Although the overall F-measure of LSD [2] is slightly better than Linelet [18], the qualitative visualizations of Linelet [18] are cleaner.

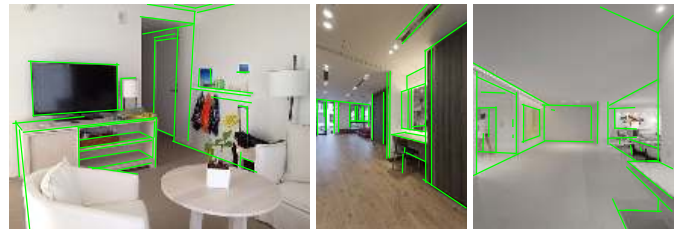
For deep learning based approaches, AFM++ significantly outperforms AFM [33] and DWP [12] with fewer false detections and more accurate line segment localization. Compared with AFM, we are able to resolve the overshooting issue in the endpoint estimation because of the better regional attraction learning and outlier removal module. In contrast to DWP [12], AFM and AFM++ get rid of junction detection and line heat map prediction, thus resolving the local ambiguity for line segment detection in an efficient way. Since DWP [12] requires junctions for line segment detection, the results are not well localized by the inaccurately estimated orientation of junctions. Besides, the incorrect edge pixels will mislead the merging module in DWP [12] to generate false detections by mistakenly connecting some junction pairs.

Inference Speed. We compare the inference speed of the aforementioned algorithms on the Wireframe dataset. The time cost is calculated over the entire testing dataset and the average frames-per-second (FPS) is reported in the last column of Table 2. All the experiments were conducted on a PC workstation equipped with an Intel Xeon E5-2620 2.10 GHz CPU and 4 NVIDIA Titan X GPU devices. Only one GPU is used and the CPU programs are executed in a single thread.

As reported in Table 2, in addition to the state-of-the-art performance, our method is also computationally inexpensive. Benefiting from the simplicity of our novel formulation, AFM-based methods run faster than all the other methods except for LSD. AFM (U-Net) is the fastest among the AFM based approaches, and is the second compared to LSD. DWP [12] spends much time for junction and line heat map merging. Meanwhile, our method resizes the input images into 320×320 and then transforms the output line segments to the original size without loss of information, which further reduces the computational cost. Compared with AFM [33], the outlier removal module in AFM++ only retains well-estimated attraction vectors, which also improves the computational speed.



(a) Results by the model trained on 320×320 samples



(b) Results by the model trained on 512×512 samples
Fig. 9. Line segments detected on images of different resolutions.

5.3 Interpretability

In this section, we are going to discuss what are the network learned. Generally speaking, the learning target of our network is the attraction field map, however, it is hard to understand the learning process by simply observing the predictions of attraction field map. Alternatively, we use Guided Backpropagation [53] to visualize which pixels are important for the attraction field prediction and line segment detection. Guided Backpropagation [53] interprets pixels' importance for prediction by calculating the gradient flow from the prediction layer to the input images. The magnitude of the gradients flowed back to the input image indicates the change of the pixels that will affect the final prediction. Different from the vanilla backpropagation, Guided Backpropagation only retains the positive gradients in the ReLU layer and passes the modified gradients to the



Fig. 10. Visualized interpretation of the learned network. The top row displays some examples of images and the bottom row displays corresponding saliency maps obtained by Guided Backpropagation [53].

TABLE 3

Performance change by increasing image resolution, using a better optimization method and adding the outlier removal module.

Backbone/ Optimizer	resolution	Outlier removal	Wireframe dataset	YorkUrban dataset	FPS
<i>a-trous</i> / SGD	320 × 320	w/o	0.774	0.647	6.6
	512 × 512	w/o	0.794	0.660	2.3
	320 × 320	w	0.807	0.663	8.0
	512 × 512	w	0.826	0.674	3.5
<i>a-trous</i> / Adam	320 × 320	w/o	0.792	0.659	6.6
	512 × 512	w/o	0.802	0.665	2.3
	320 × 320	w	0.823	0.672	8.0
	512 × 512	w	0.831	0.680	3.5

previous layer. The gradients flowed to the input images are used for visualization. As discussed in [53], the gradients with positive values indicate corresponding pixels with high influence for prediction. Accordingly, we use the positive gradient maps (with respect to the input image) as the saliency maps for visualization. In the computation, the gradients for the last layer are set to 1. As shown in Figure 10, it is interesting to see that the learned network will automatically perceive the geometric structures of the input image. This visualization results could help us to understand why the convolutional neural networks can be used for line segment detection.

5.4 Tweaks and Discussion for Further Improvements

In this section, we explore how to further improve the performance with some useful tweaks. In detail, we train the *a-trous* Residual U-Net with higher resolution images (512 × 512) and different optimization methods. Furthermore, the outlier removal module with statistical priors is verified to be effective for line segment detection. By combining these tweaks, we obtain a higher F-measure, that is, 0.831 on the Wireframe dataset.

Training with Higher Resolutions. Since we adopt the encoder-decoder architecture for the attraction field learning, it is interesting to determine whether higher-resolution samples are conducive to extracting finer features for AFM learning. In this experiment, we increase the resolution of the training samples from 320 × 320 (default setting) to 512 × 512 for training and testing, while keeping the other settings the same as the previous configuration. The results are reported in Table 3.

Generally speaking, increasing the resolution of training samples increases the accuracy of LSD evidently. For example, when using *a-trous* as the backbone and stochastic gradient descent (SGD) as the optimizer, the F-measure is improved by about 2 percent (from 0.773 to 0.794). For qualitative evaluations, some results of the line segments detected with different resolution samples (320 × 320, 512 × 512) are plotted in Figure 9. As is shown, the higher resolution of the training samples is, more complete results with fewer false detection rate are obtained. However, the increased image size will slow down the inference speed.

Better Optimization Method. It has been demonstrated that the Adam optimizer performs better than SGD in image classification [34]. Inspired by this, we use the Adam optimizer to optimize the model rather than SGD adopted in AFM [33]. As shown in Table 3, the Adam optimizer improves the F-measure by 2 percent on the Wireframe dataset compared with SGD. The Adam optimizer will not increase the computational cost in testing phase.

Outlier Removal. Although the proposed regional attraction uses all the learned attraction vectors, the ConvNets cannot ensure that the vectors in each pixel can be predicted as accurately as possible. Besides, our numerical stable normalization in Equation (11) will implicitly give the attraction vectors with smaller magnitude a large penalty.

Therefore, the outlier removal module with statistical priors can filter out the inaccurately estimated attraction vectors in an efficient way. In this experiment, we filter out the attraction vectors of which the ℓ_2 norm is greater than $\gamma = 0.02 \times \min(H, W)$. Recall that H and W are the size of training samples. As reported in Table 3, the outlier removal improves the F-measure performance using the same training resolution and optimizer. Meanwhile, the outlier removal can reduce the amount of attraction vectors for the squeeze module and slightly improves the inference speed.

6 CONCLUSION AND FURTHER WORK

In this paper, we proposed a method of representing and characterizing the 1D geometry of line segments by using all pixels in the image lattice. The problem of line segment detection (LSD) is then posed as a problem of region coloring which is addressed by learning convolutional neural networks. The region coloring formulation of LSD harnesses the best practices developed in deep learning based semantic segmentation methods such as the encoder-decoder architecture and the *a-trous* convolution. In the experiment, our method is tested on two widely used LSD benchmarks, *i.e.*, the Wireframe [12] and the YorkUrban [4] datasets, with state-of-the-art performance obtained in both accuracy and speed.

In the future, we will exploit how to simultaneously detect line segments and junctions together in a convolutional neural network. Considering the simplicity and superior performance, we hope that the new perspective provided in this work can facilitate and motivate better line segment detection and geometric scene understanding. Furthermore, we will study the application of the proposed line segment detector to many up-level vision tasks such as Structure-from-Motion (SfM), SLAM and single-view 3D reconstruction.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China under Grant 61922065, Grant 61771350 and Grant 41820104006. This work was also supported in part by EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1. Nan Xue was also supported by China Scholarship Council. T. Wu was supported in part by ARO Grant W911NF1810295 and NSF IIS-1909644. The views presented in this paper are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

- [1] J. B. Burns, A. R. Hanson, and E. M. Riseman, "Extracting straight lines," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 4, pp. 425–455, 1986.
- [2] R. G. von Gioi, J. Jakubowicz, J. M. Morel, and G. Randall, "LSD: A Fast Line Segment Detector with a False Detection Control," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, 2010.
- [3] E. J. Almazan, R. Tal, Y. Qian, and J. H. Elder, "MCMLSD: A Dynamic Programming Approach to Line Segment Detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [4] P. Denis, J. H. Elder, and F. J. Estrada, "Efficient Edge-Based Methods for Estimating Manhattan Frames in Urban Imagery," in *European Conference on Computer Vision (ECCV)*, 2008, pp. 197–210.
- [5] O. D. Faugeras, R. Deriche, H. Mathieu, N. Ayache, and G. Randall, "The Depth and Motion Analysis Machine," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 6, no. 2&3, pp. 353–385, 1992.
- [6] L. Duan and F. Lafarge, "Image Partitioning Into Convex Polygons," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3119–3127.
- [7] Z. Yu, X. Gao, H. Lin, A. Lumsdaine, and J. Yu, "Line Assisted Light Field Triangulation and Stereo Matching," in *IEEE International Conference on Computer Vision (ICCV)*, 2013.
- [8] C. Zou, A. Colburn, Q. Shan, and D. Hoiem, "LayoutNet: Reconstructing the 3D Room Layout from a Single RGB Image," *CoRR*, vol. abs/1803.08999, 2018.
- [9] Y. Zhao and S.-C. Zhu, "Scene Parsing by Integrating Function, Geometry and Appearance Models," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 3119–3126.
- [10] C. Xu, L. Zhang, L. Cheng, and R. Koch, "Pose Estimation from Line Correspondences: A Complete Analysis and a Series of Solutions," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1209–1222, 2017.
- [11] T. Xiang, G. Xia, X. Bai, and L. Zhang, "Image stitching by line-guided local warping with global similarity constraint," *Pattern Recognition*, vol. 83, pp. 481–497, 2018.
- [12] K. Huang, Y. Wang, Z. Zhou, T. Ding, S. Gao, and Y. Ma, "Learning to Parse Wireframes in Images of Man-Made Environments," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
- [13] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111–122, 1981.
- [14] J. Matas, C. Galambos, and J. Kittler, "Robust detection of lines using the progressive probabilistic hough transform," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 119–137, 2000.
- [15] K. Yang, S. S. Ge, and H. He, "Robust line detection using two-orthogonal direction image scanning," *Computer Vision and Image Understanding*, vol. 115, no. 8, pp. 1207–1222, 2011.
- [16] D. Shi, J. Gao, P. S. Rahmdel, M. Antolovich, and T. Clark, "UND: unite-and-divide method in fourier and radon domains for line segment detection," *IEEE Trans. Image Processing*, vol. 22, no. 6, pp. 2500–2505, 2013.
- [17] R. F. C. Guerreiro and P. M. Q. Aguiar, "Connectivity-enforcing hough transform for the robust extraction of line segments," *IEEE Trans. Image Processing*, vol. 21, no. 12, pp. 4819–4829, 2012.
- [18] N. G. Cho, A. Yuille, and S. W. Lee, "A Novel Linelet-Based Representation for Line Segment Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 40, no. 5, pp. 1195–1208, 2018.
- [19] A. Desolneux, L. Moisan, and J. Morel, "Meaningful alignments," *Int. J. Compt. Vision*, vol. 40, no. 1, pp. 7–23, 2000.
- [20] —, "A grouping principle and four applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 4, pp. 508–513, 2003.
- [21] S. Xie and Z. Tu, "Holistically-Nested Edge Detection," in *IEEE International Conference on Computer Vision (ICCV)*, 2015.
- [22] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.
- [23] K.-K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. V. Gool, "Convolutional Oriented Boundaries," in *European Conference on Computer Vision (ECCV)*, 2016.
- [24] I. Kokkinos, "Pushing the Boundaries of Boundary Detection Using Deep Learning," in *International Conference on Learning Representations (ICLR)*, 2016.
- [25] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai, "Richer Convolutional Features for Edge Detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [26] J. Kittler, "On the accuracy of the sobel edge detector," *Image Vision Comput.*, vol. 1, no. 1, pp. 37–42, 1983.
- [27] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 207, no. 1167, pp. 187–217, 1980.
- [28] J. F. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.
- [29] V. Torre and T. A. Poggio, "On edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 2, pp. 147–163, 1986.
- [30] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture

- cues," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 530–549, 2004.
- [31] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2015.
- [32] Q. Hou, J. Liu, M.-M. Cheng, A. Borji, and P. H. S. Torr, "Three Birds One Stone: A Unified Framework for Salient Object Segmentation, Edge Detection and Skeleton Extraction," in *European Conference on Computer Vision (ECCV)*, 2018.
- [33] N. Xue, S. Bai, F. Wang, T. Wu, G.-S. Xia, and L. Zhang, "Learning attraction field representation for robust line segment detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [34] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.
- [35] Y. Furukawa and Y. Shinagawa, "Accurate and robust line segment extraction by analyzing distribution around peaks in hough space," *Computer Vision and Image Understanding*, vol. 92, no. 1, pp. 1–25, 2003.
- [36] Z. Xu, B. Shin, and R. Klette, "Accurate and robust line segment extraction using minimum entropy with hough transform," *IEEE Trans. Image Processing*, vol. 24, no. 3, pp. 813–822, 2015.
- [37] —, "Closed form line-segment extraction using the hough transform," *Pattern Recognition*, vol. 48, no. 12, pp. 4012–4023, 2015.
- [38] —, "A statistical method for line segment detection," *Computer Vision and Image Understanding*, vol. 138, pp. 61–73, 2015.
- [39] R. G. von Gioi, J. Jakubowicz, J. Morel, and G. Randall, "On straight line segment detection," *Journal of Mathematical Imaging and Vision*, vol. 32, no. 3, pp. 313–347, 2008.
- [40] N. Xue, G.-S. Xia, X. Bai, L. Zhang, and W. Shen, "Anisotropic-Scale Junction Detection and Matching for Indoor Images." *IEEE Trans. Image Processing*, vol. 27, no. 1, pp. 78 – 91, 2018.
- [41] G.-S. Xia, J. Delon, and Y. Gousseau, "Accurate Junction Detection and Characterization in Natural Images," *International Journal of Computer Vision*, vol. 106, no. 1, pp. 31–56, 2014.
- [42] K. K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. Van Gool, "Convolutional Oriented Boundaries: From Image Segmentation to High-Level Tasks," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 819–833, 2018.
- [43] A. M. Andrew, "Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science," by J.A. Sethian, Cambridge University Press, Cambridge, UK, 2nd edn 1999 (first published 1996 as *Level Set Methods*) xviii + 420 pp., ISBN (paperback) 0-521-64557-3, (hardback) 0-521-64204-3 (pbk, £18.95)," *Robotica*, vol. 18, no. 1, pp. 89–92, 2000.
- [44] V. Estellers, D. Zosso, R. Lai, S. J. Osher, J. Thiran, and X. Bresson, "Efficient algorithm for level set method preserving distance function," *IEEE Trans. Image Processing*, vol. 21, no. 12, pp. 4722–4734, 2012.
- [45] J. Yuan, "Learning building extraction in aerial scenes with convolutional networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 11, pp. 2793–2798, 2018.
- [46] M. Zollhöfer, A. Dai, M. Innmann, C. Wu, M. Stamminger, C. Theobalt, and M. Nießner, "Shading-based refinement on volumetric signed distance functions," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 96:1–96:14, 2015.
- [47] J. J. Park, P. Florence, J. Straub, R. A. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 165–174.
- [48] R. Kimmel, N. Kiryati, and A. M. Bruckstein, "Sub-pixel distance maps and weighted distance transforms," *Journal of Mathematical Imaging and Vision*, vol. 6, no. 2-3, pp. 223–233, 1996.
- [49] K. He, X. Zhang, S. Ren, and Sun Jian, "Deep Residual Learning for Image Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [50] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," in *European Conference on Computer Vision (ECCV)*, 2018.
- [51] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015.
- [52] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *International Conference on Machine Learning (ICML)*, 2010.
- [53] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller, "Striving for simplicity: The all convolutional net," in *International Conference on Learning Representations (ICLR), Workshop Track Proceedings*, 2015.