Supplemental Material

Baojiang Zhong*, *Senior Member, IEEE* and Kai-Kuang Ma[†], *Life Fellow, IEEE*
*School of Computer Science and Technology, Soochow University, Suzhou 215006, China
[†]School of Electrical and Electronic Engineering, Nanyang Technological University, 639798, Singapore

Abstract—In this supplemental material, an illustrative example of scale-space line segment detection is demonstrated firstly, which can help understand each step of our proposed scale-space approach. Then, the bugs in Cho et al's method for objective evaluation are reported and fixed. Finally, a subjective performance comparison conducted between each seed detector and the corresponding scale-space detector developed with the proposed scale-space approach is presented.

Index Terms—Scale-space, line segment detection, multi-scale fusion, scale selection, Bayesian inference.

1 An Illustrative Example

To help understand each step of our proposed scale-space approach described in Section 4 of the paper, an illustrative example is presented, which was, in fact, embedded in Figure 7 of the paper (i.e., a global picture of the proposed approach).

1.1 A Glance

Figure 1 provides a glance of our illustrative example, where the Linelet [1] is taken as the *seed* line segment detector and incorporated into our proposed approach to develop the *scale-space* detector.

Figure 1(a) presents the test image. We use this image for illustration since its line segments are well defined and hence the detection results are easy to be evaluated visually. For simulating a real-world environment, a small amount of zero-mean Gaussian white noise (variance $= 1 \times 10^{-4}$) has been added to the original noise-free image. Figure 1(b) shows the detected line segments of the Linelet [1]. Note that this result does not represent state-of-the-art performance of the current detectors including the Linelet itself. However, by using this illustrative experiment, the fragmentation problem encountered by the existing detectors, as well as poor orientation accuracy of each produced line fragment, can be distinctly disclosed. These problems could happen even on 'noise-free' images, where visually negligible noise generated from various aspects (such as image digitization, compression attack, and low illumination) is already able to degrade the performance of existing detectors. Figure 1(c) shows the detection performance of the scale-space detector developed upon the Linelet by using our approach.

1.2 Line Segment Detection at Multiple Scales

The scale sequence $\{\sigma_j\}$ for performing our scale-space approach is determined by using $\sigma_1 = 0.25$, $\omega = \sqrt{2}$, and $n_s = 8$ (the setting of n_s is for illustration only). The seed detector, Linelet [1], is then implemented to detect line segments at each scale individually. The detection results are presented in Figure 2. The line segment set produced at σ_j is denoted as M_j, for $j = 1, 2, \cdots, n_s$.

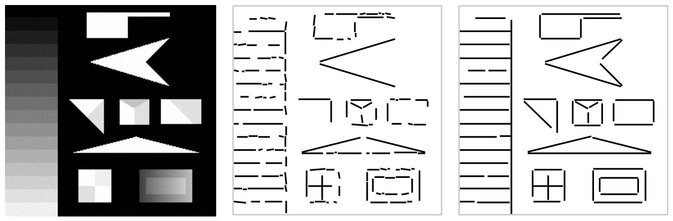

(a) (b) (c)

Fig. 1. A glance of the illustrative example. (a) The input test image; (b) Performance of the Linelet [1]; (c) Performance of the scale-space Linelet developed by using our approach.

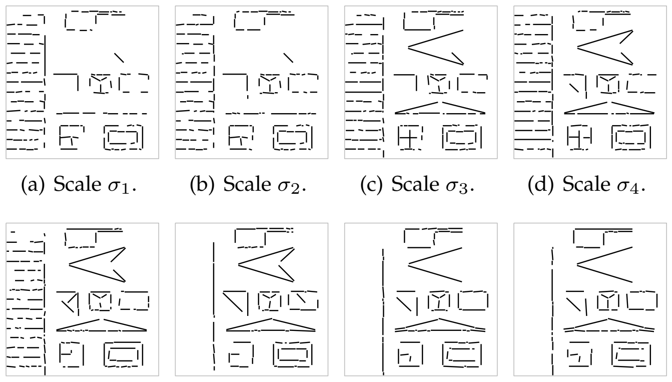

(a) Scale σ_1. (b) Scale σ_2. (c) Scale σ_3. (d) Scale σ_4.
(e) Scale σ_5. (f) Scale σ_6. (g) Scale σ_7. (h) Scale σ_8.

Fig. 2. Line segment sets produced by the Linelet [1] at the 8 selected scales, respectively.

1.3 Best-Response Scale Identification

By using our proposed model (referring to (36) in the paper), σ_4 is identified as the best-response scale, i.e., $\sigma_c = \sigma_4$.

1.4 Adaptive Scale Range Determination

Since $\sigma_c = \sigma_4$, the scale range for conducting line segment fusion is determined as $[\sigma_2, \sigma_6]$.

1.5 Line Segment Fusion Across Scales

The estimation process of $P(l|\ell)$ (the likelihood density that measures the reliability of l with respect to *line orientation*) is

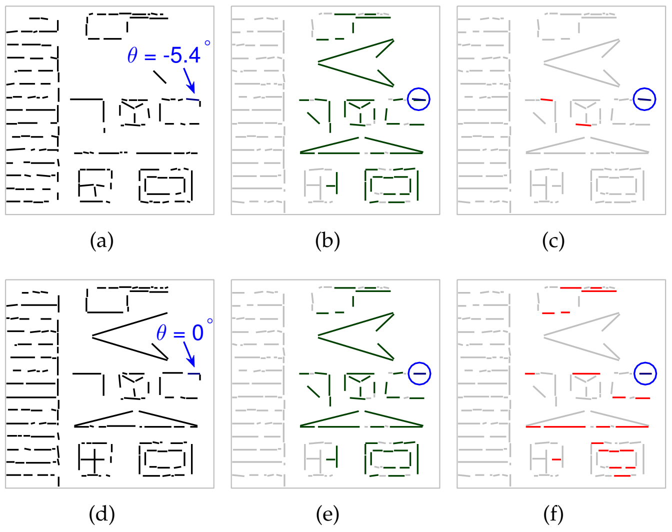


Fig. 3. The estimation of likelihood density, i.e., $P(l|\ell)$ (best seen in color): (a) the line segment set M_2, where the line segment pointed by an arrow, denoted as l, is to be processed; (b) l is overlaid onto the line segment set M_4, where the highlighted line segments (in green) are the nearest 50 neighbors of l; (c) the newly highlighted line segments (in red) are those neighbors whose orientation difference with l is less than $\tau_\theta = 4°$. Same as (a)-(c), (d)-(e) demonstrates the likelihood density estimation of a line segment in M_3.

TABLE 1
Computation of the line segment reliability (i.e., $R(l)$) for the two line segments highlighted in Figures 3(a) and 3(d), respectively.

Line Segments	$\|l\|_2$	$P(l\|\ell)$	$R(l)$
Highlighted in Figure 3(a)	16.03	0.017	0.278
Highlighted in Figure 3(d)	15.03	0.207	3.113

demonstrated in Figure 3, where l is the input line segment and ℓ denotes its ground-truth counterpart.

Figure 3(a) presents the line segment set M_2. The line segment pointed by an arrow, denoted as l, has a line orientation of $-5.4°$ and a line length of 16.03 pixels. For this line segment, the value of $P(l|\ell)$ is estimated as follows. First, l is overlaid on the line segment set produced at the best-response scale, i.e., M_4. Then, its nearest 50 neighbors in M_4 are identified, as highlighted in Figure 3(b) (in green). Among these 50 line segments, only two of them have their orientation difference compared with l smaller than the angle tolerance $\tau_\theta = 4°$, as highlighted in Figure 3(c) (in red). Finally, $P(l|\ell)$ is estimated by using our proposed model (referring to (42) in the paper). In the same way, the value of $P(l|\ell)$ is also estimated for a line segment in the set M_3, as illustrated in Figures 3(d) to 3(f).

The results of measured line segment reliability, i.e., $R(l)$, for the two line segments are documented in Table 1. The line segment highlighted in Figure 3(d), although shorter than the one highlighted in Figure 3(a), has a more precise line orientation and therefore achieves a higher reliability than the latter. This is in agreement with the perception of our human visual system.

For our illustrative example, the fused line segment sets obtained after each iteration of fusion (referring to (40) in the paper) are shown in Figure 4. The fusion procedure contains four fusion iterations in total, and thus four fused line segment sets are generated, among which the last one is the final outcome of our line segment fusion step.

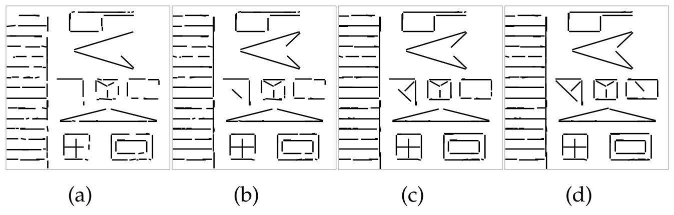


Fig. 4. Fused line segment sets produced after the first to fourth iteration of fusion. The fourth set is the outcome of the line segment fusion step, which has 124 line segments in total.

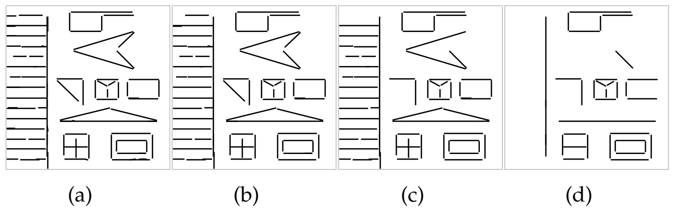


Fig. 5. Line segment validation with different values of the parameter τ_p: (a) $\tau_p = 0.4$ and 85 line segments are validated; (b) $\tau_p = 0.6$ and 72 line segments are validated; (c) $\tau_p = 0.8$ and 65 line segments are validated; (d) $\tau_p = 1.0$ and 35 line segments are validated.

1.6 Line Segment Validation in Scale Space

Figure 5 shows the line segment validation results produced under different values of the parameter τ_p, among which $\tau_p = 0.6$ is our default setting—that is, the line segment set presented in Figure 5(b) is taken as the outcome of this step. Further note that if we take $\tau_p = 0.2$, then all the line segments presented in Figure 4(d) will be accepted as true positives. To be specific, in this example, the radius of the scale range for conducting line segment fusion is $r = 2$, and hence $(2r + 1)\tau_p = 1$ when $\tau_p = 0.2$, while the persistence value of any line segment is no less than 1—that is, no line segment will be discarded as false positive.

1.7 Post-Processing

A self-fusion is performed on the line segment set presented in Figure 5(b). The result has already been shown in Figure 1(c), which is the final output of our scale-space approach.

2 Corrections on Cho et al's Method for Objective Evaluation

Cho et al [1] developed a method for conducting objective evaluation on line segment detection, and their evaluation codes are available online [2]. However, Cho et al's method has two kinds of bugs: one *technique bug* and several *code bugs*, which are reported as follows. In our paper, these bugs have all been fixed, and the corrected evaluation codes have been made available online [3].

2.1 A Technique Bug

In [1], the line segment detectors coded in C language (e.g., LSD [4] and EDlines [5]) were all incorrectly evaluated. In

TABLE 2
Objective evaluation of the Linelet [1] on the YorkUrban-LineSegment dataset by using Cho et al's evaluation codes with different orders of the detected line segments.

Line Segment Order	AP	AR	IOU	F-score
Original Order	0.3285	0.4280	0.4157	0.3671
Descending in Length	0.3376	0.4280	0.4181	0.3731
Ascending in Length	0.2938	0.4280	0.3999	0.3430

TABLE 3
Objective evaluation of the Linelet [1] on the YorkUrban-LineSegment dataset by using our corrected evaluation codes with different orders of the detected line segments.

Line Segment Order	AP	AR	IOU	F-score
Original Order	0.4478	0.4280	0.2830	0.4342
Descending in Length	0.4478	0.4280	0.2830	0.4342
Ascending in Length	0.4478	0.4280	0.2830	0.4342

fact, for the YorkUrban-LineSegment dataset constructed by Cho et al, the ground-truth line segment sets were prepared to evaluate detectors coded in Matlab. To evaluate a detector coded in C language, each endpoint of the detected line segments, (x, y), should be updated to $(x + 1, y + 1)$. This is to eliminate a difference between data formats in storing the endpoints of line segments with C (starting form $(0, 0)$) and Matlab (starting form $(1, 1)$).

2.2 Code Bugs

The evaluation codes provided by [1] also have several bugs. Two clear evidences for revealing these bugs are provided as follows.

1) Based on the numbers of true positives (TP), false positives (FP), and false negatives (FN), the *intersection over union* (IOU) and the F-score for evaluating the detection performance are defined by

$$\text{IOU} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}}, \tag{1}$$

$$\text{F-score} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}, \tag{2}$$

respectively. By (1) and (2), it can be easily proved that

$$0.5\text{F-score} \leq \text{IOU} \leq \text{F-score}. \tag{3}$$

However, the evaluation results presented in [1] frequently violate the mathematical rule (3). That is, the value of the IOU is often larger than the value of the F-score when Cho et al's evaluation codes are used. This is the first evidence of the code bugs in [1].

2) When evaluation is conducted on a line segment set, the *order* of line segments within the set should be immaterial. That is, the evaluation result should be invariant with respect to the order of line segments. However, the evaluation by [1] has violated such a rule too. This is the second evidence indicating the existence of code bugs. To demonstrate, the line segment sets (in total, 102 sets) produced by the Linelet [1] on the YorkUrban-LineSegment dataset are tested (this amounts to an objective performance evaluation of the Linelet). First, these line segment sets are *directly* evaluated with the evaluation codes downloaded from [2]. Then, the line segments in each set are re-arranged in a *descending* order with respect to their line lengths, and the line segment sets are again evaluated. Finally, evaluation is also performed with an *ascending* order in terms of line length. The results are documented in Table 2. One can see that three different results are yielded undesirably.

After we fixed the code bugs, the same evaluation result can be produced, as shown in Table 3. This correction has been practiced in our paper.

3 Subjective Evaluation

Subjective evaluation was performed on all the 306 test images (102 images for each noise cases), and can be easily repeated by using our evaluation code (available online [3]). For convenience, two test images (#11 and #63) selected from the YorkUrban-LineSegment dataset, as well as their noise-corrupted versions under the low- and high-level noise cases, are used for presenting a subjective evaluation as follows. The used test images are presented in Figure 6, and their ground-truth line segment sets are shown in Figure 7, respectively. The line segment detection results of the twelve detectors under evaluation are demonstrated in Figures 8 to 13, respectively.

References

[1] N. G. Cho, A. Yuille, and S. W. Lee, “A novel linelet-based representation for line segment detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 5, pp. 1195–1208, May 2018.

[2] ——, “A novel linelet-based representation for line segment detection (code and db),” https://github.com/NamgyuCho/Linelet-code-and-YorkUrban-LineSegment-DB.

[3] B. Zhong and K.-K. Ma, “Line segments in scale space: Theory and detection (the codes, data, and supplemental materials),” https://github.com/VisualComputingAtSoochow/ScaleSpaceLSD.

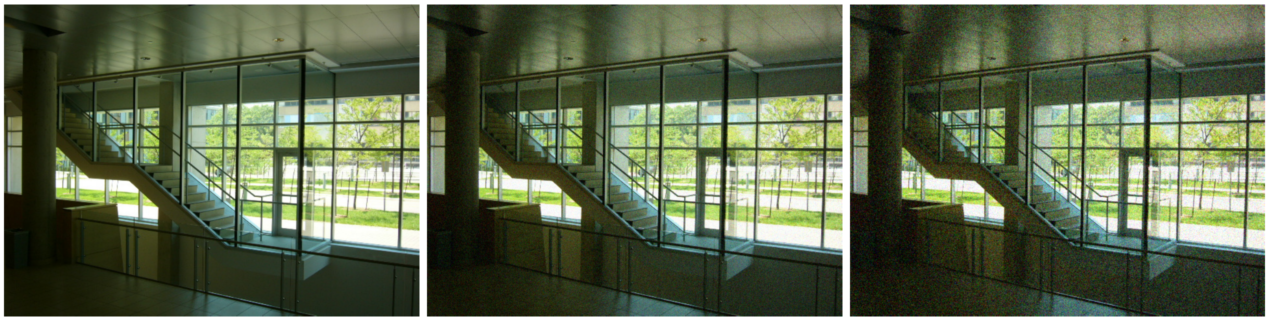
[4] R. G. von Gioi, J. Jakubowicz, J. M. Morel, and G. Randall, “LSD: A fast line segment detector with a false detection control,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, Apr. 2010.

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

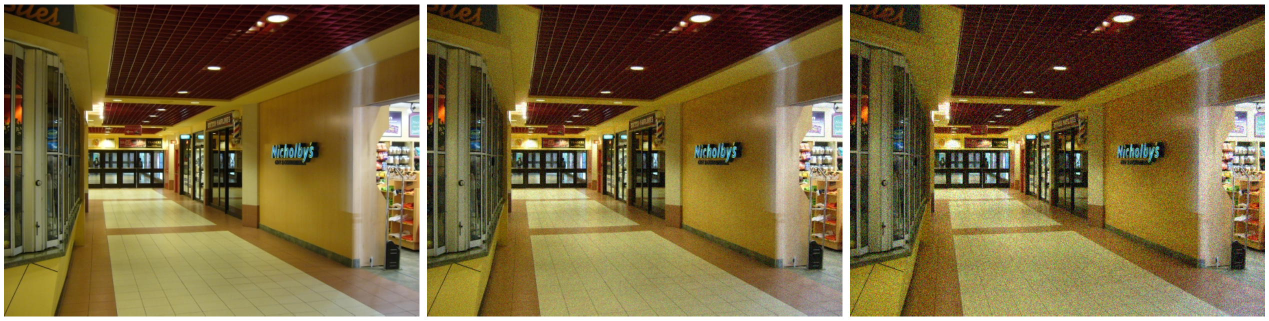
[6] V. Pătrăucean, P. Gurdjos, and R. G. von Gioi, “Joint *a Contrario* ellipse and line detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 788–802, Apr. 2017.

[7] E. J. Almazàn, 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*, Jul. 2017, pp. 5854–5862.

[8] Y. Zhang, D. Wei, and Y. Li, “AG3line: Active grouping and geometry-gradient combined validation for fast line segment extraction,” *Pattern Recognition*, vol. 113, p. 107834, 2021.

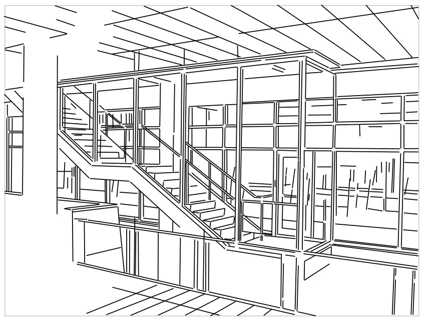


(a) Image #11.

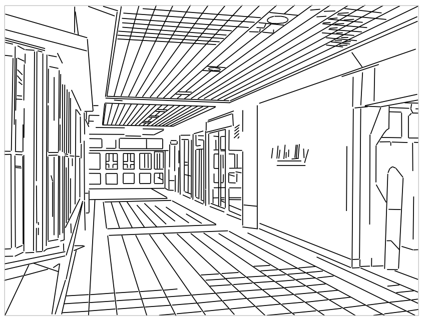


(b) Image #63.

Fig. 6. Test images (#11 and #63) taken from the YorkUrban-LineSegment dataset [1] in the noise-free case (first column), the low-level noise case (middle column), the high-level noise case (last column).

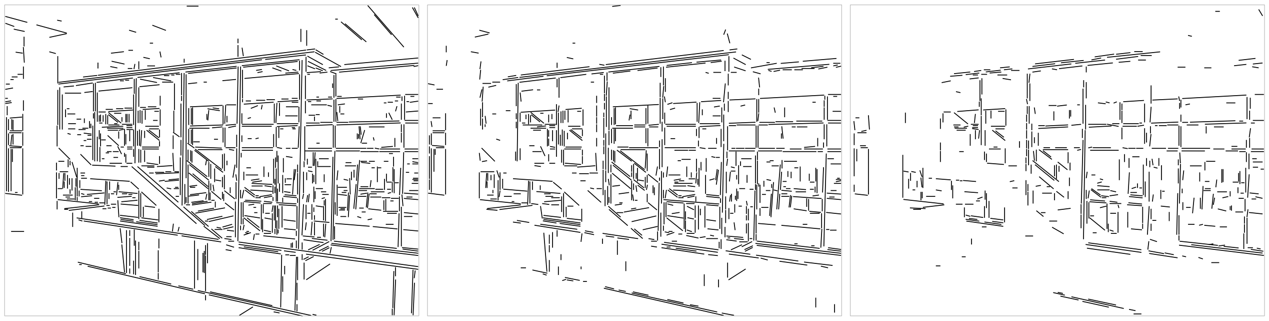


(a)

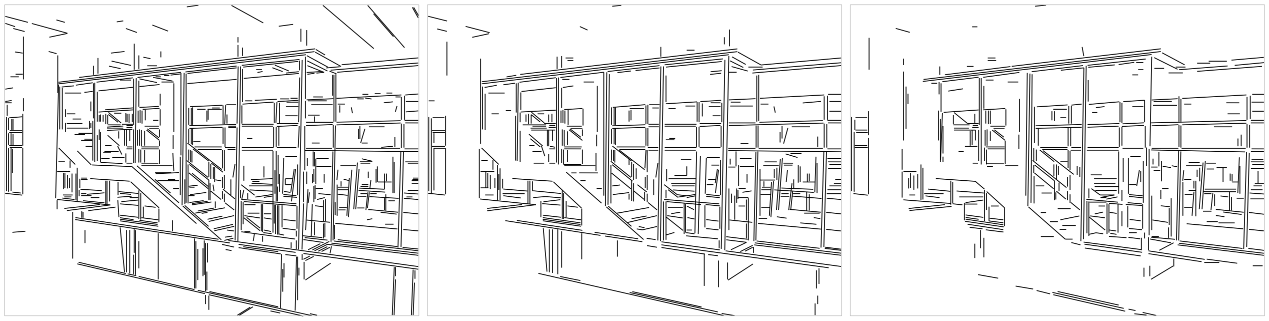


(b)

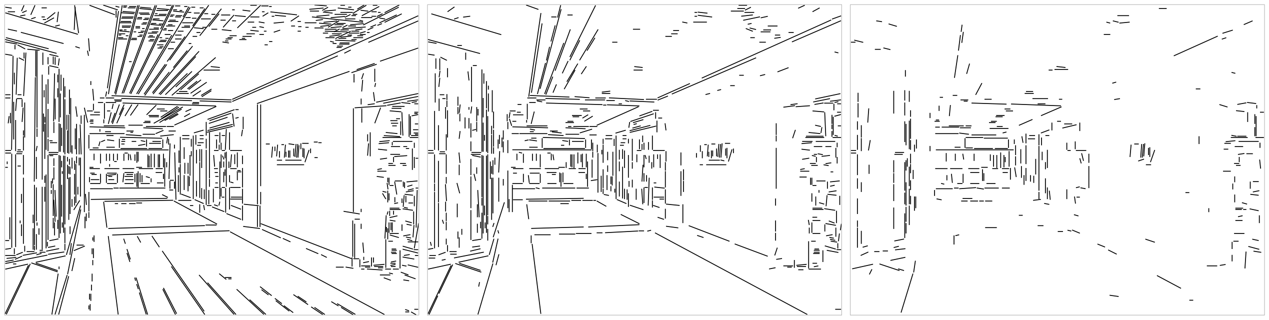
Fig. 7. The ground-truth line segment sets of the test images: (a) The ground-truth line segments of image #11; (b) The ground-truth line segments of image #63.



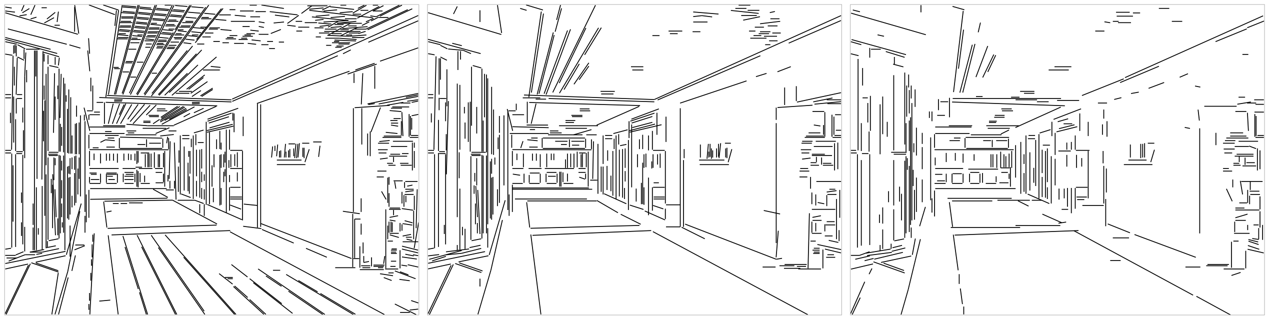
(a) Performance of the Linelet [1] on image #11 in different noise cases. F-score = 0.5282, 0.3604 and 0.1574, respectively.



(b) Performance of the scale-space Linelet on image #11 in different noise cases. F-score = 0.6275, 0.5471 and 0.4489, respectively.

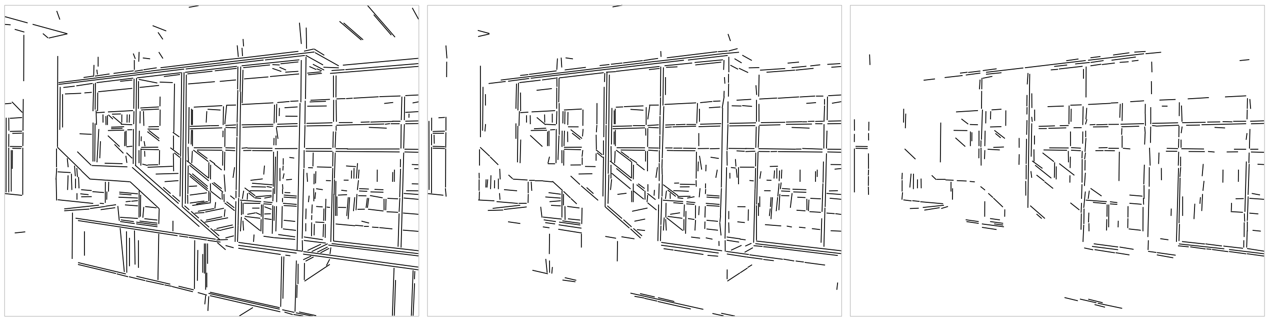


(c) Performance of the Linelet [1] on image #63 in different noise cases. F-score = 0.3885, 0.2520 and 0.0866, respectively.

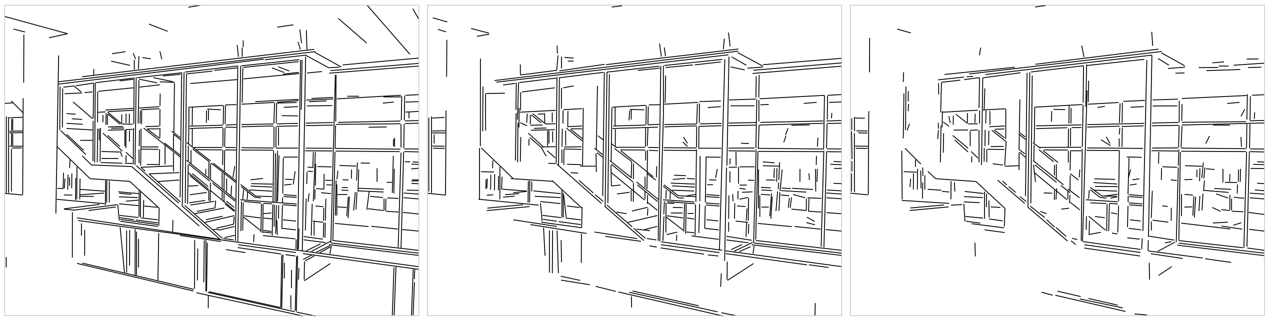


(d) Performance of the scale-space Linelet on image #63 in different noise cases. F-score = 0.4457, 0.3911 and 0.2625, respectively.

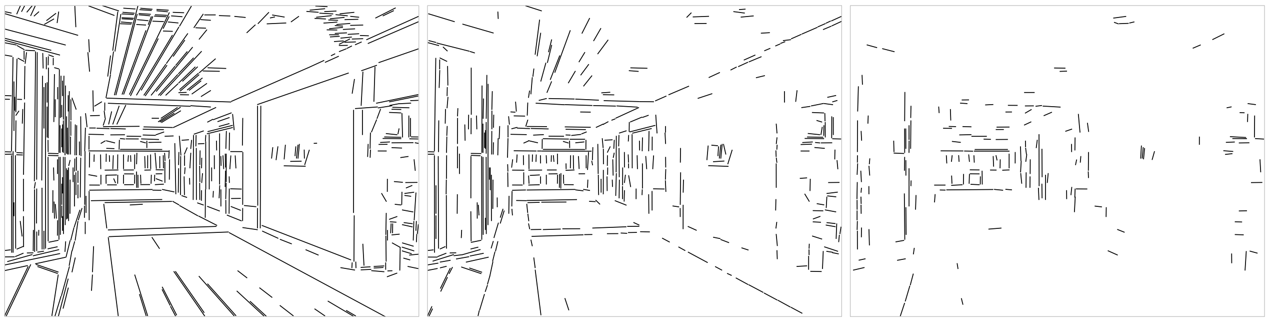
Fig. 8. Performance comparison of the Linelet [1] and the scale-space Linelet in the noise-free (first column), the low-level noise (middle column), and the high-level noise (last column) cases.



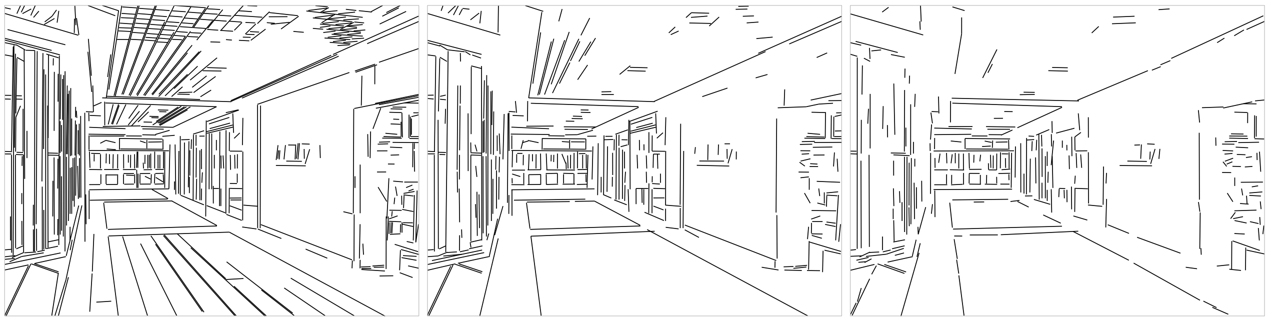
(a) Performance of the LSD [4] on image #11 in different noise cases. F-score = 0.5660, 0.3633 and 0.1730, respectively.



(b) Performance of the scale-space LSD on image #11 in different noise cases. F-score = 0.6619, 0.5262 and 0.3758, respectively.

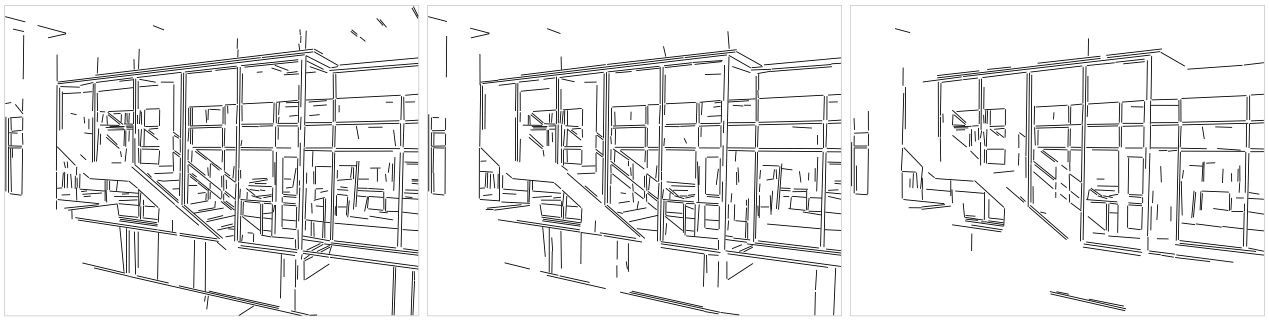


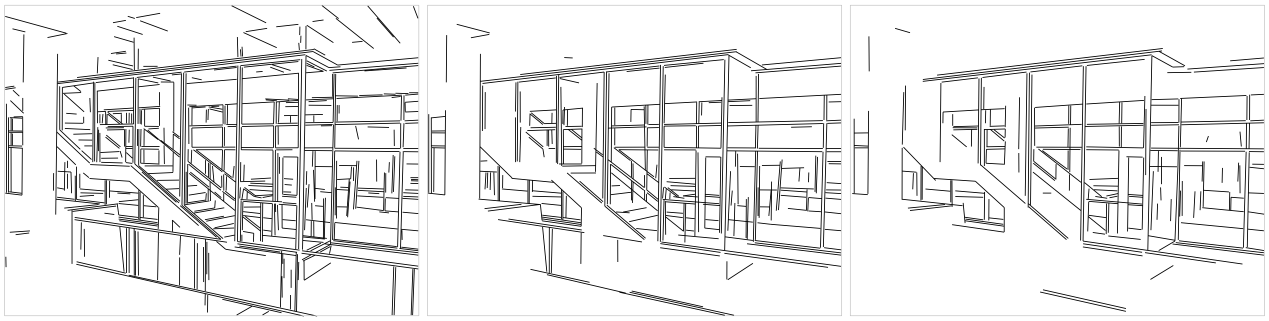
(c) Performance of the LSD [4] on image #63 in different noise cases. F-score = 0.3547, 0.1878 and 0.0614, respectively.

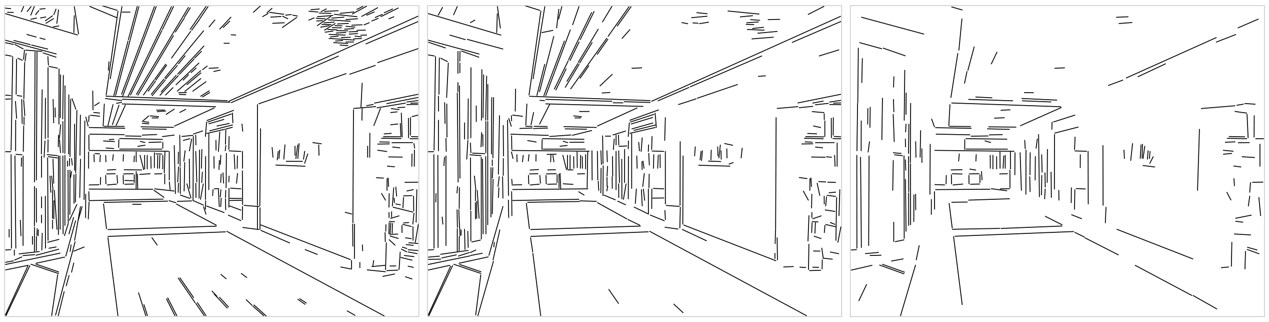


(d) Performance of the scale-space LSD on image #63 in different noise cases. F-score = 0.4669, 0.3266 and 0.2195, respectively.

Fig. 9. Performance comparison of the LSD [4] and the scale-space LSD in the noise-free (first column), the low-level noise (middle column), and the high-level noise (last column) cases.


(a) Performance of the EDLines [5] on image #11 in different noise cases. F-score = 0.5868, 0.5030 and 0.3857, respectively.


(b) Performance of the scale-space EDLines on image #11 in different noise cases. F-score = 0.6846, 0.5807 and 0.4549, respectively.


(c) Performance of the EDLines [5] on image #63 in different noise cases. F-score = 0.4043, 0.3776 and 0.2291, respectively.

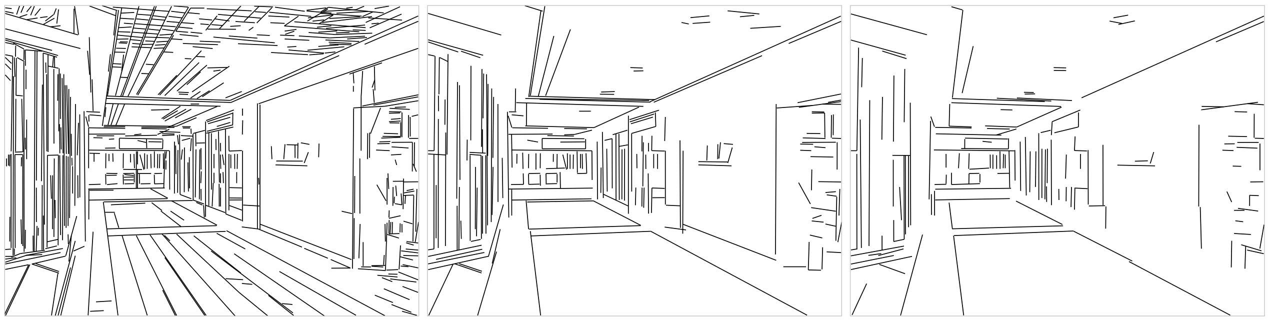
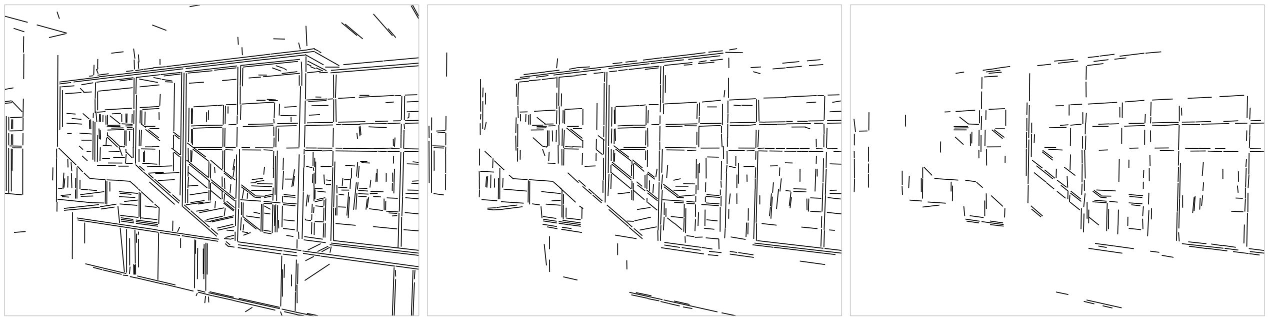
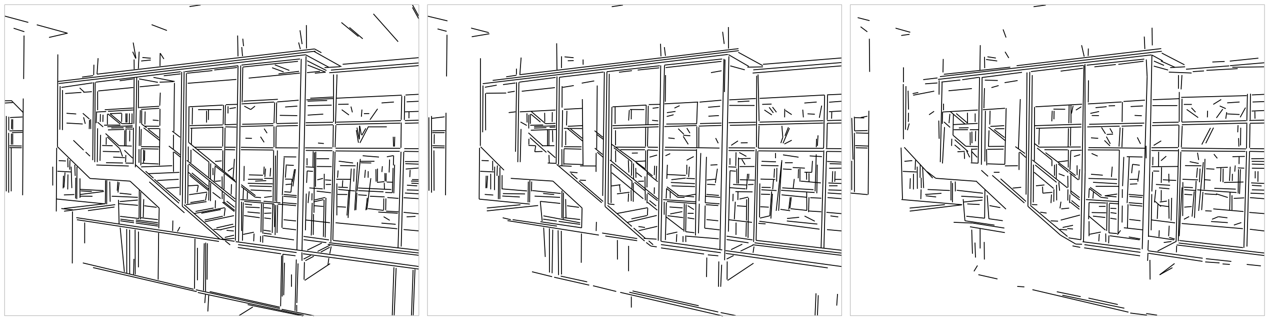

(d) Performance of the scale-space EDLines on image #63 in different noise cases. F-score = 0.4843, 0.3930 and 0.2912, respectively.

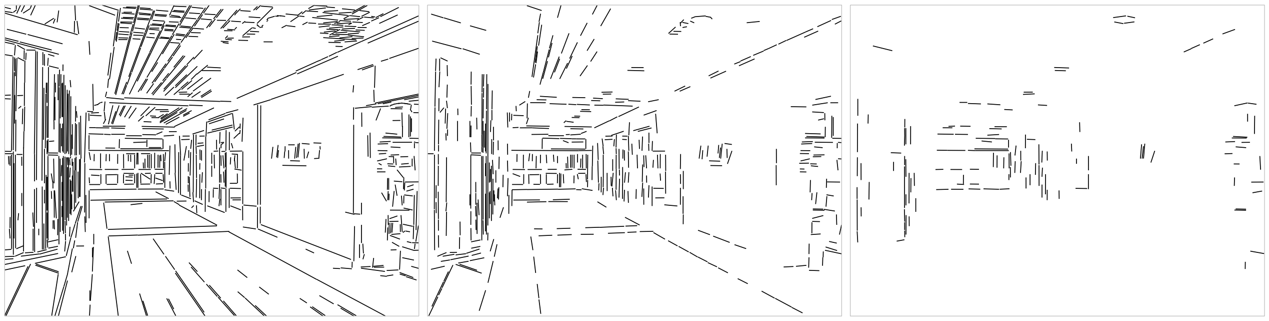
Fig. 10. Performance comparison of the EDLines [5] and the scale-space EDLines in the noise-free (first column), the low-level noise (middle column), and the high-level noise (last column) cases.



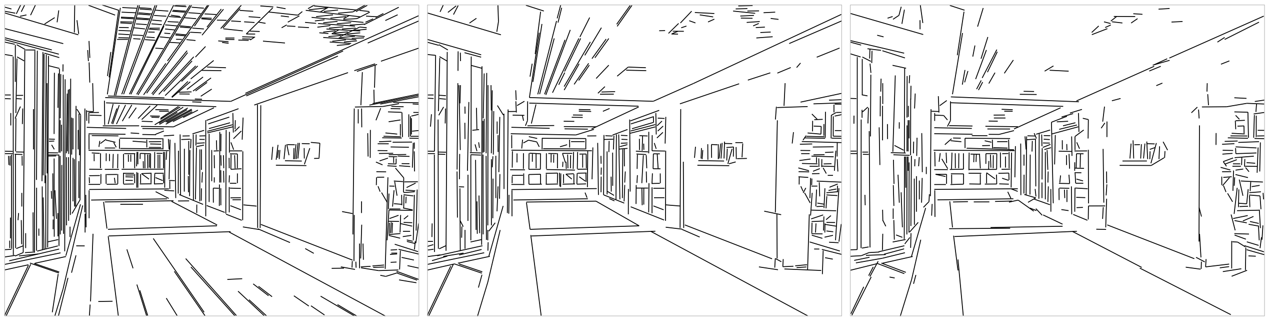
(a) Performance of the ELSDc [6] on image #11 in different noise cases. F-score = 0.5662, 0.3204 and 0.1144, respectively.



(b) Performance of the scale-space ELSDc on image #11 in different noise cases. F-score = 0.6469, 0.5158 and 0.4155, respectively.

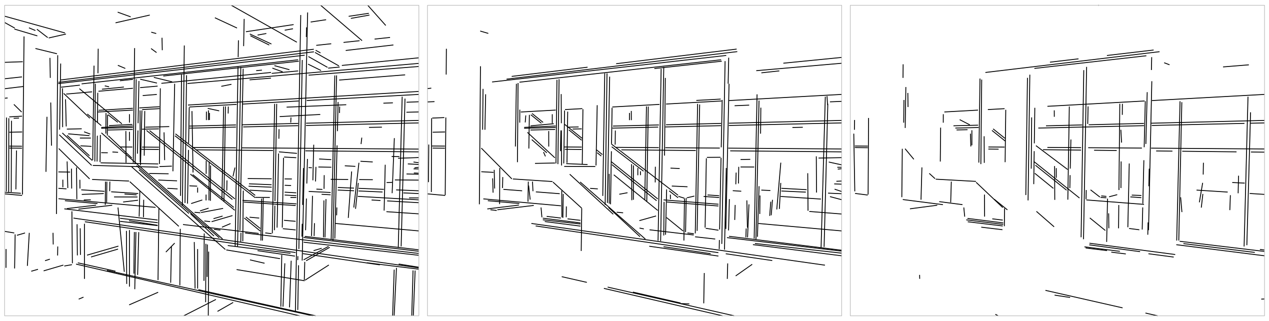


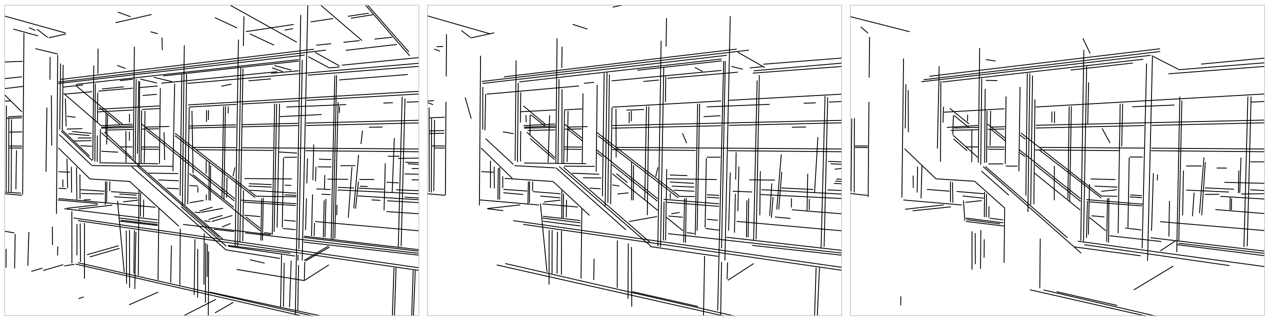
(c) Performance of the ELSDc [6] on image #63 in different noise cases. F-score = 0.3898, 0.2054 and 0.0617, respectively.

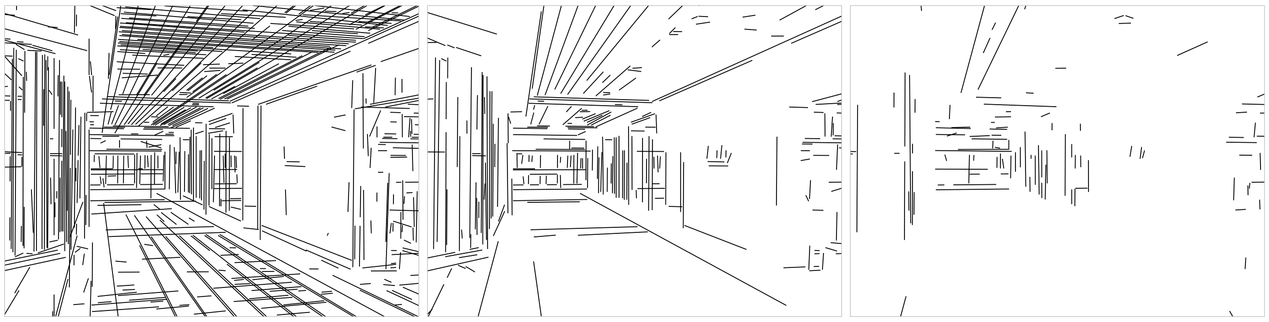


(d) Performance of the scale-space ELSDc on image #63 in different noise cases. F-score = 0.4288, 0.3572 and 0.2553, respectively.

Fig. 11. Performance comparison of the ELSDc [6] and the scale-space ELSDc in the noise-free (first column), the low-level noise (middle column), and the high-level noise (last column) cases.


(a) Performance of the MCMLSD [7] on image #11 in different noise cases. F-score = 0.6576, 0.5006 and 0.2891, respectively.


(b) Performance of the scale-space MCMLSD on image #11 in different noise cases. F-score = 0.6691, 0.5995 and 0.5038, respectively.


(c) Performance of the MCMLSD [7] on image #63 in different noise cases. F-score = 0.4788, 0.3703 and 0.1243, respectively.

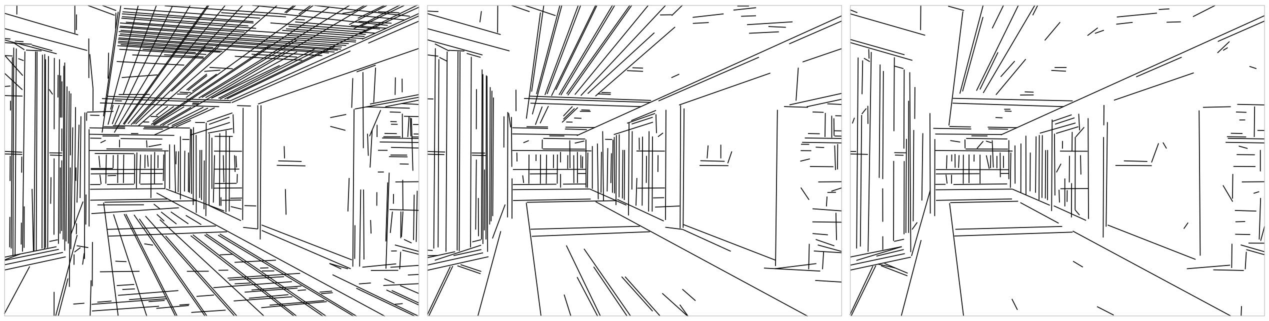
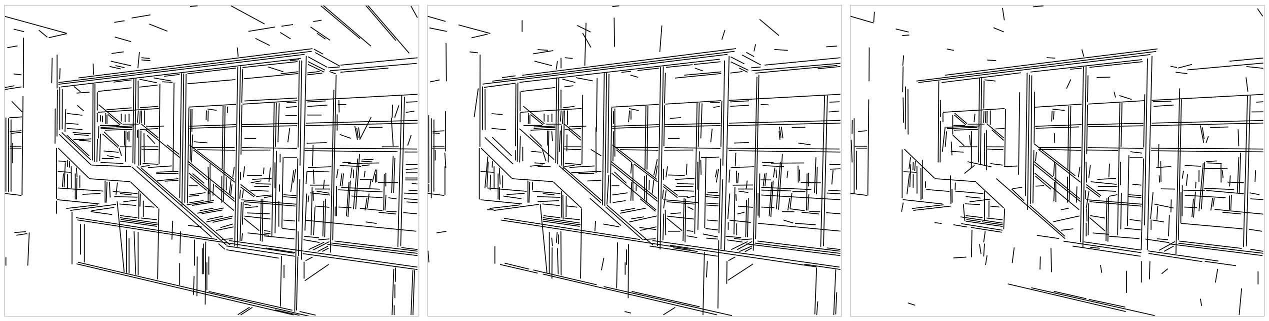
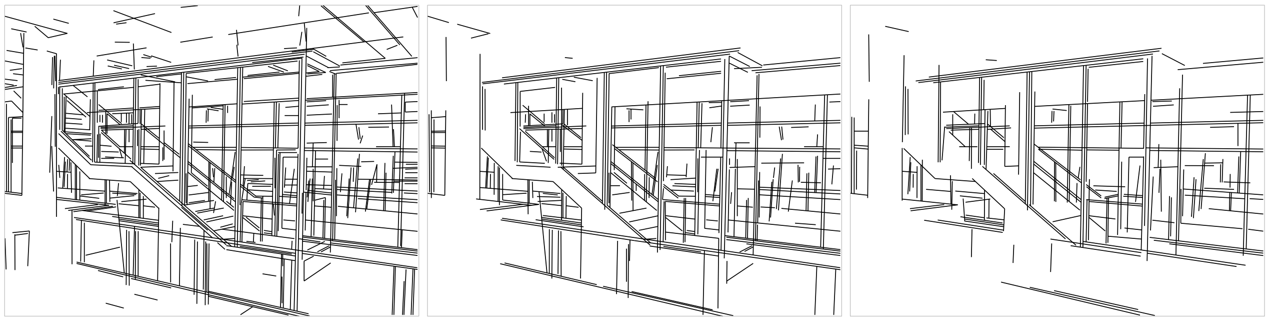

(d) Performance of the scale-space MCMLSD on image #63 in different noise cases. F-score = 0.4567, 0.4196 and 0.2844, respectively.

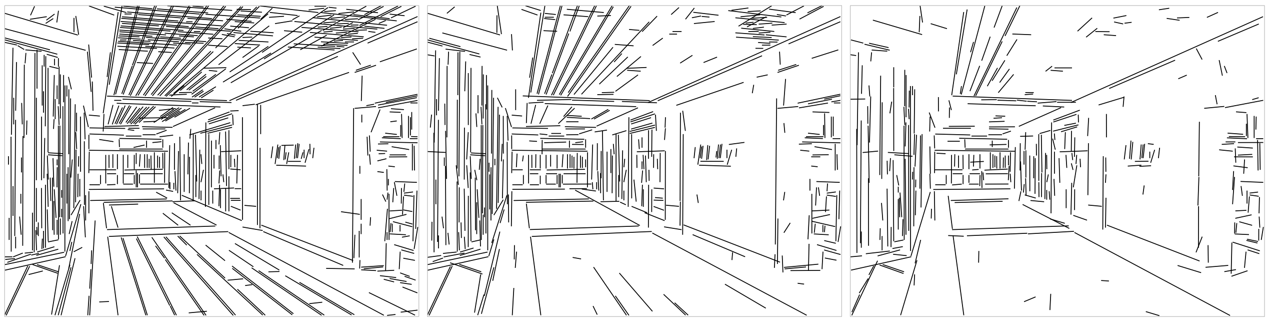
Fig. 12. Performance comparison of the MCMLSD [7] and the scale-space MCMLSD in the noise-free (first column), the low-level noise (middle column), and the high-level noise (last column) cases.



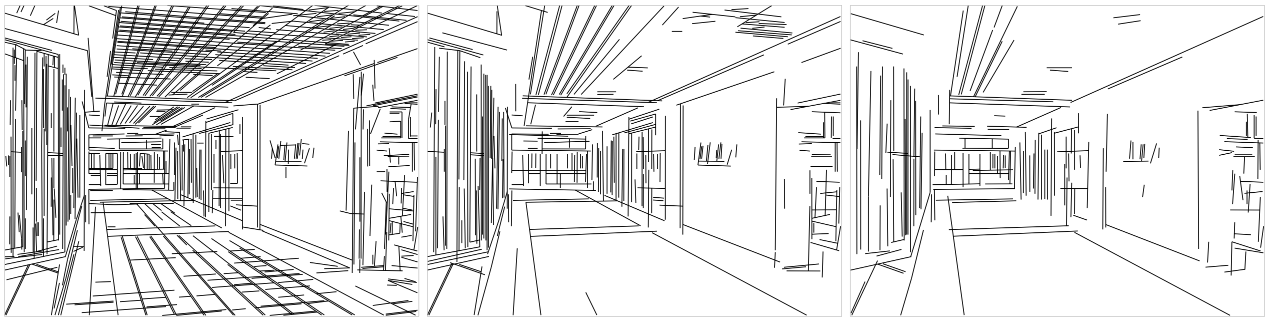
(a) Performance of the AG3line [8] on image #11 in different noise cases. F-score = 0.6395, 0.5929 and 0.4816, respectively.



(b) Performance of the scale-space AG3line on image #11 in different noise cases. F-score = 0.6828, 0.6458 and 0.5379, respectively.



(c) Performance of the AG3line [8] on image #63 in different noise cases. F-score = 0.4350, 0.4269 and 0.3204, respectively.



(d) Performance of the scale-space AG3line on image #63 in different noise cases. F-score = 0.5308, 0.4793 and 0.3827, respectively.

Fig. 13. Performance comparison of the AG3line [8] and the scale-space AG3line in the noise-free (first column), the low-level noise (middle column), and the high-level noise (last column) cases.