Supplemental Materials

Semantic Line Combination Detector

Jinwon Ko
Korea University
jwko@mcl.korea.ac.kr

Dongkwon Jin
Korea University
dongkwonjin@mcl.korea.ac.kr

Chang-Su Kim
Korea University
changsukim@korea.ac.kr

A. Implementation Details

A.1. Line detector

We design a line detector by modifying S-Net in [1] to select K reliable lines from a set of line candidates. Figure 1 shows the structure of the modified line detector performing three steps: encoding, classification/regression, and NMS.

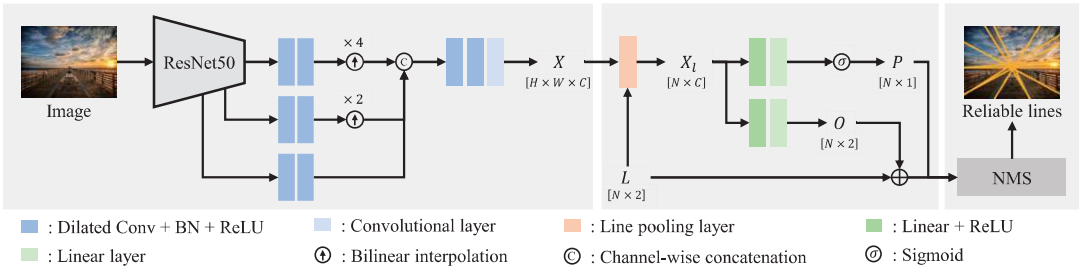


Figure 1. The architecture of the modified line detector.

Encoding: Given an image, the encoder extracts a convolutional feature map $X \in \mathbb{R}^{H \times W \times C}$. Specifically, we first extract multi-scale feature maps (the three coarsest feature maps) using ResNet50 [2] as the backbone. We then match each channel dimension to C by applying a pair of 3×3 dilated convolutional layers with batch normalization and ReLU activation. We also match the resolutions of the two coarser maps to the finest one using bilinear interpolation and concatenate them. Finally, We obtain a combined feature map X by applying two 3×3 dilated convolutional layers and one 3×3 convolutional layer.

Classification and regression: A line candidate, which is an end-to-end straight line in an image, can be parameterized by polar coordinates in the Hough space [3]. Let $l = (\rho, \theta)$ denote a line, where ρ is the distance from the center of the image and θ is its angle from the x-axis. Then, we generate an initial set of N line candidates, $L = [l_1, l_2, \ldots, l_N] \in \mathbb{R}^{N \times 2}$, by quantizing ρ and θ uniformly. For the N line candidates, we predict a probability vector $P \in \mathbb{R}^{N \times 1}$ and an offset matrix $O \in \mathbb{R}^{N \times 2}$. To this end, we extract a line feature matrix $X_l \in \mathbb{R}^{N \times C}$ of L by employing a line pooling layer [4]. Then, we feed X_l into classification and regression layers to predict P and O, respectively. More specifically, $P = \sigma(f_1(G))$ and $O = f_2(G)$, where f_1 and f_2 consist of two fully-connected layers, respectively, and $\sigma(\cdot)$ is the sigmoid function. Then, we update the line candidates by $\hat{L} = L + O$.

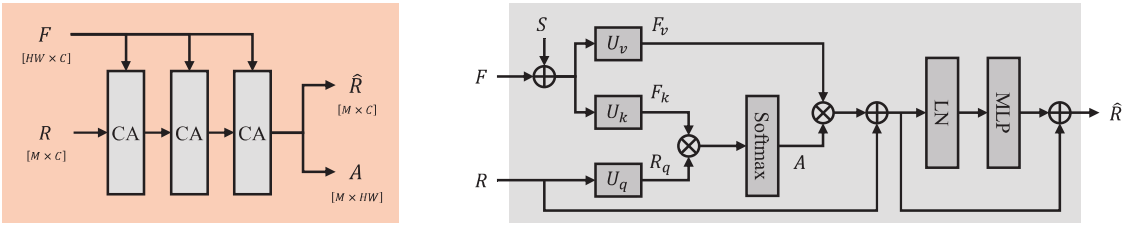
NMS: We select K reliable lines from the updated set $\hat{L}$ based on the line probability matrix P. Specifically, we select the line with the highest probability and then remove the overlapping lines, whose L_2-distances in polar coordinates from the selected one are lower than a threshold. We iterate this process K times to determine K reliable lines.

Generating line combinations: From the K reliable lines, we generate all 2^K line combinations. This process is done in parallel. Specifically, we compose a combination matrix $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{2^K}] \in \mathbb{R}^{K \times 2^K}$. Each column $\mathbf{c}_i \in \mathbb{R}^K$ is a boolean vector, where the kth element $\mathbf{c}_i^k$ is set to 1 if the kth reliable line belongs to the ith line combination, and 0 otherwise.

A.2. SLCD

We develop the semantic line combination detector (SLCD) to find the best line combination. The structure of SLCD is shown in Figure 3 in the main paper. It performs four steps: encoding, semantic feature grouping, compositional feature extraction, and score regression. Here, we describe the semantic feature grouping and score regression steps in detail.

Semantic feature grouping: Figure 2 (a) and (b) show the semantic feature grouping module and its cross-attention (CA) block, respectively. In Figure 2 (a), the module updates the region query matrix R three times using CA blocks. Each CA block takes F and R as input and yields updated region query matrix $\hat{R}$. In the last block, it additionally outputs the attention matrix A. In Figure 2 (b), LN and MLP denote layer normalization and multi-layer perceptron, respectively. The MLP consists of two fully connected layers with ReLU activation. Also, the projection matrices $U_q, U_k, U_v \in \mathbb{R}^{C\times C}$ are implemented by a fully connected layer.



(a) Semantic feature grouping module (b) Cross-attention block

Figure 2. The architectures of the semantic feature grouping module and its cross-attention block.

Score regression: We assess each line combination using a regressor. The regressor takes the compositional feature map $Z \in \mathbb{R}^{HW\times 6C}$ as input and predicts a composition score s. More specifically, we first obtain a reduced feature map by $Z' = f(\psi(Z)) \in \mathbb{R}^{\frac{H}{4}\frac{W}{4}\times C}$, where ψ is the bilinear interpolation to reduce the resolution by a factor of $\frac{1}{4}$. Also, f consists of 2D convolution layers to reduce the channel dimension to C. We then estimate the score by $s = g(\bar{Z}')$, where $\bar{Z}'$ is a flattened Z', and g consists of a series of fully connected layers with ReLU activation.

A.3. Training and hyper-parameters

Let us describe the training process of the modified line detector in detail.

Line detector: We generate a ground-truth (GT) line probability vector $\bar{P} \in \mathbb{R}^{N\times 1}$ and a GT offset matrix $\bar{O} \in \mathbb{R}^{N\times 2}$. Specifically, we set $\bar{P}_l = 1$ if the L_2-distance in polar coordinates between a line candidate and the best matching GT line is lower than a threshold, and $\bar{P}_l = 0$ otherwise. Also, we obtain $\bar{O}_l$ by computing the offset vector between a line candidate l and its matching GT line. To train the modified line detector, we minimize the loss

$$\mathcal{L} = \lambda_1 \ell_{\text{cls}}(P, \bar{P}) + \lambda_2 \ell_{\text{reg}}(O, \bar{O}) \tag{1}$$

where ℓ_{cls} is the cross-entropy loss, and ℓ_{reg} is the smooth L_1 loss. We set λ_1 and λ_2 to 1 and 5, respectively. We use the AdamW optimizer [5] with an initial learning rate of 10^{-4} and halve it after every 80,000 iterations five times. We use a batch size of eight. Also, we resize training images to 480×480 and augment them via random horizontal flipping and random rotation between -5° and 5°.

Hyper-parameters: Table 1 lists the hyper-parameters in the proposed algorithm. We will make the source codes publicly available.

Table 1. Hyper-parameter settings.

Line detector	Description	SLCD	Description
$H = 60$	Feature height	$H = 60$	Feature height
$W = 60$	Feature width	$W = 60$	Feature width
$C = 96$	# of feature channels	$C = 96$	# of feature channels
$N = 1024$	# of line candidates	$M = 8$	# of region queries
$K = 8$	# of reliable lines	$\tau = 1$	scaling factor

B. CDL Dataset

B.1. Data preparation

Semantic lines represent the layout and composition of an image. Despite the close relationship between semantic lines and photographic composition, existing semantic line datasets do not consider image composition. To assess semantic line detection results for various types of composition, we construct a compositionally diverse semantic line dataset, called CDL, in which the photos are categorized into seven composition classes: *Horizontal*, *Vertical*, *Diagonal*, *Triangle*, *Symmetric*, *Low*, and *Front*. In *Low* and *Front*, humans and animals are essential parts of the image composition. In the other classes, most images are outdoor ones, as in the other datasets. CDL contains 7,100 scenes split into 6,390 training and 710 test images. Figure 3 shows example images and annotations in the proposed CDL.

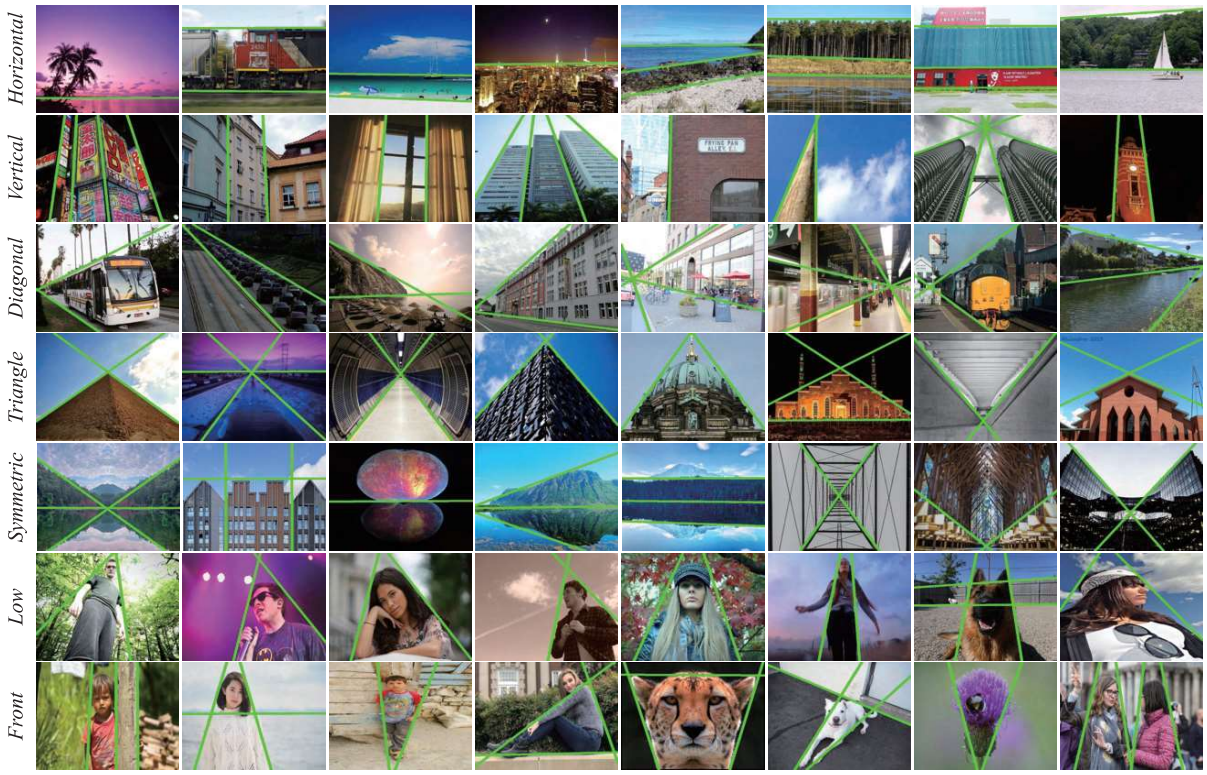
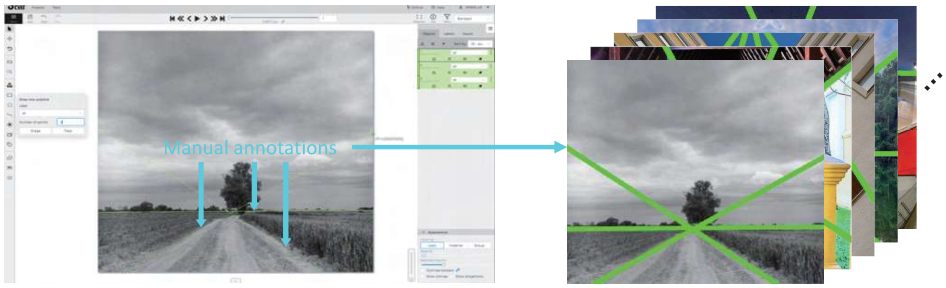


Figure 3. Example images and annotations in the proposed CDL dataset.

B.2. Manual annotation

For each image, we annotated the start and end points of each semantic line. We employed an annotation tool, CVAT [6], as shown in Figure 4. We adjusted the coordinates of each line so that the lines represented the image composition optimally and harmoniously. Eight people inspected each annotated image, and unsatisfactory annotations were identified and re-annotated. The inspection was repeated until a consensus was made. The manual annotation process took more than 200 man-hours.



CVAT environment Annotation results

Figure 4. Manual annotation process of semantic lines using CVAT. Ground-truth lines are depicted in green.

C. More analysis

C.1. Performances

Table 2 reports the performances on the SEL and SEL_Hard datasets. We compare the area under curve (AUC) performances of the precision, recall, and F-measure curves, which are denoted by AUC_P, AUC_R, and AUC_F, respectively [4]. Also, Table 3 compares the EA-scores [7] on the NKL and proposed CDL datasets. Note that, unlike HIoU [1], AUC performances and EA-scores do not consider the overall harmony of detected lines. They only consider the positional accuracy of each detected line. We see that the proposed SLCD provides better results on most datasets in terms of detecting harmonious lines while accurately identifying individual lines.

Table 2. Comparison of the AUC and HIoU on the SEL and SEL_Hard datasets.

	SEL				SEL_Hard			
	AUC_P	AUC_R	AUC_F	HIoU	AUC_P	AUC_R	AUC_F	HIoU
SLNet [4]	80.72	84.22	82.43	77.87	74.22	70.68	72.41	59.71
DHT [3]	87.74	80.25	83.83	79.62	83.55	67.98	75.09	63.39
DRM [8]	85.44	**86.87**	86.15	80.23	87.19	**77.69**	**82.17**	**68.83**
HSLD [1]	89.61	83.93	86.68	81.03	87.60	72.56	79.38	65.99
Proposed	**95.27**	80.03	**86.99**	**84.09**	**90.32**	70.14	78.96	68.15

Table 3. Comparison of the EA-scores and HIoU on the NKL and CDL datasets.

	NKL				CDL			
	Precision	Recall	F-measure	HIoU	Precision	Recall	F-measure	HIoU
SLNet [4]	74.41	74.50	74.45	65.49	63.99	74.08	68.61	57.78
DHT [3]	78.76	**83.17**	80.88	69.08	72.02	**79.49**	75.52	63.24
DRM [8]	74.27	79.28	76.67	67.42	72.80	72.13	72.46	63.96
HSLD [1]	79.60	81.14	80.36	74.29	71.74	75.86	73.73	64.98
Proposed	**84.39**	80.60	**82.44**	**76.21**	**75.69**	78.41	**77.02**	**68.85**

Table 4 shows the HIoU scores on the proposed CDL dataset according to the composition classes. We train the existing detectors on CDL using their publicly available source codes. We see that the proposed SLCD outperforms all the existing techniques in all classes without exception. Among the seven classes, *Low* and *Front* are challenging ones because the complex boundaries of objects (humans or animals) make it difficult to identify the implied semantic lines. Qualitative results are shown in Section D.3.

Table 4. Comparison of the HIoU scores (%) according to the composition classes in the CDL dataset.

	Horizontal	*Vertical*	*Diagonal*	*Triangle*	*Symmetric*	*Low*	*Front*	Total
SLNet [4]	72.67	51.32	64.52	47.04	69.18	46.87	49.25	57.78
DHT [3]	76.02	59.70	64.87	59.45	65.61	47.82	60.43	63.24
DRM [8]	75.86	61.19	67.86	51.69	69.09	49.68	63.41	63.96
HSLD [1]	76.43	60.36	69.01	58.35	69.66	54.91	60.75	64.98
Proposed	**77.55**	**66.15**	**72.87**	**67.89**	**72.03**	**57.13**	**64.84**	**68.85**

C.2. Ablations

N and K: As described in Section 3.1 in the main paper, to generate a manageable number of line combinations, we filter out redundant line candidates and obtain K reliable lines. Table 5 lists the performances according to the number N of line candidates and the number K of reliable lines on CDL: (a) Recall (EA-score) rate of the line detector according to N, and (b) HIoU of the proposed SLCD according to K. The processing times are also reported in seconds per frame (spf).

In Table 5 (a), as N increases, the recall rate gets higher. However, too large values of N make the training of the line detector challenging, yielding a lower recall rate at $N = 1444$, compared to $N = 1024$. Hence, we set $N = 1024$. In Table 5 (b), as K increases, the HIoU score gets higher. At $K = 10$, SLCD achieves the highest HIoU, but it becomes too slow. It takes 4 times longer at $K = 10$ than at $K = 8$, whereas their HIoU gap is 0.27 only. As a tradeoff between performance and speed, we set $K = 8$.

Table 5. Comparison of the performance according to the number N of line candidates and the number K of reliable lines on CDL.

$K = 8$	Line detector			
$N \triangleright$	400	900	1024	1444
Recall	88.61	90.16	91.77	91.23
spf	0.027	0.028	0.030	0.032

(a) Recall rate of the line detector according to N

$N = 1024$	SLCD		
$K \triangleright$	6	8	10
HIoU	67.96	68.85	69.12
spf	0.033	0.074	0.316

(b) HIoU of SLCD according to K.

C.3. Visualizations

Sorting results according to composition scores: Note that we determine an optimal group of semantic lines in an image, by finding the line combination with the highest composition score, estimated by the proposed SLCD. Figure 5 (b) and (c) show examples of top-4 and bottom-4 line combinations, respectively. The top-4 detection results represent image composition faithfully. In contrast, the bottom-4 detection results represent it poorly. These examples indicate that the proposed SLCD evaluates each line combination reliably based on composition analysis.

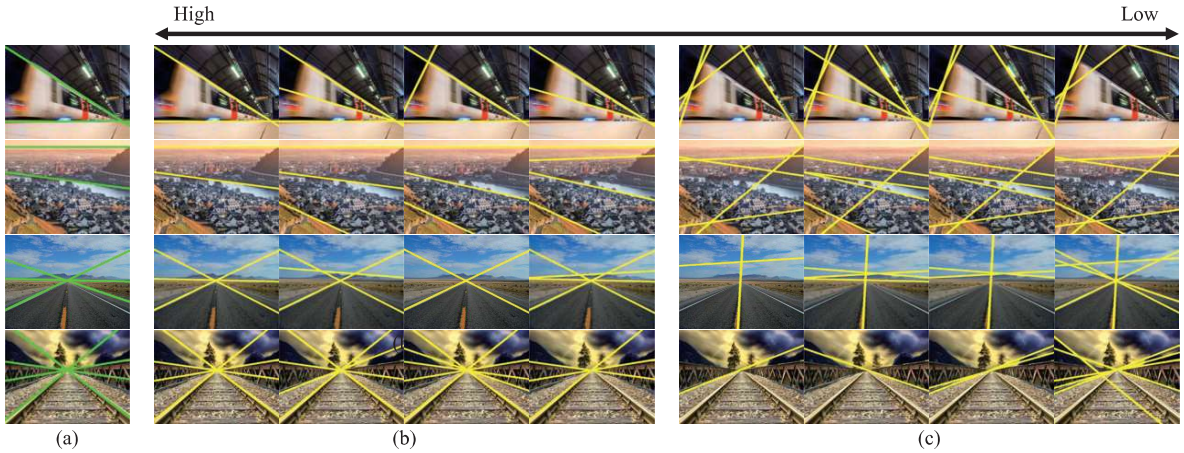


Figure 5. Sorting of line combinations based on predicted composition scores: (a) ground-truth lines, (b) top-4 line combinations, and (c) bottom-4 line combinations.

D. Qualitative Results

D.1. Comparison on SEL and SEL_Hard

Figure 6 and Figure 7 compare the proposed SLCD with conventional algorithms on SEL and SEL_Hard, respectively. SLCD provides better detection results than the conventional algorithms.

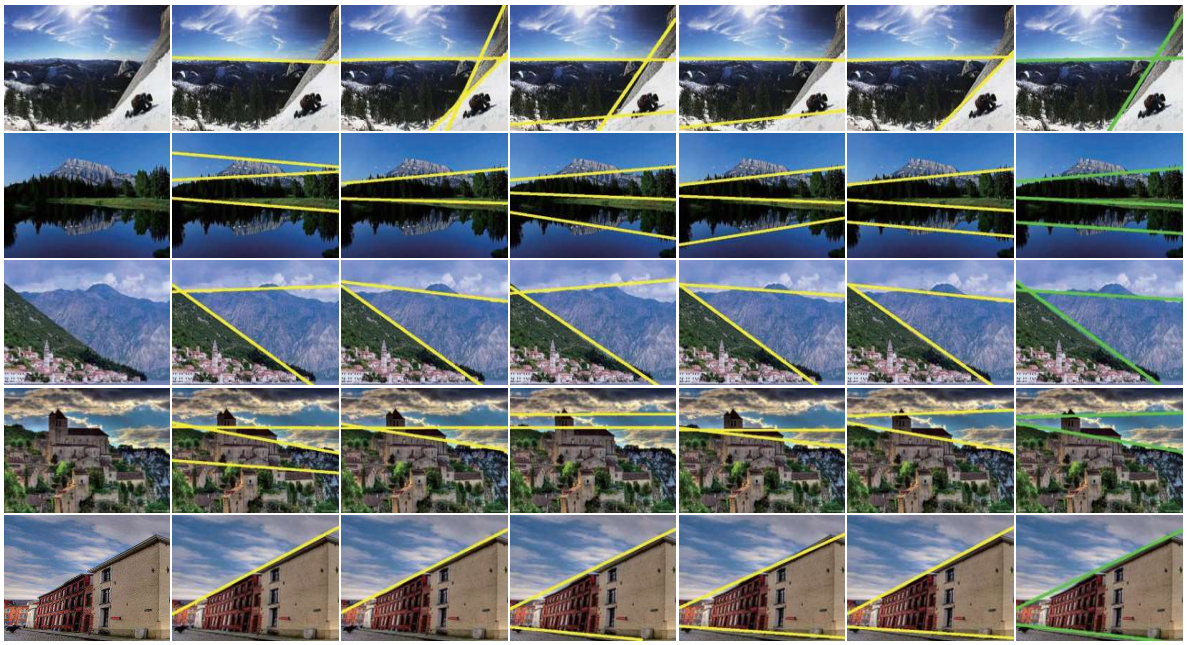


Image SLNet [4] DHT [3] DRM [8] HSLD [1] Proposed Ground-truth

Figure 6. Comparison of semantic line detection results on the SEL dataset.

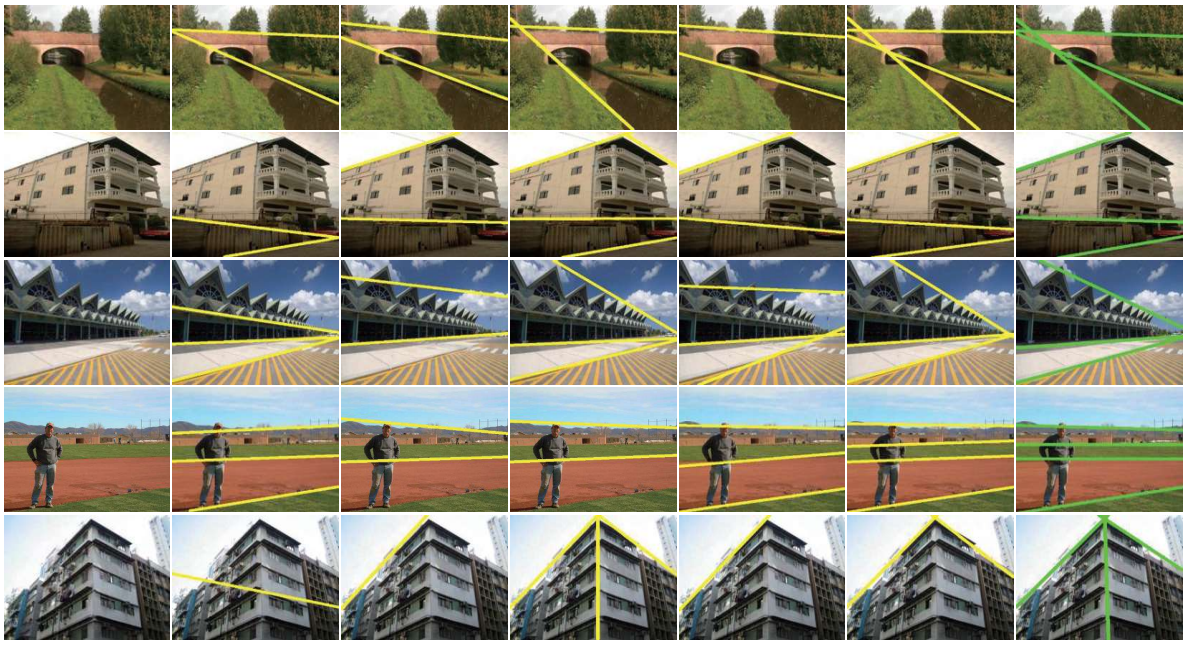


Image SLNet [4] DHT [3] DRM [8] HSLD [1] Proposed Ground-truth

Figure 7. Comparison of semantic line detection results on the SEL_Hard dataset.

D.2. Comparison on NKL

Figure 8 compares the proposed SLCD with existing line detectors on NKL.

Figure 8. Comparison of semantic line detection results on the NKL dataset.

D.3. Comparison on CDL

Figures 9∼15 compare semantic line detection results of the proposed CDL with those of the conventional algorithms on the *Horizontal*, *Vertical*, *Diagonal*, *Triangle*, *Symmetric*, *Low*, and *Front* classes in the CDL dataset, respectively. We see that the proposed algorithm provides more reliable detection results consistently.

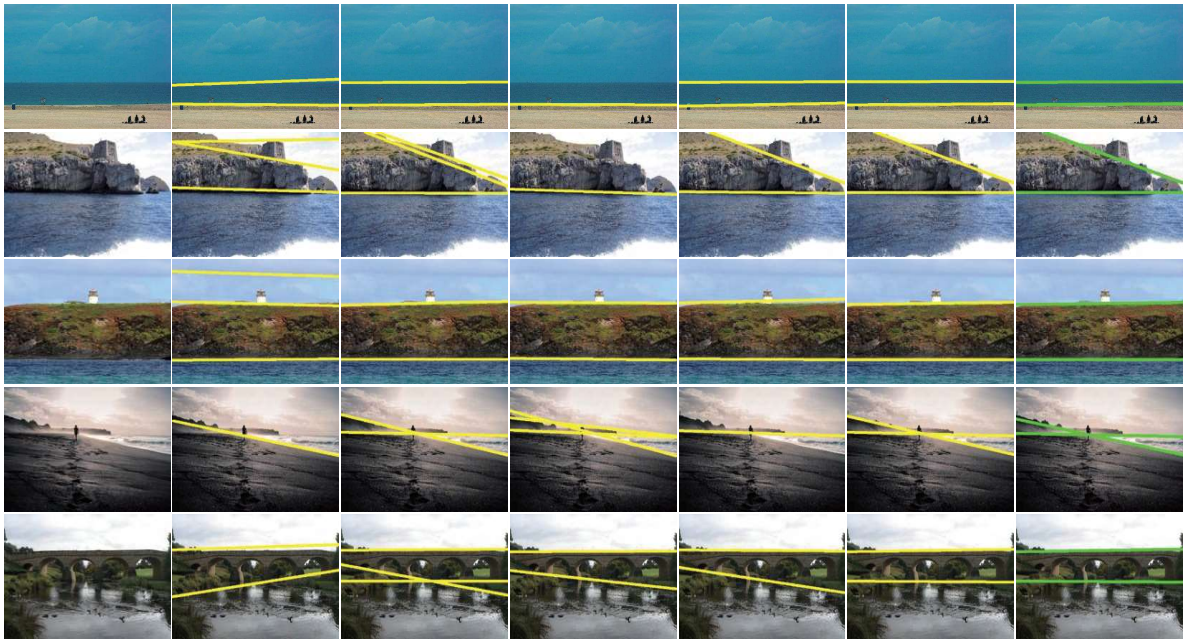


Image SLNet [4] DHT [3] DRM [8] HSLD [1] Proposed Ground-truth

Figure 9. Comparison of semantic line detection results on the *Horizontal* class in the CDL dataset.

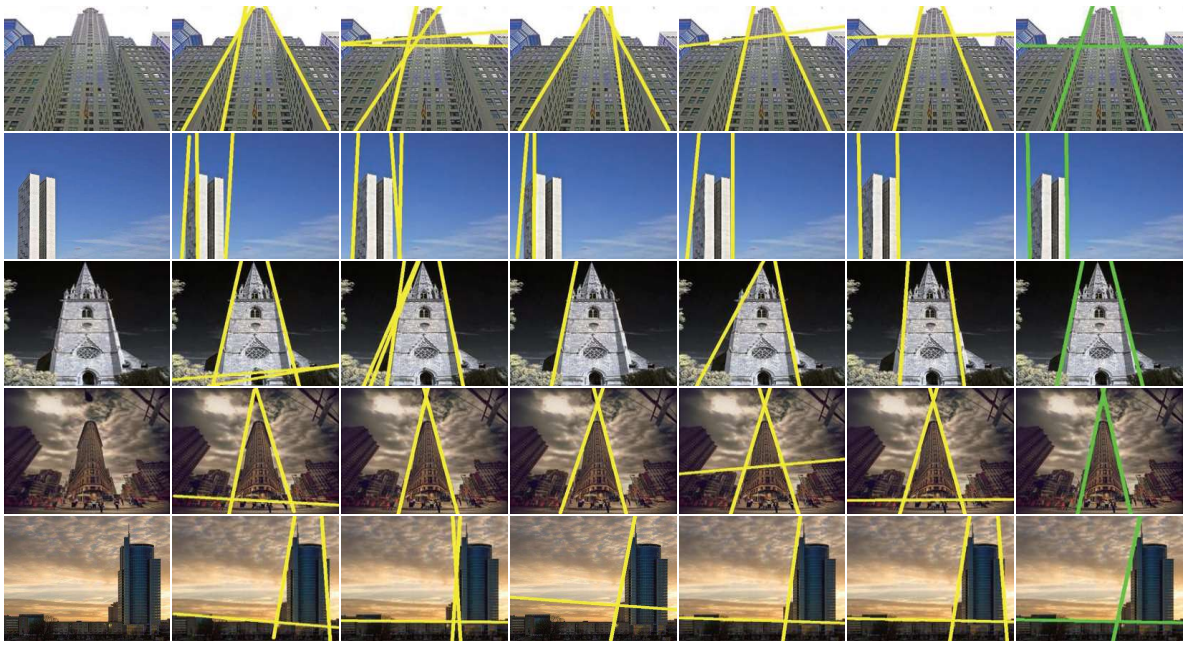


Image SLNet [4] DHT [3] DRM [8] HSLD [1] Proposed Ground-truth

Figure 10. Comparison of semantic line detection results on the *Vertical* class in the CDL dataset.

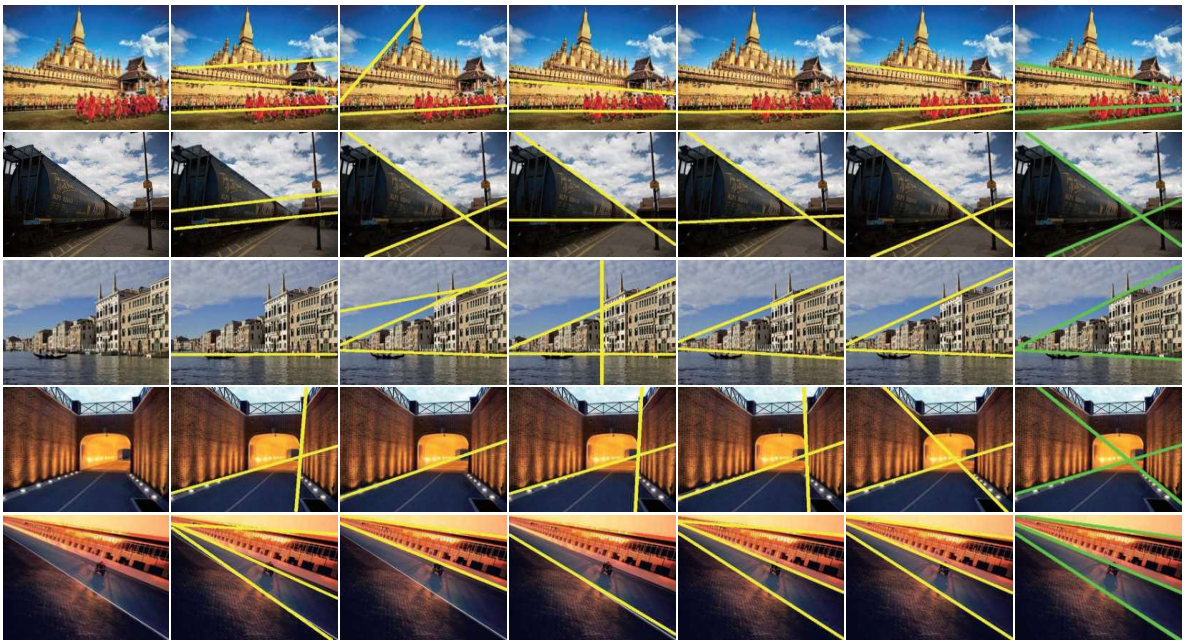


Image SLNet [4] DHT [3] DRM [8] HSLD [1] Proposed Ground-truth

Figure 11. Comparison of semantic line detection results on the *Diagonal* class in the CDL dataset.

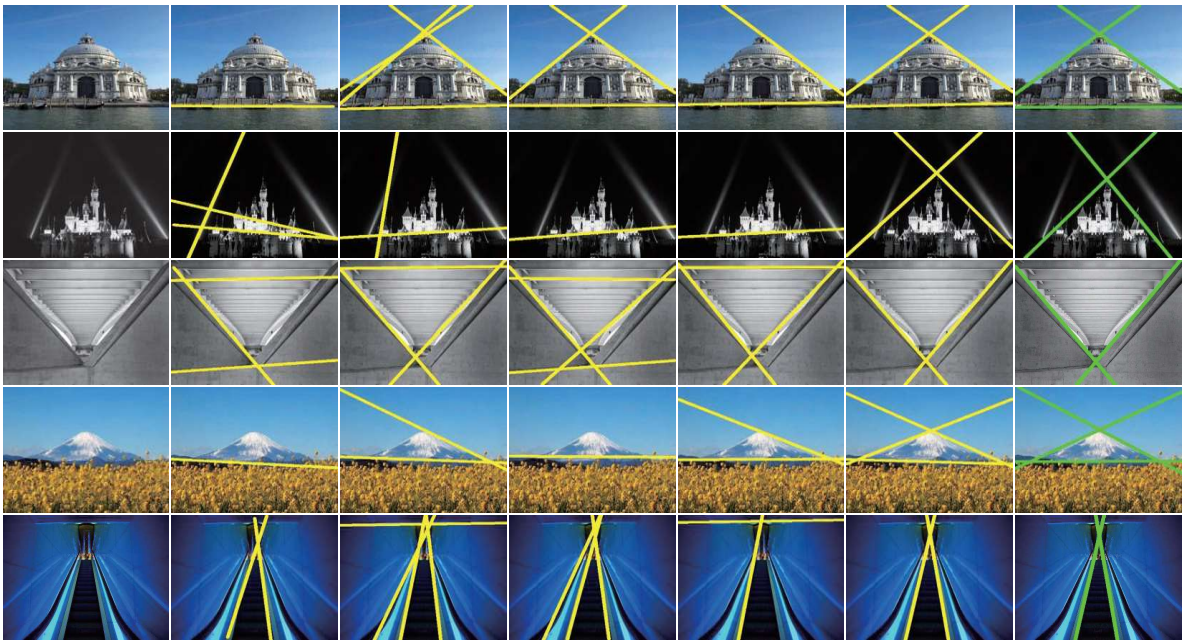


Image SLNet [4] DHT [3] DRM [8] HSLD [1] Proposed Ground-truth

Figure 12. Comparison of semantic line detection results on the *Triangle* class in the CDL dataset.

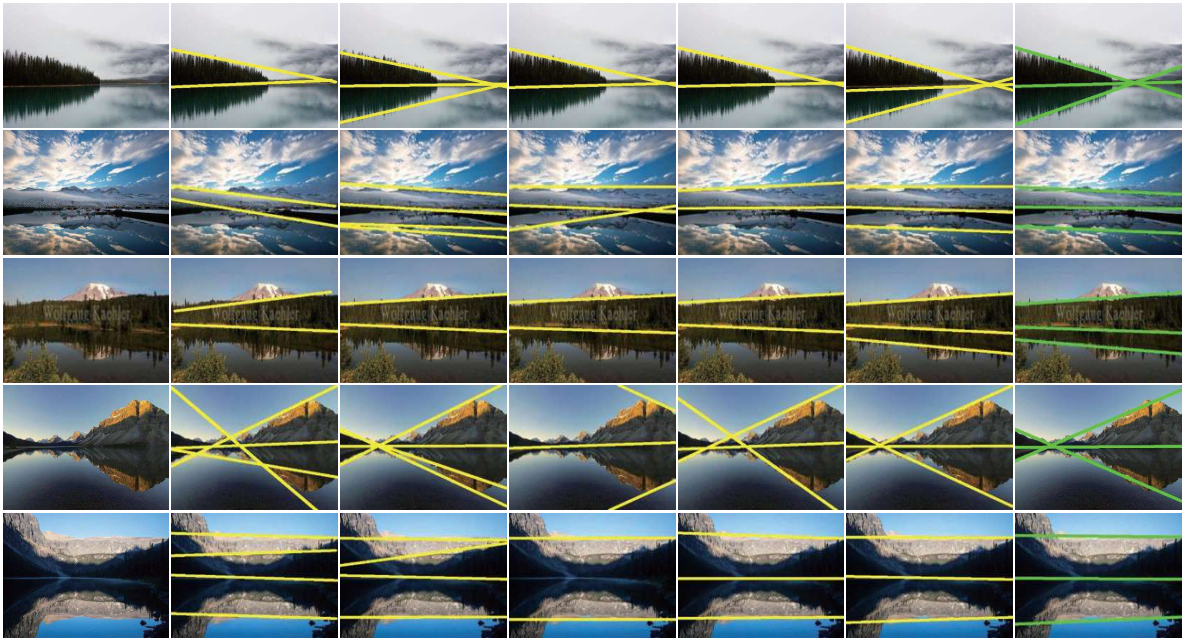


Image SLNet [4] DHT [3] DRM [8] HSLD [1] Proposed Ground-truth

Figure 13. Comparison of semantic line detection results on the *Symmetric* class in the CDL dataset.

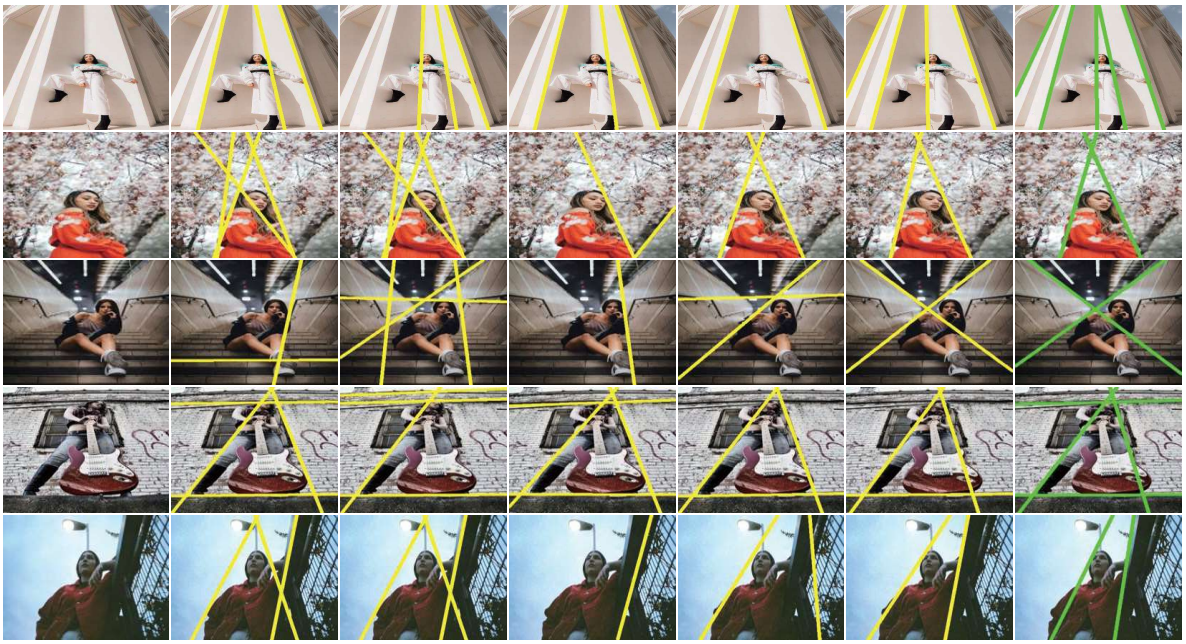


Image SLNet [4] DHT [3] DRM [8] HSLD [1] Proposed Ground-truth

Figure 14. Comparison of semantic line detection results on the *Low* class in the CDL dataset.

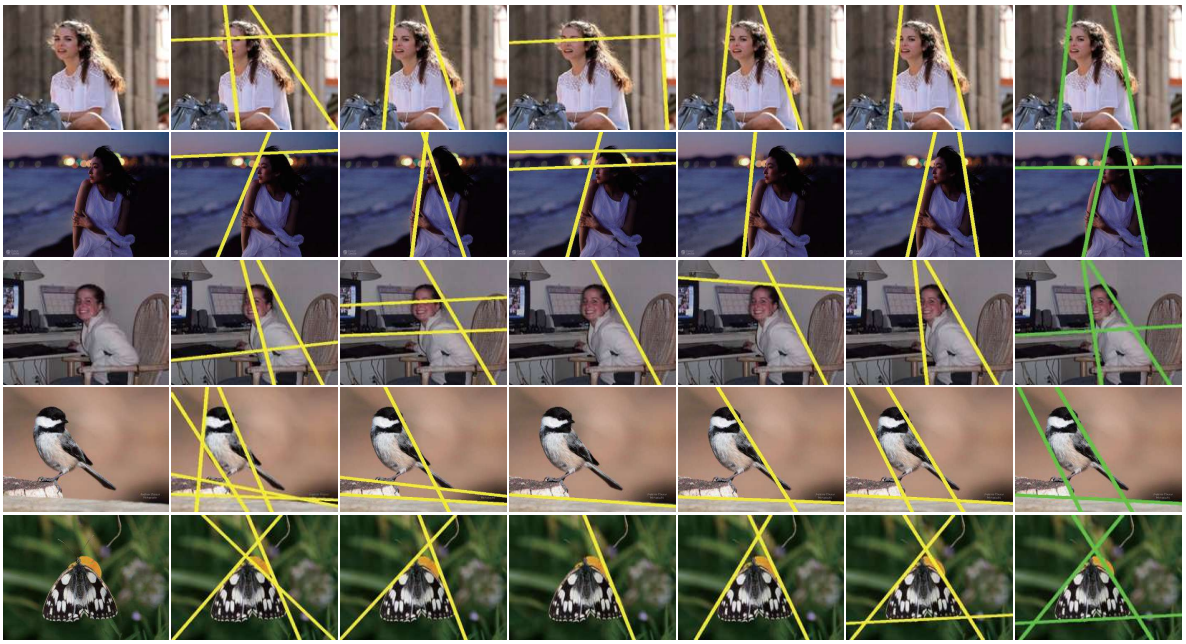


Image SLNet [4] DHT [3] DRM [8] HSLD [1] Proposed Ground-truth

Figure 15. Comparison of semantic line detection results on the *Front* class in the CDL dataset.

E. Applications

E.1. Dominant vanishing point detection

We apply the proposed SLCD to detect dominant vanishing points by identifying the vanishing lines. We declare the intersecting point of two lines as a vanishing point. We assess the proposed SLCD on the AVA landscape dataset [9] which contains 2,000 training and 275 test landscape images. Figure 16 shows more results of vanishing point detection.

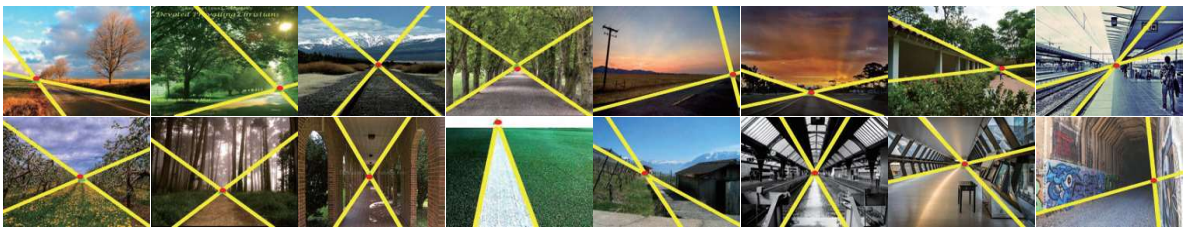


Figure 16. More results of vanishing point detection.

E.2. Reflection symmetry axis detection

We test the proposed SLCD on three datasets: ICCV [10], NYU [11], and SYM_Hard [8]. ICCV provides 100 training and 96 test images, NYU contains 176 test images, and SYM_Hard consists of 45 test images. In these datasets, each images contains a single reflection symmetry axis. We trained the model on the ICCV and tested it on all datsets. Figure 17 shows more results of vanishing point detection. In the top row, the ground-truth and predicted axes are depicted by dashed red and solid yellow lines, respectively. The membership maps are also visualized in the bottom row.

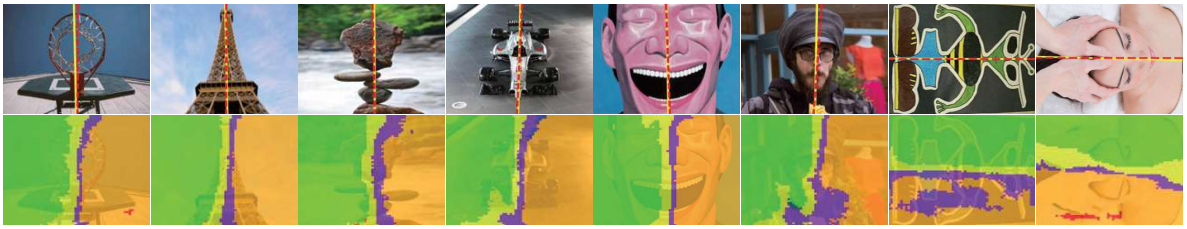
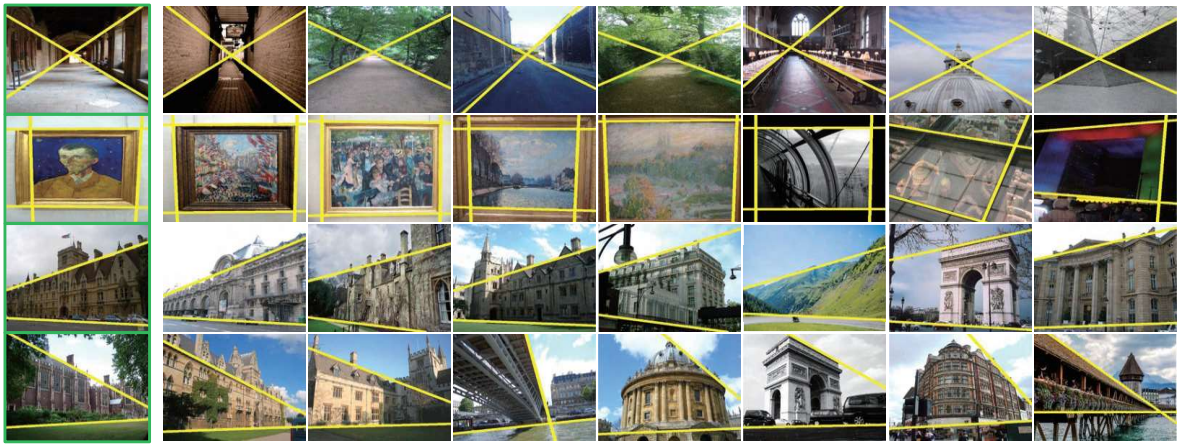


Figure 17. More results of symmetric axis detection.

E.3. Composition-based image retrieval

We test the proposed SLCD on Oxford 5k and Paris 6k dataset [12], containing photographs of landmarks and local attractions in two places. Since it contains indoor and outdoor images mainly, we use the network trained on NKL dataset. We first detect semantic lines for all the images, while storing the positional feature map P. Then, we filter out images whose the composition scores are lower than a threshold 0.75 since a low score indicates a low quality image with inharmonious composition, as shown in Figure 5. Then, for a randomly selected query image, we compute the ℓ_2-distances between the positional feature maps P of the query and the remaining images. We determine the images with the smallest distances as retrieval results. Figure 18 shows more results of composition-based retrieval.



Query image Top-7 retrieval results

Figure 18. More results of composition-based retrieval.

E.4. Road lane detection

We apply the proposed SLCD to the lane detection task, by employing the CULane dataset [13]. CULane is a road lane dataset in which lanes are annotated with 2D coordinates. To obtain ground-truth semantic lines corresponding to the lanes in an image, we declare the most overlapping line with each lane as a semantic line. Figure 19 shows some examples of original lane points and ground-truth semantic lines, depicted by red dots and green lines, respectively.

Figure 19. Examples of original lane points and ground-truth semantic lines.

We train SLCD and compare it with the second-best line detector HSLD [1]. Figure 20 shows some lane detection results on the test set. SLCD detects semantic lines more reliably in road scenes than HSLD does, even though some lines are implied due to occlusion or poor illumination. Table 6 compares the HIoU and AUC results. We see that SLCD outperforms HSLD meaningfully.

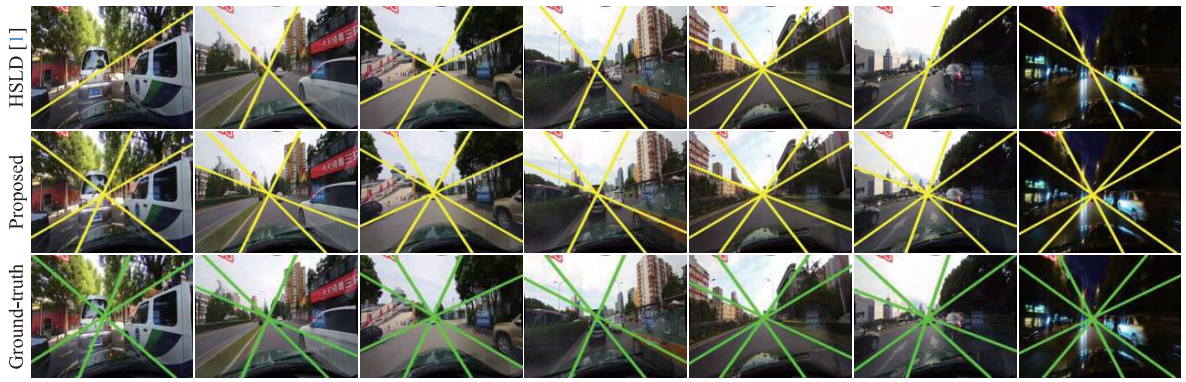


Figure 20. Comparison of semantic line detection results on the CULane dataset.

Table 6. Comparison of the HIoU and AUC results on the CULane dataset.

	AUC_P	AUC_R	AUC_F	HIoU
HSLD [1]	92.38	80.11	85.80	74.14
Proposed	92.91	84.86	88.55	76.97

References

[1] D. Jin, W. Park, S.-G. Jeong, and C.-S. Kim, "Harmonious semantic line detection via maximal weight clique selection," in *Proc. IEEE CVPR*, 2021. 1, 4, 5, 13

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE CVPR*, 2016. 1

[3] Q. Han, K. Zhao, J. Xu, and M.-M. Cheng, "Deep hough transform for semantic line detection," in *Proc. ECCV*, 2020. 1, 4, 5

[4] J.-T. Lee, H.-U. Kim, C. Lee, and C.-S. Kim, "Semantic line detection and its applications," in *Proc. IEEE ICCV*, 2017. 1, 4, 5

[5] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017. 2

[6] "CVAT." [Online]. Available: https://www.cvat.ai/. 4

[7] K. Zhao, Q. Han, C.-B. Zhang, J. Xu, and M.-M. Cheng, "Deep hough transform for semantic line detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, pp. 4793–4806, 2021. 4

[8] D. Jin, J.-T. Lee, and C.-S. Kim, "Semantic line detection using mirror attention and comparative ranking and matching," in *Proc. ECCV*, 2020. 4, 5, 12

[9] Z. Zhou, F. Farhat, and J. Z. Wang, "Detecting dominant vanishing points in natural scenes with application to composition-sensitive image retrieval," *IEEE Trans. Multimedia*, vol. 19, pp. 2651–2665, 2017. 11

[10] C. Funk, S. Lee, M. R. Oswald, S. Tsogkas, W. Shen, A. Cohen, S. Dickinson, and Y. Liu, "2017 ICCV Challenge: Detecting symmetry in the wild," in *Proc. IEEE ICCV*, 2017. 12

[11] M. Cicconet, D. G. Hildebrand, and H. Elliott, "Finding mirror symmetry via registration and optimal symmetric pairwise assignment of curves: Algorithm and results," in *Proc. IEEE ICCV Workshops*, 2017. 12

[12] F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Revisiting Oxford and Paris: Large-scale image retrieval benchmarking," in *Proc. IEEE CVPR*, 2018. 12

[13] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang, "Spatial as deep: Spatial cnn for traffic scene understanding," in *AAAI*, 2018. 13