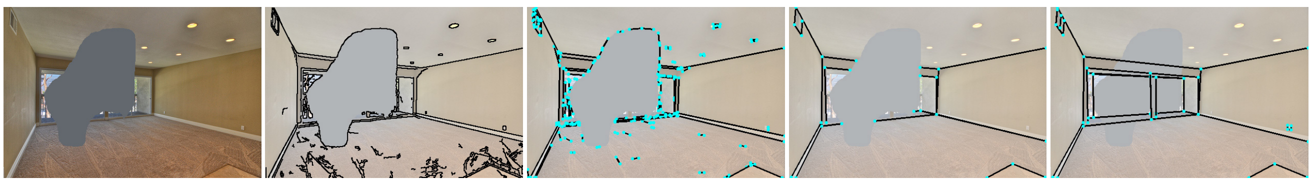
Hole-robust Wireframe Detection

Naejin Kong, Kiwoong Park, Harshith Goka
Samsung Research AI Center
{naejin.kong, kyoong.park, h9399.goka}@samsung.com



(a) Image with hole (b) Canny Edges [1] (σ=1) (c) Linelet [2] (d) HT-HAWP [16] (e) Ours

Figure 1: **Conventional edges or line segments ((b) and (c)) compared with *"Wireframes"* ((d) and (e)).** (a) Input image occluded by a large hole in dark gray color. Conventional edge and line segment detection such as (b) and (c) does not tell which of those detected are meant for salient structure. Wireframes are far better aligned with salient structural information of the scene as in (d) and (e). However, existing wireframe detection such as [16] in (d) still does not know how to handle hole occlusion, while our approach in (e) handles it very robustly. (black contours: edges/lines, light blue dots: junctions)

Abstract

"Wireframe" is a line segment based representation designed to well capture large-scale visual properties of regular, structural shaped man-made scenes surrounding us. Unlike the wireframes, conventional edges or line segments focus on all visible edges and lines without particularly distinguishing which of them are more salient to man-made structural information. Existing wireframe detection models rely on supervising the annotated data but do not explicitly pay attention to understand how to compose the structural shapes of the scene. In addition, we often face that many foreground objects occluding the background scene interfere with proper inference of the full scene structure behind them. To resolve these problems, we first time in the field, propose new conditional data generation and training that help the model understand how to ignore occlusion indicated by holes, such as foreground object regions masked out on the image. In addition, we first time combine GAN in the model to let the model better predict underlying scene structure even beyond large holes. We also introduce pseudo labeling to further enlarge the model capacity to overcome small-scale labeled data. We show qualitatively and quantitatively that our approach significantly outperforms previous works unable to handle holes, as well as improves ordinary detection without holes given.

1. Introduction

More and more people these days live in cities, hang around urban places, or spend most time indoors due to the pandemic. These environments surrounding us are cluttered with man-made architectures and artificial objects that appear to be in regular, structural shapes instead of random, repeated patterns in natural scenery, due to reasons to achieve structural integrity, ease of mass production, physical balance and so on. *"Wireframes"* [11] can represent this visual property effectively and compactly, as a set of line segments and their crossing junctions aligned well with straight and boxy corners salient to the man-made shapes.

Conventional edges or line segments may not be suitable for this task, as they focus on all visible edges and lines without particularly distinguishing which of them are more salient to man-made structural information. For instance, Canny edge detection [1] may produce spurious edges that include all kinds from texture change, object boundaries, or illumination difference (e.g., Figure 1 (b)). Adjusting the global threshold to avoid too many edges may end up with too sparse and discontinuous detection either. Moreover, edge detection is easily distracted by strong noise and illumination in the pixel signals that disturb recognizing the true underlying structure. Conventional line segment detection such as [2] in Figure 1 (c) would still show similar issues except being better aligned with straight structures.

In contrast, wireframes focus on the salient structural shapes that dominantly appear in the scene. This property makes them a far better representation to man-made scene structure, as shown in Figure 1 (d) and (e). However, existing wireframe detection models such as [16] in (d) do not explicitly pay attention to understand the underlying structural composition of the scene, but rely solely on the supervision of labeled wireframes in the dataset. As a result, these methods are unable to predict hidden structure beyond hole occlusion due to insufficient knowledge. On the contrary, our approach is designed to more directly understand the scene composition by learning to regenerate unknown scene contents inside large hidden areas given by holes, thus better fulfills the purpose of wireframes and produces highly hole-robust detection as in Figure 1 (e).

This hole-robustness is important, as we often face that a desired background scene is occluded by many foreground objects, such as furniture and electric gadgets in indoor scenes or passerby and cars in street scenes. Hence, a more practical wireframe detection algorithm should be able to work around the occlusion and infer the correct structure underneath. Note that these occlusions can be indicated by holes masked out on the image, by using existing foreground object segmentation models (ex., Detectron2 [28]) or manual user annotation.

Our hole-robust detection solution is novel in three folds:

1. Proposing new **conditional data generation and training** for hole-robust detection
2. Combining **GAN** [9] with the supervised model to improve its capacity to infer underlying structure
3. Introducing semi-supervised learning with **pseudo labeling** to further enlarge model capacity

Moreover, our approach is flexible and general, applicable to any recent models such as HAWP [31] (HT-HAWP [16]) and L-CNN [35] (HT-LCNN [16]), as well as very recent F-Clip [3] (preprint still under review).

To the best of our knowledge, we are first to raise an issue on this hole-robustness problem for wireframes along with a novel solution. Our approach is not only bound to hole-robust detection but also enhances the ordinary detection performance when holes are not given, thanks to an enhanced scene understanding capability and enlarged model capacity achieved from our novel hole-robust approach.

2. Related Works

Traditional line segment detection. Line segment detection is one of the classic and core tasks in computer vision. Traditional line segment detectors [27, 2, 19] rely on direct pixel signal features, such as image gradients. Another popular and classic approach is the Hough-transform line detection [6], which globally detects straight lines with an angle-distance based representation of the line segments. However, classical line segment detection methods have a limitation such that there is no way to tell how the lines are connected to each other by junctions, thus produce many uncorrelated scattered lines as shown in Figure 1 (c).

Deep learning based approaches. Since the main concept on wireframes [11] emerged, various methods have been proposed to detect the wireframes based on deep learning in three types of mainstream approaches so far.

One mainstream approach first identifies junctions and then finds connectivity in between two sampled junctions as the endpoints of each line segment. [11] proposed a framework that maintains two independent network branches to detect a junction heatmap and a line heatmap in parallel, which are then merged to compose line segments. Inspired by this work, [35] proposed the first end-to-end wireframe detection solution in a two-stage design, called L-CNN. This framework is motivated by typical object detection methods [8, 7, 23, 10], where the first stage predicts both the pixel-wise junction and line heatmaps, and then the second stage samples initial line segment candidates by using the detected junctions and deep feature maps. Then, these line segment proposals are verified through sophisticated classification to leave only true positive line segments. PPGNet [33] predicts a junction heatmap only, and then infers line segment candidates by using the predicted junctions as well as an adjacency matrix formulated from those junctions.

Another mainstream approach pursues direct line segment detection from the deep network without going through intermediate line segment sampling, based on a region-partition based representation of the line segments. This representation interprets a line segment vector as an angle-distance combination with respect to a polar coordinate system. [30] introduced a 6-D Attraction Field Map where every pixel in the AFM map corresponds to only one line segment. [31] refined this Attraction Field Map representation, and proposed a framework called HAWP, that uses a 4-D holistic Attraction Field Map composed of four channels; a distance channel with respect to distance from a pixel to its projected point onto a line segment, an angular orientation channel with respect to global rotation of the line segment, and two more angular channels with respect to the angles of two endpoints. Recently, [16] discovered that a Hough-transform [6] based global line prior combined with the backbone network of L-CNN [35] and that of HAWP [31] further improves their performances.

Inspired by object detection that has evolved from a two-stage scheme [8, 7, 23, 10] to a one-stage scheme [22, 18, 15, 14], there have been very recent attempts to follow a similar one-stage scheme for the wireframe detection. F-Clip [3] (preprint still under review) introduced a center point based representation of a line segment vector, which achieved a high speed while maintaining the ac-

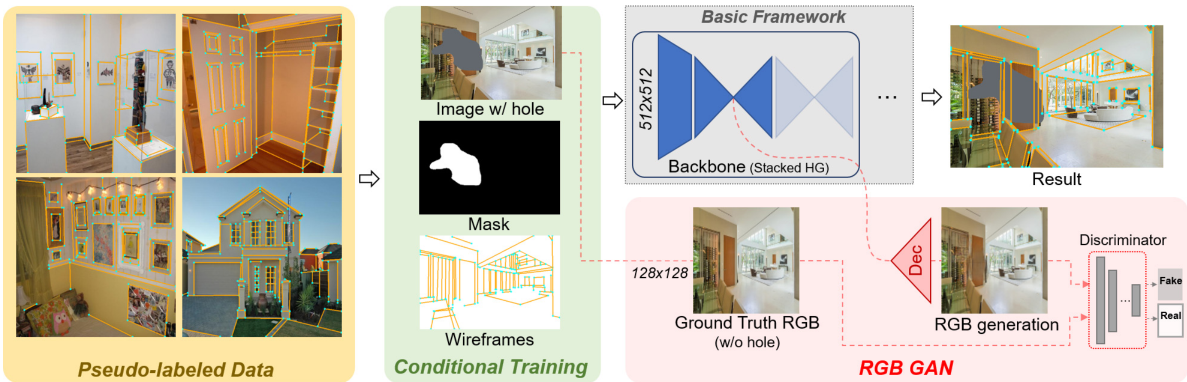


Figure 2: Overview of our approach (see **SuppMat:Sect.2** for details on the RGB GAN architecture).

curacy on par with HAWP [31] by using a stacked HourGlass [21] backbone commonly shared in recent works. They also showed that slower but higher accuracy can be achieved by replacing the backbone with HRNet [25] that is far more complex and fundamentally different from the stacked HourGlass. TP-LSD [29] also detects line segments in a single stage by introducing a tri-point representation composed of a center point as well as two endpoints, but it underperforms HAWP [31]. LETR [29] introduced Transformers [26] to the wireframe detection task for the first time. It is very slow due to the overhead of the Transformers, and does not show a clear advantage against HAWP [31] on the standard Wireframe test set [11].

We select L-CNN [35] (and HT-LCNN [16]), HAWP [31] (and HT-HAWP [16]), and F-Clip [3] as the representative frameworks from the three mainstream approaches, and prove that our approach applied to these frameworks improves hole-robustness as well as ordinary detection.

3. Approach

In this section, we choose HT-HAWP [16] as our representative basic framework, because it is the best performing one on conventional wireframe detection among recent models sharing the stacked HourGlass [21] backbone. We will show improvements with our approach applied to other basic frameworks in Section 4. Figure 2 overviews our approach, where the basic framework (gray shaded) can be flexibly chosen from other recent models. The yellow, green and red shaded modules compose our full approach.

3.1. Basic Framework: HT-HAWP

We call an HAWP framework [31] with Hough-transform priors [16] as "HT-HAWP", which is adopted as our representative basic framework. The HAWP framework [31] is composed of three steps as follows: 1) stacked HourGlass [21] backbone that predicts a 4-D Attraction Field Map that represents line segment vectors in terms of a 4-channel dense map as well as a junction heatmap that encodes the positions and offsets of the junctions in [-1,1] normalized coordinates, 2) matching raw predicted line segments and junctions, and 3) verifying the matched line segments through classification that outputs a final set of verified line segments coupled with their scores, where a segment with a higher score is more probable to be true positive. In addition, as shown in [16], adding inside the backbone a learnable Hough-transform [6] based mini-network that extracts explicit line-like features from interim network features can further improve the line detection. It can be repeatedly used in between residual modules, the basic building blocks of HourGlass (see **SuppMat:Sect.2** for details).

3.2. Conditional Training

We first redesigned the original training for this model to new conditional training with mask holes, as depicted in a green shaded area in Figure 2. For this, we reformed the original Wireframe training dataset [11] of 5,000 ⟨image, wireframe⟩ pairs into new ⟨image with hole, wireframe, mask⟩ pairs by superimposing 0.1-10% size real foreground object silhouettes onto each target image (see Figure 3), where these silhouettes were detected from a Places365 test set [34] using a public segmentation library [28]. Larger then 10% size did not show a clear benefit. We express the pixel value inside the hole using a mean pixel value over all RGB images in the dataset, which performed better than using a peak value such as zero or one, as each input image is normalized by subtracting this mean value during training.

We train on this new conditional dataset by progressively increasing the hole size along with epoch increment, which performed better than choosing a hole in random size. Figure 3 visualizes this. In addition, we place the hole by avoiding isolated components in the scene, such as a window, picture frame, home appliance like TV, or any tiny object that may form a loop of lines. As the hole goes larger, it may become more probable that an isolated line loop is fully contained within the hole, and then known regions

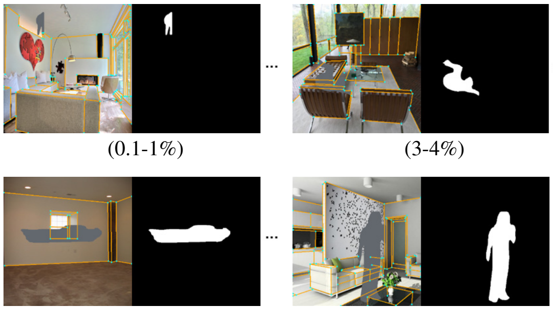
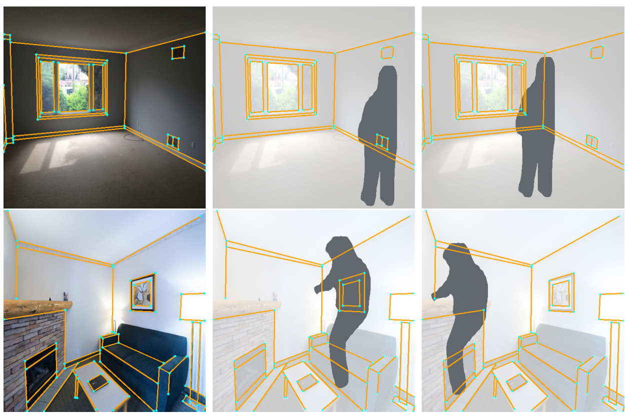


Figure 3: Conditional data with progressive hole size increment.



(a) No hole (b) Random (c) Avoid-isolation

Figure 4: **Hole placement schemes**. Random placement as (b) may entirely hide scene components unlike (c) that avoids such isolation. (Hole size: 9.91% above, 9.96% below)

may not provide any clue to infer the entire unknown structure. Figure 4 shows examples without and with the scheme applied, which makes visually convincing why we need to avoid isolation. **SuppMat:Sect.3** describes further details on our conditional data generation and training methods. Ablation studies in **SuppMat:Sect.6.1** justify our choices.

3.3. Combining GAN

The conditional training provides a huge improvement on hole-robustness, but as the hole goes larger, it may become harder to detect line segments without predicting the scene composition inside, at least roughly. To further support this case, we transform the original supervised backbone network [21] into an RGB GAN generator by adding a small 128×128 decoding branch to the first bottleneck along with a PatchGAN discriminator [12] as shown in the red shaded module of Figure 2. Training the model with our RGB GAN encourages that the backbone's encoder in connection with the RGB decoder learns to better understand the overall scene composition principles by referring to the background contents outside the hole, which helps regenerate invisible scene contents inside the hole as realistically as possible. We also tested GANs directly applied to a junction heatmap, line heatmap, or attraction field map, but found that it was not as effective as the RGB GAN (see **SuppMat:Sect.6.2** for more details).

We derived losses to train the RGB GAN by referring to a basic formulation in [20] for GAN-based *image inpainting*. As to be described below, we adopted some of their unique choices, such as to use an L1 norm instead of a squared L2 or Frobenius norm in the original form of a perceptual or style loss, or to use five layers in the VGG-19 network instead of VGG-16 for the perceptual or style loss. However, note that our RGB GAN does not aim at perfect image inpainting but is purposed to encourage the backbone to better understand overall scene structure composition principles.

Let G is the generator network (i.e., early part of the backbone + RGB decoder) and D is the discriminator network. Then, we define an adversarial loss [9] for an objective $\min_G \max_D \mathcal{L}_{adv}$ as:

$$\mathcal{L}_{adv} = \mathbb{E}_{\mathbf{x}} \left[\log D(\mathbf{x})\right] + \gamma \cdot \mathbb{E}_{\tilde{\mathbf{x}}} \log\left[1 - D(\tilde{\mathbf{x}})\right] \quad (1)$$

where a generated RGB image is $\tilde{\mathbf{x}} = G(\mathbf{x} \odot (1 - \mathbf{m}), \mathbf{m})$, $\mathbf{x}$ is a ground truth image without hole that is resized into 128×128, $\mathbf{m}$ is a binary mask map (1 inside the hole and 0 outside), $\odot$ is an element-wise multiplication, and $\gamma = 0.1$. We use a non-saturating version of Eq. (1) for the generator loss to avoid vanishing gradients, which minimizes $-\mathbb{E}_{\tilde{\mathbf{x}}} \log\left[D(\tilde{\mathbf{x}})\right]$ instead of $\mathbb{E}_{\tilde{\mathbf{x}}} \log\left[1 - D(\tilde{\mathbf{x}})\right]$.

We add a perceptual loss [13] to encourage perceptually more realistic looking generation:

$$\mathcal{L}_{per} = \sum_{i=1}^{L} \frac{1}{C_i H_i W_i} \left\| \Phi_1(\mathbf{x}) - \Phi_i(\tilde{\mathbf{x}}) \right\|_1 \quad (2)$$

where Φ_i is a $C_i \times H_i \times W_i$ feature map of the i-th layer from a VGG-19 [24] network, pretrained on the ImageNet dataset [4]. In our implementation, we used *relu1_1*, *relu2_1*, *relu3_1*, *relu4_1* and *relu5_1* layers (i.e., $L = 5$).

We also add a style loss [17] on the same pretrained VGG-19 layers adopted in Eq. (2).

$$\mathcal{L}_{sty} = \sum_{i=1}^{L} \left\| G_j^{\Phi}(\mathbf{x}) - G_j^{\Phi}(\tilde{\mathbf{x}}) \right\|_1 \quad (3)$$

where $G_j^{\Phi}(\mathbf{x})$ is a $C_j \times C_j$ *Gram* matrix from a $C_j \times H_j \times W_j$ feature map Φ_i, where each element at (c, c') in the matrix is $\frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \Phi_j(\mathbf{x})_{(h,w,c)} \Phi_j(\mathbf{x})_{(h,w,c')}$.

We then provide a reconstruction loss over the known pixels (i.e., outside the mask hole) in a masked L1 norm:

$$\mathcal{L}_{rec} = \frac{1}{N} \left\| (\mathbf{x} - \tilde{\mathbf{x}}) \odot (1 - \mathbf{m}) \right\|_1 \quad (4)$$

where N is the number of non-zero pixels in $(1 - \mathbf{m})$ to get an average. In our experiments, a feature matching loss [32] recently known to be useful to regularize GAN training did not help much but rather degraded the detection quality, so we decided to drop it. Finally, our total GAN loss $\mathcal{L}$ is:

$$\mathcal{L} = \lambda_{adv} \cdot \mathcal{L}_{adv} + \lambda_{per} \cdot \mathcal{L}_{per} + \lambda_{sty} \cdot \mathcal{L}_{sty} + \lambda_{rec} \cdot \mathcal{L}_{rec} \quad (5)$$

where we use $\lambda_{adv} = 1$, $\lambda_{per} = 0.1$, $\lambda_{sty} = 250$, $\lambda_{rec} = 1$.

In **SuppMat:Sect.2.4** we visualize generated RGB images from our trained RGB GAN. It demonstrates that the backbone encoder learned to infer overall scene structure inside the hole, which will be then, propagated over the rest of the model to induce more robust structural estimation.

3.4. Further Enhancement with Pseudo Labeling

Existing models rely entirely on the supervision of the Wireframe training dataset [11] composed of 5,000 examples. The scale of this training data would be too small even with strong data augmentation. To alleviate this issue, we further introduce a semi-supervised learning approach through pseudo labeling, which increases the training scale without need to add more manual annotations. Indeed, this additional approach significantly leveled up the performance, as demonstrated by experiments in Section 4.

We created a new pseudo-labeled dataset by using the Places365 Challenge dataset [34], where the pseudo label means estimated wireframe line segments and junctions. First, we only choose about 3.3M images in 157 structural scene categories that we had manually defined (e.g., campus, terminal, hall, store, and so on), out of all 8M images. Next, we retrieve estimated wireframes on these images by applying the pretrained HT-HAWP [16] model. Finally, we filter out supposedly less accurate or less densely annotated examples based on the three criteria:

- Number of lines in an image > 74.98
- Total length of all lines in an image > 6456.57
- The ratio $\frac{\text{\# of Junctions}}{\text{\# of Lines}}$ in an image < 1.34

The threshold values in these criteria were determined by inspecting the distributions of ground truth wireframe labels in each image of the Wireframe training dataset [11], which follow quite ideal normal distributions. Assuming an ideal distribution of the pseudo labels would be similar to that of the ground truth labels, we applied the same thresholds from the ground truth dataset to filter out more possibly well detected and more dense labels from the candidate pseudo-labeled images. **SuppMat:Sect.4** depicts three criteria histograms and justifies an effectiveness of this thresholding.

Once the thresholds are applied, 142k images remain in the final pseudo labeled dataset. This is about 28 times larger than the 5k images in the original Wireframe training set. The yellow shaded area in Figure 2 shows visualization of the pseudo-labeled examples sampled from the final pseudo dataset, demonstrating the great annotation quality. Note that these pseudo-labeled images are still without hole, that is, they have not yet transformed into conditional data.

We transform the Wireframe training set with hole by applying conditional generation in Section 3.2 with avoid-isolation. We also transform the pseudo-labeled data in the similar manner, but with random placement. We avoid additionally generating a huge mask pool for all 142k images but take random masks from the pre-generated mask pool for the Wireframe training set (10 candidates $\times$ 10 hole intervals $\times$ 5,000 images), which was still enough to enhance the performance significantly. We first train on the large-scale pseudo data with hole in order to enlarge the model capacity, and then finally fine-tune on the small-scale ground truth data with hole to further enhance fine-level accuracy.

4. Experiments

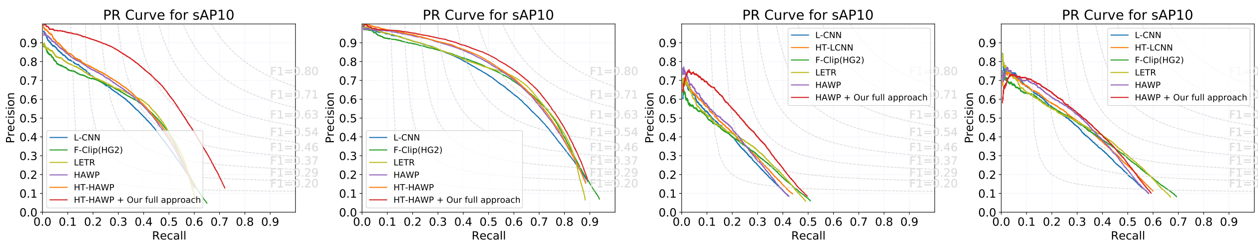
We provide implementation details in **SuppMat:Sect.5**. In order to test the hole-robust performance, we created new test sets with hole derived from standard test sets: the Wireframe test set [11] (462 examples) and the York Urban dataset [5] (102 examples all used for testing. Created in 2008 thus, its labeling may not be fully consistent with a more recently emerged wireframe concept). We superimposed an object silhouette hole in each image of the test set by following the similar manner described in Section 3.2. Specifically, we placed a hole by avoiding isolated components within the scene in case of 0-10% hole size (starting from 0.1% in practice), and placed it at an entirely random location in case of 10-30% hole size since it is very hard to avoid isolation with such a large hole (see Figure 4 to compare two cases). We also test the ordinary detection performance using the original standard test sets [11, 5] without hole, therefore not taking account of any hole occlusion.

Note that the runtime speed of a model applying our approach is similar to its basic model, as the RGB decoder branch is eliminated without affecting the detection result at inference time. Ours using HT-LCNN or HT-HAWP [16] may be slightly slower than the basic model, as we replace the first residual module with a whole HourGlass unit.

We show the effect our approach on various basic frameworks in Section 4.1. Due to space, we show further analysis and studies in **SuppMat:Sect.6** as follows: **⟦6.1⟧** Ablation on our conditional training made in various settings to show the effectiveness of progressive hole size increment against random size hole selection and the avoid-isolation scheme against random location placement. **⟦6.2⟧** Applying GAN [9] to various types instead of RGB, such as a junction, line, or attraction field map, to show how we had concluded RGB GAN is the best one. **⟦6.3⟧** Testing the sole effect of pseudo labeling without mixing holes on the ordinary wireframe detection performance to show it is still effective but limited to improve hole-robustness significantly with-

	Wireframe Test Set [11]												York Urban Dataset [5]											
	10-30% Hole				0-10% Hole				without Hole				10-30% Hole				0-10% Hole				without Hole			
	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H
L-CNN [35]	32.83	35.78	40.4	61.3	47.73	51.68	53.2	75.3	58.91	62.86	59.4	80.3	14.65	16.16	21.9	44.6	20.23	22.15	27.5	54.6	24.33	26.39	30.4	58.0
HT-LCNN [16]	33.56	36.48	41.4	57.1	48.89	52.79	54.2	76.6	60.33	64.22	60.6	81.6	15.84	17.48	23.5	40.4	21.47	23.62	29.3	50.1	25.72	28.02	32.5	53.0
F-Clip(HG2) [3]	33.51	36.68	/	65.3	49.46	53.87	/	79.5	61.25	65.77	/	84.3	15.86	17.54	/	48.6	22.02	24.07	/	58.6	27.10	29.26	/	62.1
F-Clip(HR) [3]	34.56	37.53	/	66.3	51.44	55.60	/	80.6	64.30	68.34	/	85.7	16.53	18.25	/	50.4	23.53	25.76	/	**61.3**	**28.49**	**30.80**	/	**65.0**
LETR [29]	32.84	37.23	/	69.5	50.24	56.22	/	82.5	59.19	65.64	/	86.1	15.55	18.26	/	49.1	21.40	25.33	/	59.1	25.65	29.61	/	62.1
HAWP [31]	34.80	37.74	40.8	64.5	50.80	54.84	54.0	79.4	62.52	66.49	60.2	85.0	15.79	17.48	22.5	46.3	21.61	23.79	28.6	57.3	26.15	28.54	31.6	61.3
HT-HAWP [16]	35.64	38.54	41.8	65.5	51.58	55.50	55.1	80.1	63.26	67.12	61.3	85.7	15.57	17.18	23.2	46.8	21.45	23.63	29.0	56.6	25.31	27.65	31.9	60.7
HAWP + Our full approach	47.59	51.50	49.6	74.9	61.83	65.98	61.0	85.0	65.98	69.76	63.0	86.9	**20.81**	**23.01**	**27.0**	**53.4**	**25.62**	**28.01**	**31.7**	60.2	27.23	29.58	**32.8**	62.6
HT-HAWP + Our full approach	**48.02**	**51.82**	**49.9**	**76.6**	**62.21**	**66.31**	**61.3**	**85.5**	**66.19**	**69.92**	**63.2**	**87.0**	19.96	21.92	25.4	52.9	24.72	26.94	29.8	57.6	26.06	28.26	30.9	59.2

Table 1: **Effect of our full approach, evaluated on the Wireframe test set [11] and York Urban dataset [5], *with* and *without* hole.** sAP^T (T: distance threshold) and mAP^J are positional accuracy for line segments and junction locations, respectively, and AP^H is a line heatmap based metric, all widely used in the field. F-Clip [3] does not predict junctions thus mAP^J is unavailable. Note that in gray marked rows, F-Clip(HR) uses an HRNet [25] backbone, and LETR largely relies on Transformer [26] encoder-decoder, which make them both fundamentally very different from the other frameworks sharing the stacked HourGlass [21] backbone.



(a) Wireframe [11], **10-30% hole** (b) Wireframe [11], ***without* hole** (c) York Urban [5], **10-30% hole** (d) York Urban [5], ***without* hole**

Figure 5: PR curves for sAP^{10} on the tests *with* and *without* hole in Table 1. Those on 0-10% hole and more PR curves for AP^H can be found in **SuppMat:Sect.8**.

out conditional training and RGB GAN further combined. **SuppMat:Sect.9-11** provide extra studies on the effect of initialization, light conditions, and inpainted input.

4.1. Effect of Our Approach

Table 1 compares our full models applying all of the conditional training, RGB GAN and pseudo labeling, with recent models that are incapable of handling hole occlusion, by employing widely used accuracy metrics sAP^T (T is a distance threshold): structural average precision with respect to line segment positions where a smaller T makes the metric more challenging, mAP^J: mean average precision of junction positions, and AP^H: average precision of rasterized line heatmaps. See [35] for more details on these metrics. We show and compare their F-scores in **SuppMat:Sect.7**.

Our full models show the best hole-robust performance at all metrics on both standard test sets (except for a second best AP^H on York Urban 0-10%). Also, our approach contributes to improved ordinary detection without hole as well: on the Wireframe test set [11] without hole, our full models outperform all of the recent models. On the York Urban dataset [5] without hole, our full model applied to the HAWP framework is the second best right after the very recent F-Clip(HR) [3] (preprint work yet unverified by peer reviews) on sAP^5 and AP^H, and is the best on mAP^J. Note that our full model always significantly outperforms its baselines, either HAWP [31] or HT-HAWP [16], with very large margins. In Section 4.2, we show improvements by our approach applied to more various basic frameworks.

In Figure 5, we show Precision-Recall curves for sAP^{10} with respect to models tested in Table 1. The curves demonstrate that models with our full approach significantly outperform existing models when holes are given, and also works better than those when no hole is given. We show PR curves for sAP^{10} on the 0-10% hole and more PR curves for the AP^H metric in **SuppMat:Sect.8**.

Note that F-Clip(HR) [3] and LETR [29] are fundamentally different from the other frameworks thus hard to be compared in parallel. F-Clip(HR) is mostly benefited from using a HRNet [25] backbone in a complex cascade architecture, whose complexity and properties differ fundamentally from stacked HourGlass [21] shared among recent works as well as F-Clip(HG2) [3]. LETR largely relies on Transformer [26] encoder-decoder where a CNN backbone plays only to extract features, which also makes it fundamentally different from the other frameworks.

Once F-Clip(HR) and LETR are excluded from Table 1, our full model, HAWP + Our full approach, presents the best numbers at all metrics on the York Urban dataset [5] regardless with or without hole. Note that the York Urban dataset is not entirely suitable for evaluating wireframe detection: it is not fully annotated as pointed out in [11, 16],

Approach				Wireframe Test Set [11]												York Urban Dataset [5]											
				10-30% Hole				0-10% Hole				without Hole				10-30% Hole				0-10% Hole				without Hole			
	3.2	3.3	3.4	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H	sAP^5	sAP^{10}	mAP^J	AP^H
L-CNN [35]				32.83	35.78	40.4	61.3	47.73	51.68	53.2	75.3	**58.91**	**62.86**	59.4	**80.3**	14.65	16.16	21.9	44.6	20.23	22.15	27.5	54.6	24.33	26.39	30.4	**58.0**
	✓			39.57	43.17	45.4	67.2	52.26	56.50	56.4	76.1	57.58	61.59	59.0	79.3	18.99	20.69	**25.6**	42.7	**23.86**	**26.06**	**30.9**	49.2	**25.86**	**28.03**	**32.4**	51.5
	✓	✓		**40.84**	**44.63**	**46.2**	**68.9**	**54.01**	**58.30**	**57.3**	**77.5**	58.88	62.85	**59.6**	80.1	**19.02**	**20.91**	25.1	**49.6**	23.47	25.61	30.0	**55.5**	25.25	27.42	31.3	57.6
HT-LCNN [16]				33.56	36.48	41.4	57.1	48.89	52.79	54.2	76.6	60.33	64.22	60.6	81.6	15.84	17.48	23.5	40.4	21.47	23.62	29.3	50.1	**25.72**	**28.02**	**32.5**	53.0
	✓			42.08	45.69	47.6	70.8	55.21	59.29	58.5	78.7	60.20	64.11	60.9	81.2	19.12	21.17	**26.1**	44.7	23.36	25.63	**30.7**	50.0	24.87	27.22	31.9	52.0
	✓	✓		**43.35**	**47.06**	**48.3**	**72.1**	**57.02**	**61.02**	**59.3**	**80.2**	**61.57**	**65.40**	**61.6**	**82.4**	**19.77**	**21.63**	25.7	**51.5**	**23.73**	**25.95**	29.9	**56.6**	25.37	27.58	31.1	**58.4**
(a): HAWP [31]				34.80	37.74	40.8	64.5	50.80	54.84	54.0	79.4	62.52	66.49	60.2	85.0	15.79	17.48	22.5	46.3	21.61	23.79	28.6	57.3	26.15	28.54	31.6	61.3
(b):	✓			44.00	47.78	46.9	72.3	57.37	61.64	57.8	82.6	62.25	66.11	60.2	84.9	19.06	21.01	25.0	51.1	23.56	25.86	29.7	57.8	25.33	27.53	31.0	60.2
(c):	✓	✓		44.17	48.11	47.1	73.1	57.99	62.35	58.2	82.9	62.60	66.60	60.5	85.1	20.30	22.34	25.8	53.0	24.90	27.35	31.2	**60.6**	26.69	29.08	32.4	62.3
(d):	✓		✓	47.36	51.15	**49.6**	74.5	61.16	65.30	60.7	84.2	65.60	69.34	62.9	86.2	20.50	22.58	**27.0**	52.0	25.52	27.99	**31.7**	59.2	27.13	29.54	**32.9**	61.2
(e):	✓	✓	✓	**47.59**	**51.50**	**49.6**	**74.9**	**61.83**	**65.98**	**61.0**	**85.0**	**65.98**	**69.76**	**63.0**	**86.9**	**20.81**	**23.01**	**27.0**	**53.4**	**25.62**	**28.01**	**31.7**	60.2	**27.23**	**29.58**	32.8	**62.6**
(f): HT-HAWP [16]				35.64	38.54	41.8	65.5	51.58	55.50	55.1	80.1	63.26	67.12	61.3	85.7	15.57	17.18	23.2	46.8	21.45	23.63	29.0	56.6	25.31	27.65	**31.9**	60.7
(g):	✓			45.11	48.94	48.2	74.4	58.59	62.85	59.2	83.7	63.31	67.19	61.4	85.5	18.62	20.60	25.4	52.8	23.47	25.72	**30.6**	59.3	25.02	27.32	31.6	**61.3**
(h):	✓	✓		45.72	49.56	48.0	75.0	59.33	63.59	59.1	84.0	63.56	67.49	61.2	85.6	19.44	21.39	24.6	52.8	24.35	26.62	29.6	**59.6**	25.91	28.16	30.5	61.0
(i):	✓		✓	46.82	50.37	49.2	75.0	60.50	64.48	60.3	84.4	64.90	68.59	62.4	86.4	18.99	20.89	**25.9**	**53.4**	23.23	25.34	30.3	59.1	24.84	26.92	31.3	61.2
(j):	✓	✓	✓	**48.02**	**51.82**	**49.9**	**76.6**	**62.21**	**66.31**	**61.3**	**85.5**	**66.19**	**69.92**	**63.2**	**87.0**	**19.96**	**21.92**	25.4	52.9	**24.72**	**26.94**	29.8	57.6	**26.06**	**28.26**	30.9	59.2
F-Clip(HG2) [3]				33.51	36.68	/	65.3	49.46	53.87	/	79.5	61.25	65.77	/	84.3	15.86	17.54	/	48.6	22.02	24.07	/	58.6	27.10	29.26	/	62.1
	✓			43.86	48.05	/	73.6	56.74	61.57	/	82.2	61.38	65.81	/	84.1	20.10	22.27	/	54.4	25.39	27.84	/	60.6	27.35	29.60	/	62.9
	✓	✓		**44.92**	**49.14**	/	**74.6**	**58.10**	**63.01**	/	**83.2**	**62.38**	**66.82**	/	**84.8**	**20.76**	**22.91**	/	**54.7**	**25.57**	**28.03**	/	**60.9**	**27.50**	**29.83**	/	**63.2**
F-Clip(HR) [3]				34.56	37.53	/	66.3	51.44	55.60	/	80.6	64.30	68.34	/	85.7	16.53	18.25	/	50.4	23.53	25.76	/	61.3	28.49	30.80	/	**65.0**
	✓			**47.39**	**51.48**	/	**76.0**	**61.36**	**65.78**	/	**84.7**	**65.31**	**69.39**	/	**86.1**	**22.50**	**24.57**	/	**56.4**	**28.27**	**30.88**	/	**62.9**	**29.98**	**32.50**	/	64.9

Table 2: **Applying our approach to various recent frameworks.** Approach in Section 3.2: *'Conditional Training'*, 3.3: *'RGB GAN'*, and 3.4: *'Pseudo Labeling'*. Best metrics within each framework group are marked in **bold** font. Our approach can be applied to various recent frameworks sharing the *stacked HourGlass* [21] backbone, and is consistently effective either with hole or without hole in general. See **SuppMat:Sect.8** to find PR curves on (a),(c),(e) and (f),(h),(j).

and was created based on a Manhattan assumption (scenes were built on a cartesian grid) that largely differs from the wireframe concept.

4.2. Applying to Various Frameworks

In Table 2, our approach was tested with various recent frameworks, L-CNN [35], HT-LCNN [16], HAWP [31], HT-HAWP [16] and F-Clip(HG2) [3], which share the stacked HourGlass [21] backbone. The results show that applying our full or partial approach is consistently effective to improve the performance on both test sets either without or with hole, except baseline L-CNN on the Wireframe test set [11] without hole and baseline HT-LCNN on York Urban [5] without hole. See **SuppMat:Sect.7** for their F-scores.

We also tested on F-Clip(HR) [3] (gray group in Table 2), whose backbone is based on HRNet [25] that fundamentally differs to stacked HourGlass [21]. Applying conditional training enhanced all metrics notably, but we could not turn this backbone into an RGB GAN generator in spite of various attempts. We will explore how to adapt our approach to this heterogeneous framework in the future work.

SuppMat:Sect.8 shows PR curves on (a),(c),(e) and (f),(h),(j) in Table 2. These curves prove that our approach yet without pseudo labeling is already significantly effective in improving hole-robustness, which is even further improved with pseudo labeling. Also, our approach enhances the ordinary detection performance without hole as well.

4.3. Qualitative Comparisons

In this section, we demonstrate the qualitative performance of our approach by taking a few examples from the Wireframe test set [11] and the York Urban [5], with hole and without hole. Figure 6 (a)-(c) show that existing works do not know how to handle hole occlusion, thus fail to detect inside the hole or mistakenly detect lines around the hole boundaries. In contrast, our approach in Figure 6 (d) detects lines and junctions well even across the large holes. Figure 7 shows that our approach still works consistently well on the ordinary detection when no hole is given. See **SuppMat:Sect.12** for more qualitative comparisons, and real-world examples with foreground object occlusions.

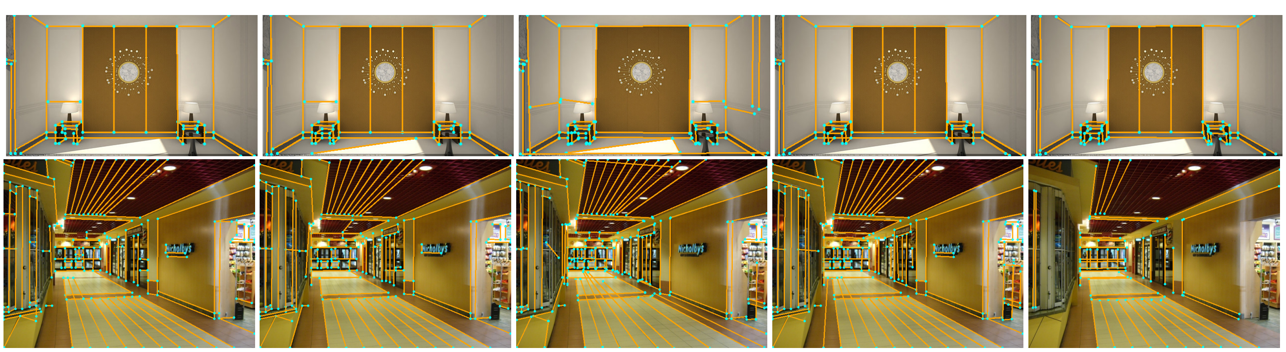
5. Conclusions

We propose a novel approach to detect wireframes in a hole-robust manner unlike any of the previous works. We derive this goal by introducing conditional training with masked data generation and special training techniques with respect to the hole size and hole placement, combining RGB GAN in the backbone network to let it better understand the scene composition principles while learning to regenerate hidden scene contents, and semi-supervised learning with pseudo labeling to further enlarge the model capacity.

As demonstrated with extensive experiments and analysis in Section 4 and **Suppmat:Sect.6-12**, our approach is highly robust to hole occlusion, which significantly outperforms existing works that are incapable to properly detect across holes, and even improves the ordinary detection performance when no hole is given. In the future work, we will further extend the approach to different backbone architectures, and explore global attention to enable even deeper scene understanding by merging local and global features.

(a) HT-LCNN [16] (b) HT-HAWP [16] (c) F-Clip(HR) [3] (d) Ours (e) Ground Truth

Figure 6: **Results on the Wireframe test set [11] (1st to 3rd rows) and York Urban dataset [5] (4th and 5th row) *with* hole.** (a)-(c) Existing works do not know how to handle holes occluding the scene. (d) Our full approach (using HT-HAWP [16] as the basic framework) robustly detect lines and junctions regardless of large holes given.



(a) HT-LCNN [16] (b) HT-HAWP [16] (c) F-Clip(HR) [3] (d) Ours (e) Ground Truth

Figure 7: **Results on the Wireframe test set [11] (1st row) and York Urban dataset [5] (2nd row) *without* hole.** Our full approach (using HT-HAWP [16] as the basic framework) still performs well on the ordinary detection without holes given.

References

[1] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):679–698, 1986.

[2] Nam-Gyu Cho, Alan Yuille, and Seong-Whan Lee. A novel linelet-based representation for line segment detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(5):1195–1208, 2017.

[3] Xili Dai, Xiaojun Yuan, Haigang Gong, and Yi Ma. Fully convolutional line parsing. *arXiv preprint arXiv:2104.11207*, 2021.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[5] Patrick Denis, James H Elder, and Francisco J Estrada. Efficient edge-based methods for estimating Manhattan frames in urban imagery. In *European Conference on Computer Vision (ECCV)*, pages 197–210. Springer, 2008.

[6] Richard O Duda and Peter E Hart. Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[7] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[8] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017.

[11] Kun Huang, Yifan Wang, Zihan Zhou, Tianjiao Ding, Shenghua Gao, and Yi Ma. Learning to parse wireframes in images of man-made environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 626–635, 2018.

[12] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017.

[13] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision (ECCV)*, 2016.

[14] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. RefineNet: Multi-path refinement networks for high-resolution semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1925–1934, 2017.

[15] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[16] Yancong Lin, Silvia L Pintea, and Jan C van Gemert. Deep Hough-transform line priors. In *European Conference on Computer Vision (ECCV)*, pages 323–340. Springer, 2020.

[17] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *European Conference on Computer Vision (ECCV)*, pages 85–100, 2018.

[18] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, pages 21–37. Springer, 2016.

[19] Xiaohu Lu, Jian Yao, Kai Li, and Li Li. CannyLines: A parameter-free line segment detector. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 507–511, 2015.

[20] Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Qureshi, and Mehran Ebrahimi. EdgeConnect: Structure guided image inpainting using edge prediction. In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 3265–3274, 2019.

[21] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision (ECCV)*, pages 483–499. Springer, 2016.

[22] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[23] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 91–99, 2015.

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[25] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5693–5703, 2019.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017.

[27] Rafael Grompone Von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):722–732, 2008.

[28] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. 2019. *URL https://github.com/facebookresearch/detectron2*, 2(3), 2019.

[29] Yifan Xu, Weijian Xu, David Cheung, and Zhuowen Tu. Line segment detection using transformers without edges. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4257–4266, 2021.

[30] Nan Xue, Song Bai, Fudong Wang, Gui-Song Xia, Tianfu Wu, and Liangpei Zhang. Learning attraction field representation for robust line segment detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1595–1603, 2019.

[31] Nan Xue, Tianfu Wu, Song Bai, Fudong Wang, Gui-Song Xia, Liangpei Zhang, and Philip HS Torr. Holistically-attracted wireframe parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2788–2797, 2020.

[32] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6721–6729, 2017.

[33] Ziheng Zhang, Zhengxin Li, Ning Bi, Jia Zheng, Jinlei Wang, Kun Huang, Weixin Luo, Yanyu Xu, and Shenghua Gao. PPGNet: Learning point-pair graph for line segment detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7105–7114, 2019.

[34] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2017.

[35] Yichao Zhou, Haozhi Qi, and Yi Ma. End-to-end wireframe parsing. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 962–971, 2019.