# Stroke-Based Scene Text Erasing Using Synthetic Data for Training

Zhengmi Tang, Tomo Miyazaki, *Member, IEEE*, Yoshihiro Sugaya, *Member, IEEE*, and Shinichiro Omachi, *Senior Member, IEEE*

***Abstract*— Scene text erasing, which replaces text regions with reasonable content in natural images, has drawn significant attention in the computer vision community in recent years. There are two potential subtasks in scene text erasing: text detection and image inpainting. Both subtasks require considerable data to achieve better performance; however, the lack of a large-scale real-world scene-text removal dataset does not allow existing methods to realize their potential. To compensate for the lack of pairwise real-world data, we made considerable use of synthetic text after additional enhancement and subsequently trained our model only on the dataset generated by the improved synthetic text engine. Our proposed network contains a stroke mask prediction module and background inpainting module that can extract the text stroke as a relatively small hole from the cropped text image to maintain more background content for better inpainting results. This model can partially erase text instances in a scene image with a bounding box or work with an existing scene-text detector for automatic scene text erasing. The experimental results from the qualitative and quantitative evaluation on the SCUT-Syn, ICDAR2013, and SCUT-EnsText datasets demonstrate that our method significantly outperforms existing state-of-the-art methods even when they are trained on real-world data.**

***Index Terms*— Scene text erasing, synthetic text, background inpainting.**

## I. Introduction

TEXTS created by humankind comprise rich, precise high-level semantics. Text, in daily life, provides a considerable amount of valuable information. However, with the increasing number of portable devices, such as digital cameras, tablets, smartphones, and SNS, a huge volume of scene images, including text, are shared on the Internet every second. These texts can contain private information, such as names, addresses, and vehicle number plates. With the increasing development of scene text detection and recognition technology, there is a high risk that the information collected automatically is used for illegal purposes. Therefore, scene text erasing, which is replacing text regions in scene images with proper content, has drawn considerable attention in the computer vision community in recent years.

After the prior work of Scene Text Eraser [1], scene text erasing research has developed in two directions: one-step and two-step methods. One-step methods [2]–[4] do not require the input of any text location information because they combine text detection and inpainting functions into one network, making them lightweight and fast. The drawback is that the text localization mechanism of these networks is weak, and the text-erasing process is not controllable. To allow the network to learn the complicated distribution of scene texts, a considerable number of manually text-erased real-world images are required as training data because the text distribution of the Synth-text [5] dataset, which is generated according to human rules, is significantly different from real-world cases. To generate scene-text-erased images [4], [6], photo editing software such as Photoshop is used to fill the text region of natural images with visually plausible content. However, this type of annotation is expensive and time-consuming because the annotators need to operate carefully to guarantee the quality of the erasing, especially when the text to be erased is on a complicated background.

The two-step approaches decompose the text-erasing task into two sub-problems: text detection and background inpainting. MTRNet [7] inpaints the text region by manually providing a mask of the text regions. Zdenek *et al.* [8] used a pretrained scene text detection model and a pretrained inpainting model to erase the text. This weak supervision method does not require paired training data. However, the inpainting model is trained on the Street View [9] or ImageNet [10] datasets, and thus, the pretrained models face domain shift problems, which cannot make a "perfect fit" in the context of scene text.

In this study, we propose a word-level two-stage scene-text-erasing network that works on cropped text images. First, it predicts the region of text stroke as a hole, then inpaints the hole according to the background. To compensate for the lack of pairwise real-world data, we made full use of synthetic texts. The appearance of the synthetic texts [5] was enhanced, and we trained our model only on the dataset generated by the improved synthetic text engine in an end-to-end fashion. The model can partially erase text instances in a scene image if text bounding boxes are provided. The model can also work with existing scene-text detectors for automatic scene text erasing. Examples of text-erasing results obtained using the proposed method are shown in Fig. 1.

Manuscript received March 10, 2021; revised August 2, 2021 and September 23, 2021; accepted October 25, 2021. Date of publication November 9, 2021; date of current version November 12, 2021. This work was supported in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant 18K19772, Grant 19K12033, and Grant 20H04201. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Xu-Yao Zhang. *(Corresponding author: Zhengmi Tang.)*

The authors are with the Graduate School of Engineering, Tohoku University, Sendai 980-8579, Japan (e-mail: tzm@dc.tohoku.ac.jp; tomo@tohoku.ac.jp; sugaya@iic.ecei.tohoku.ac.jp; machi@ecei.tohoku.ac.jp).

Digital Object Identifier 10.1109/TIP.2021.3125260

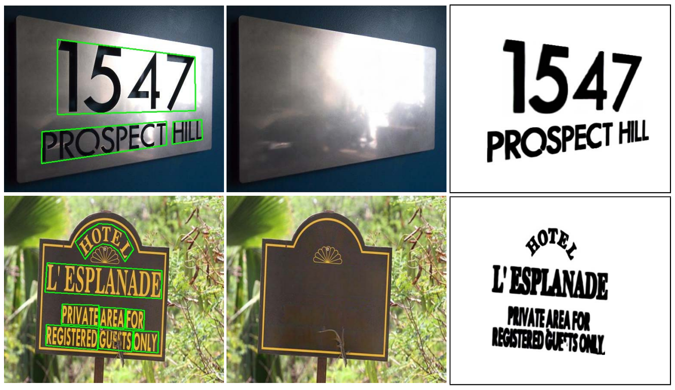


Fig. 1. Scene Text Erasing: original images and text bounding boxes (left); text-erased images by our method (middle); predicted text stroke mask (right).

The main contributions of our study can be summarized as follows:

- We propose a practical text erasing method and a stroke-based text-erasing network using cropped text images instead of the entire image, which makes prediction of the pixel-level mask of text strokes more accurate and stable. Benefiting from our erasing pipeline and network structure, our method can erase text instances while retaining and restoring more background details.
- We enhance the Text Synthesis Engine [5] to make the appearance of synthetic text instances share more similarity with real-world data.
- The quantitative and qualitative evaluation results on SCUT-Syn [3], ICDAR 2013 [11] and SCUT-EnsText [4] datasets demonstrate that our method outperforms previous state-of-the-art methods while it is only trained on the dataset generated by the improved synthetic text engine [5].

The remainder of this paper is organized as follows. Section II reviews related works on scene-text detection, image inpainting, and text erasing. Section III introduces the details of our method, including the pipeline and the proposed networks. In Section IV, we evaluate and compare our proposed method with related inpainting and scene-text erasing studies based on the results of experiments. Finally, we provide concluding statements in Section V.

## II. Related Work

### A. Scene Text Detection

The emergence of deep learning has facilitated the development of scene text detection research and shows promising performance compared to traditional manually designed algorithms [12]–[16]. Recent learning-based scene text detection methods can be roughly categorized into regression-based and segmentation-based methods. Regression-based methods aim to directly predict the bounding boxes of text instances. TextBoxes [17] adjusts the aspect ratios of anchors in SSD [18] to detect text with different shapes. CTPN [19] combines the framework of Faster R-CNN [20] with a recurrence mechanism to predict the contextual and dense components of text. RRPN [21] proposes a rotation region proposal to bind multi-oriented scene text with rotated rectangles. EAST [22] directly regresses rotated rectangles or quadrangles of text through a simplified pipeline without using any anchors. LOMO [23] detects long text and arbitrarily shaped text in scene images by refining the preliminary proposals iteratively and considering text geometry properties including text region, text center line, and border offsets.

Segmentation-based methods usually first extract text pixels from the segmentation map and then obtain bounding boxes of the text by post-processing. Zhang *et al.* [24] used the FCN and MSER for pixel-level multi-oriented text detection. Mask textspotter [25] built a model based on the framework of Mask R-CNN [26] and performed character-level instance segmentation for each alphabet. TextSnake [27] proposed a novel representation of arbitrarily shaped text and predicted heat maps of text center lines, text regions, radii, and orientations to extract text regions. PSENet [28] gradually expanded the text region from small to large kernels to make final predictions through multiple semantic segmentation maps. Liao *et al.* [29] proposed a module called differentiable binarization (DB) to perform the binarization process in a segmentation network. CRAFT [30] proposed learning each character center and the affinity between characters in the form of a heat map.

### B. Image Inpainting

Image inpainting fills the hole regions of an image with plausible content. Image inpainting research can generally be divided into two categories: non-learning and deep-learning-based approaches. Non-learning approaches transfer the surrounding content to the hole region based on low-level features using a traditional algorithm such as patch matching [31]–[33] and diffusion [34], [35]. Although these methods can work well on small holes, they cannot deal with large missing regions, which require semantics patches for the hole restoration based on a high-level understanding of the whole image.

The generative adversarial network (GAN) strategy has been widely adopted in recent deep-learning-based approaches and can be categorized as single-stage inpainting and progressive inpainting. Context encoders [36] first trained an image-inpainting deep neural network using an encoder–decoder structure and adversarial losses. Iizuka *et al.* [37] adopted dilated convolution and proposed global and local discriminators for adversarial training. Liu *et al.* [38] defined a partial convolutional layer with a mask-update mechanism to ensure that the partial convolution filters learn more valid information from the non-hole region and can robustly handle holes of any shape. Xie *et al.* [39] proposed a learnable bidirectional attention module that included forward and reverse attention maps for more effective hole filling. Li *et al.* [40] exploited the correlation between adjacent pixels by recurrently inferring and gathering the hole boundary for the encoder, referred to as the recurrent feature reasoning module.

To generate a more realistic texture, a coarse-to-fine strategy was adopted in the progressive inpainting methods. Yu *et al.* [41] exploited textural similarities to borrow feature information from known background patches to generate missing patches and designed a two-stage network architecture from coarse to fine. Yu *et al.* [42] introduced gated

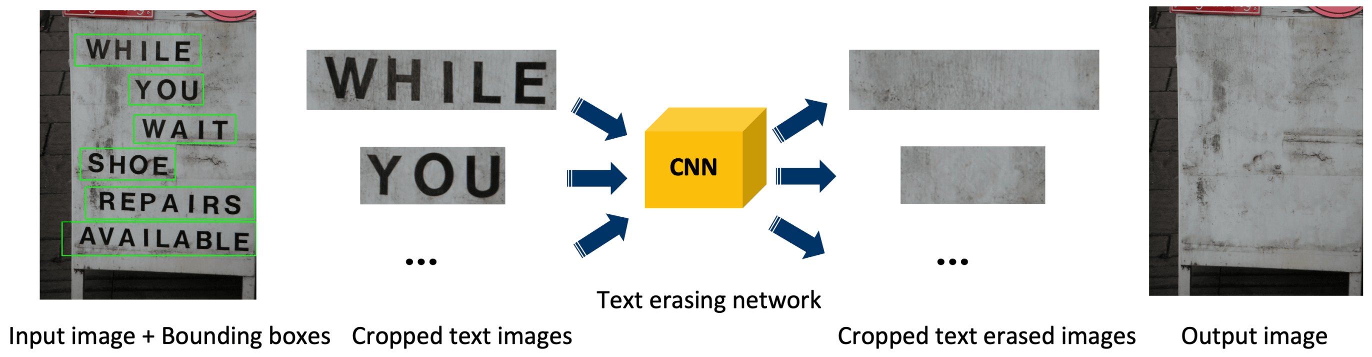


Fig. 2. Pipeline of our proposed method. Our text-erasing network uses cropped text images as inputs and outputs the corresponding text-erased images. Then, by placing these text-erased images back into the original locations of the input image, we can obtain the final text-erased image.

convolution, which further generalizes partial convolution to make the mask-update mechanism learnable. The method was then combined with the SN-PatchGAN discriminator to obtain better performance. Yi *et al.* [43] improved gated convolution through a lightweight design and proposed high-frequency residuals to generate rich and detailed textures for efficient ultra-high-resolution image inpainting.

### *C. Text Erasing*

Early text erasing research starts with the removal of born-digital text such as watermarks, captions, and subtitles on images or video sequences [44]. Owing to the plain layout, color, and regular font, born-digital text can be detected by traditional feature engineering approaches such as binarization [45], [46], and inpainted by patch matching [47] or smoothing algorithms [48].

Erasing text in the wild is a more complex and challenging task because of the various fonts with different layouts and illumination conditions. The recent rapid development of deep neural networks has made scene text erasure a promising research task in the computer vision field. Scene text erasing research can be classified into two categories: one-step and two-step methods. One-step methods use an end-to-end model to directly output a text-erased image without the aid of text location information. Pix2Pix [49] is a conditional generative adversarial network (cGAN) designed for general-purpose image-to-image translation tasks that can be applied to text removal. Nakamura *et al.* [1] made the first attempt to build a one-stage scene text eraser, a sliding window method using a skip-connected auto-encoder. This method destroys the integrity of text strokes but cannot maintain the global consistency of the entire image. EnsNet [3] adopted cGAN with a refined loss function and a local-aware discriminator to erase texts at the entire image level. Liu *et al.* [4] provided a comprehensive real-world scene-text removal benchmark, named SCUT-EnsText, and proposed EraseNet, which adopts a coarse-to-fine erasure network structure with a segmentation module to generate a mask of the text region to help with text region localization. MTRNet++ [2] shared the same coarse-to-fine inpainting idea but used a multi-branch generator. The mask-refine branch predicts stroke-level text masks to guide text removal.

Two-step methods remove the text with an awareness of text location, which can be provided by users or by a pretrained text detector. Qin *et al.* [50] proposed a cGAN with one encoder and two decoder architectures to perform content region segmentation and inpainting jointly, which can be applied for text removal in cropped text images. MTRNet [7] used manually provided text masks to guide the training and prediction processes of cGAN, thereby realizing the controllability of the text-erasing region. Zdenek *et al.* [8] proposed a weak supervision method employing a pretrained scene text detector [28] and a pretrained image inpainting model [51] to free the scene text erasing task from the requirement of paired-wise training data of scene images with text and the corresponding text-erased images. Bian *et al.* [6] proposed a cascaded GAN-based model to decouple text stroke detection and stroke removal in text removal tasks.

We consider our proposed method to be a practical solution to scene text erasing tasks because the pretrained scene text detector and synthetic data obviate the need for paired real-world data. The relatively weak text detection ability of one-step end-to-end methods can lead to excessive erasure of text-free areas and incomplete erasure of the text region. The detection ability of these networks can only be improved during end-to-end training, which would require expensive real-world training data. The advantage of our method over weak supervision methods is that using improved synthetic data for training can greatly reduce the domain shift problem during the inpainting process.

## III. Methodology

The pipeline of the proposed method is shown in Fig. 2. Given the source image containing the scene text and the corresponding text bounding boxes, text images are cropped from the source image. Subsequently, each cropped text image goes through the text-erasing network. The cropped text-erased images are then inserted into the source image to obtain an output image that does not contain text.

Our network comprises 1) a stroke mask prediction module and 2) a background inpainting module, as illustrated in Fig. 3. Specifically, the stroke mask prediction module first predicts the stroke mask of the scene text $\hat{I}_{mask}$ as the hole from $I_{in}$. Then, the background inpainting module is used to fill the hole

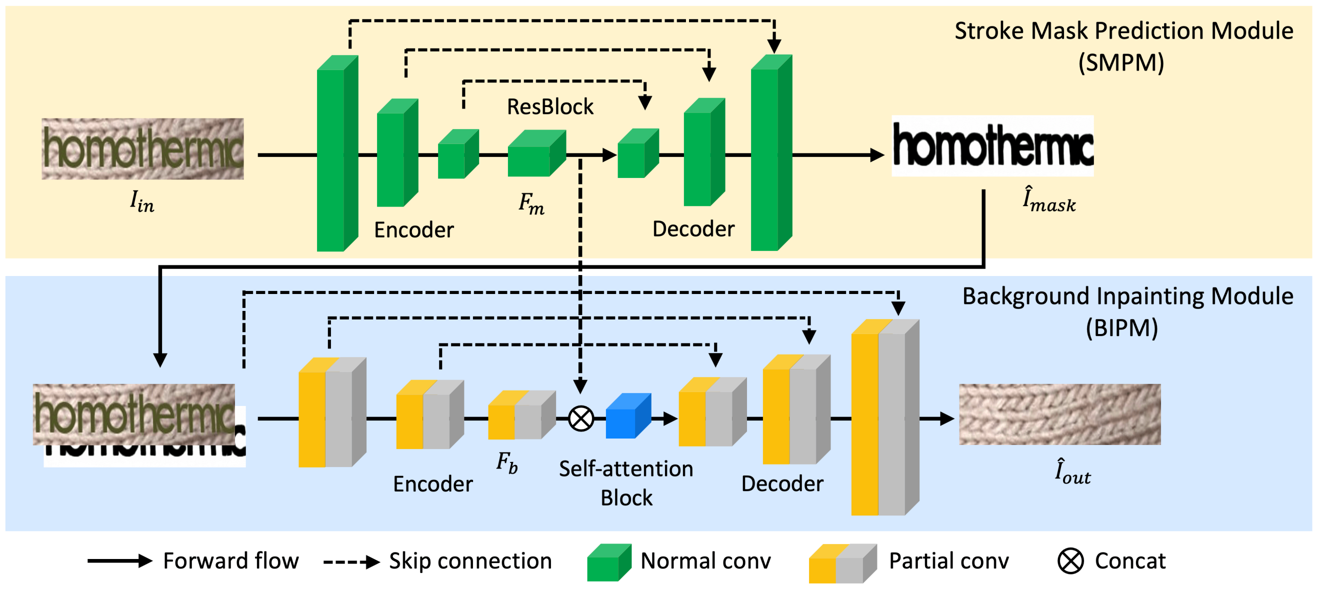


Fig. 3. Structure of our proposed network. It is composed of a stroke mask prediction module (SMPM) (top) and a background inpainting module (BIPM) (bottom). The SMPM first predicts the stroke mask of the text, and the BIPM inpaints the stroke pixels with the appropriate content to erase the text.

of the input image $I_{in}$ with the appropriate content and output text-erased image $\hat{I}_{out}$.

## A. Stroke Mask Prediction Module

An encoder–decoder FCN is adopted in this module. The input image $I_{in}$ is encoded by three down-sampling convolutional layers and four residual blocks [52], and the feature maps $F_m$ are decoded by three up-sampling transposed convolutional layers to generate the text stroke mask $\hat{I}_{mask}$. The architecture of the SMPM is presented in Table I. The skip connections concatenate feature maps of the same shape between the encoder and decoder. The feature maps $F_m$ output from the residual blocks are concatenated with the feature maps $F_b$ in the background inpainting module, which is introduced in Section B. There is usually an imbalance between the pixels of the text region and the pixels of the non-text region in $I_{in}$, and both high recall and precision are expected during mask prediction. Therefore, the dice loss [53] and L1 loss are employed to guide the generation of the text mask. In image segmentation, the dice loss is a region-based loss measure that expresses the proportion of correctly predicted pixels to the sum of the total pixels of both the prediction and ground truth. Mathematically, the dice loss is defined as

$$\mathcal{L}_{dice} = 1 - \frac{2\sum_i^N (\hat{I}_{mask})_i (I_{mask})_i}{\sum_i^N (\hat{I}_{mask})_i + \sum_i^N (I_{mask})_i}, \tag{1}$$

where N denotes the total number of pixels in the input image, and $\hat{I}_{mask}$ and $I_{mask}$ represent the prediction and ground truth of the text mask, respectively. The total loss of the stroke mask prediction module is

$$\mathcal{L}_{SMPM} = \|\hat{I}_{mask} - I_{mask}\|_1 + \lambda_0 \mathcal{L}_{dice}, \tag{2}$$

and in our experiments, $\lambda_0$ is set to 1.0.

TABLE I

ARCHITECTURE OF THE SMPM. CONV: CONVOLUTIONAL LAYER, DECONV: TRANSPOSED CONVOLUTIONAL LAYER, RESBLOCK: RESIDUAL CONVOLUTIONAL BLOCK, C: CHANNELS, K: KERNEL SIZE, S: STRIDE

| Layers | Configurations | Output |
|---|---|---|
| Conv×2 | c: 32, k: 3, s: 1 | 128×640 |
| Conv | c: 64, k: 3, s: 2 | 64×320 |
| Conv×2 | c: 64, k: 3, s: 1 | 64×320 |
| Conv | c: 128, k: 3, s: 2 | 32×160 |
| Conv×2 | c: 128, k: 3, s: 1 | 32×160 |
| Conv | c: 256, k: 3, s: 2 | 16×80 |
| Conv×2 | c: 256, k: 3, s: 1 | 16×80 |
| ResBlock×4 | c: 256, k: 3 | 16×80 |
| Conv×2 | c: 256, k: 3, s: 1 | 16×80 |
| Deconv | c: 128, k: 3, s: 1/2 | 32×160 |
| Conv×2 | c: 128, k: 3, s: 1 | 32×160 |
| Deconv | c: 64, k: 3, s: 1/2 | 64×320 |
| Conv×2 | c: 64, k: 3, s: 1 | 64×320 |
| Deconv | c: 32, k: 3, s: 1/2 | 128×640 |
| Conv×2 | c: 32, k: 3, s: 1 | 128×640 |
| Conv | c: 3, k: 3, s: 1 | 128×640 |

## B. Background Inpainting Module

In this module, the input image $I_{in}$ and the predicted mask image $\hat{I}_{mask}$ are taken as input, and the output is the background image $O_b$, in which all text stroke pixels, including text shadows caused by illumination, are replaced with proper texture. As shown in the blue part of Fig. 3, the input image is encoded by three down-sampling partial convolutional layers and concatenated with the feature map $F_m$ from the SMPM. This is followed by a self-attention block, which allows the network to learn long-range dependencies. Finally, the decoder generates the output image $\hat{I}_{out}$. The architecture of the BIPM is presented in Table II.

*1) Partial Convolutional Layers:* To generate clearer background images and suppress text ghosts and artifacts, we use partial convolutional layers [38] to allow the network to

TABLE II

ARCHITECTURE OF THE BIPM. PCONV: PARTIAL CONVOLUTIONAL LAYER, UPSAMPLE: UPSAMPLE LAYER, SCALE: SCALE FACTOR, C: CHANNELS, K: KERNEL SIZE, S: STRIDE

| Layers | Configurations | Output |
|---|---|---|
| PConv | c: 64, k: 7, s: 2 | 64×320 |
| PConv×2 | c: 64, k: 3, s: 1 | 64×320 |
| PConv | c: 128, k: 5, s: 2 | 32×160 |
| PConv×2 | c: 128, k: 3, s: 1 | 32×160 |
| PConv | c: 256, k: 3, s: 2 | 16×80 |
| PConv×2 | c: 256, k: 3, s: 1 | 16×80 |
| GCblock | c: 512, ratio: 4 | 16×80 |
| Conv | c: 256, k: 3, s: 1 | 16×80 |
| Upsample | scale: 2 | 32×160 |
| Pconv×2 | c: 256, k: 3, s: 1 | 32×160 |
| PConv | c: 128, k: 3, s: 1 | 32×160 |
| Upsample | scale: 2 | 64×320 |
| Pconv×2 | c: 128, k: 3, s: 1 | 64×320 |
| PConv | c: 64, k: 3, s: 1 | 64×320 |
| Upsample | scale: 2 | 128×640 |
| Pconv×2 | c: 64, k: 3, s: 1 | 128×640 |
| PConv | c: 3, k: 3, s: 1 | 128×640 |

learn more features from the non-text part of $I_{in}$. The partial convolution layer comprises two steps: the partial convolution operation and mask update. The partial convolution operation and mask update are defined as follows:

$$x' = \begin{cases} \mathbf{W}^T(\mathbf{X} \odot \mathbf{M})\dfrac{\text{sum}(\mathbf{1})}{\text{sum}(\mathbf{M})} + b, & \text{if sum}(\mathbf{M}) > 0 \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

$$m' = \begin{cases} 1, & \text{if sum}(\mathbf{M}) > 0 \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

where $\mathbf{W}$ and $b$ indicate the weights and bias of the convolution filter, respectively. $\mathbf{X}$ are the input pixels for the current convolution window, and $\mathbf{M}$ is the corresponding mask in the receptive field. $\odot$ denotes element-wise multiplication. $x'$ is the output feature value of the partial convolution, and $m'$ is the updated mask value. $\mathbf{1}$ is an all-one matrix with the same shape as $\mathbf{M}$.

*2) Skip Connection and Self-Attention Block:* By leveraging partial convolutional layers, we can extract more features from outside the text region, which are beneficial for background reconstruction. However, in some cases, the feature information related to texture and illumination (such as highlight, shadow, and transparency) inside the text region is also helpful for background inpainting. Thus, considering that the feature maps $F_m$ generated in the SMPM contain rich feature information of the text, we concatenate the feature maps $F_m$ and $F_b$ from the two modules using skip connection, and feed the concatenated feature maps into a self-attention network that learns both the correspondences between feature maps and non-local features. Consequently, the features inside and outside the text regions are split and weighted by a self-attention block for later decoding. Here, we adopt a Global Context (GC) block [54] as the self-attention module, the architecture of which is illustrated in Fig. 4.

*3) Training Loss:* We introduce four loss functions to measure the structural and textural differences between the output image $\hat{I}_{out}$ and the final ground truth $I_{out}$, also considering the spatial smoothness of the inpainted region of $\hat{I}_{out}$ during the training of the background inpainting module, which include pixel reconstruction loss, perceptual loss, style loss, and total variation loss. Details of the loss functions are presented below.

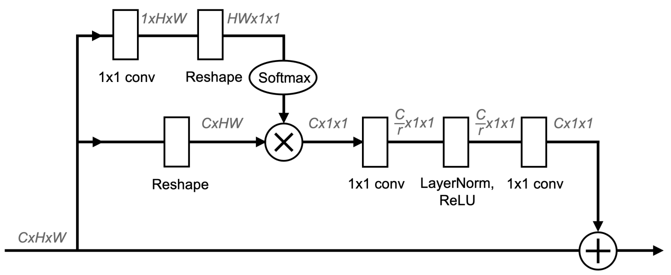


Fig. 4. Architecture of self-attention block: Global Context (GC) block.

Pixel loss is aimed at guiding pixel-level reconstruction, where more weight is added to the inpainted region. The pixel loss can be formulated as follows:

$$\begin{aligned}\mathcal{L}_{pixel} = \ & \|\hat{I}_{mask} \odot (\hat{I}_{out} - I_{out})\|_1 \\ & + 6\|(1 - \hat{I}_{mask}) \odot (\hat{I}_{out} - I_{out})\|_1. \end{aligned} \tag{5}$$

Perceptual loss and style loss, which are also known as VGG loss [55], [56], are used to make the generated image more realistic. Perceptual loss captures high-level semantics and can be considered as a simulation of human perception on images. Perceptual loss computes the differences (L1 loss) between different levels of feature representations between both $\hat{I}_{out}$ and $I_{comp}$ and $I_{out}$, and is extracted by the same pretrained VGG network [57]. $I_{comp}$ is the composed image, where the hole and non-hole regions are from $\hat{I}_{out}$ and $I_{out}$, respectively. Perceptual loss can be defined by Eq. 7:

$$I_{comp} = \hat{I}_{mask} \odot I_{out} + (1 - \hat{I}_{mask}) \odot \hat{I}_{out}. \tag{6}$$

$$\begin{aligned}\mathcal{L}_{per} = \ & \mathbb{E}[\sum_{i=1} \|\phi_i(\hat{I}_{out}) - \phi_i(I_{out})\|_1 \\ & + \sum_{i=1} \|\phi_i(I_{comp}) - \phi_i(I_{out})\|_1], \end{aligned} \tag{7}$$

where $\phi_i$ is the activation map from relu1_1 to the relu5_1 layer of an ImageNet-pretrained VGG-19 model. Style loss penalizes the differences between both $\hat{I}_{out}$ and $I_{comp}$ and $I_{out}$ in the image style, such as color, texture, and pattern. The style loss is defined as follows:

$$\begin{aligned}\mathcal{L}_{style} = \ & \mathbb{E}_i[\|G_i^{\phi}(\hat{I}_{out}) - G_i^{\phi}(I_{out})\|_1 \\ & + \|G_i^{\phi}(I_{comp}) - G_i^{\phi}(I_{out})\|_1], \end{aligned} \tag{8}$$

here, the Gram matrix $G_i^{\phi} = \phi_i \phi_i^T / C_i H_i W_i$ and $C_i H_i W_i$ is the shape of the feature map of $\phi_i$.

The total variation loss is employed to maintain spatial continuity and smoothness in the generated image to reduce the effect of noise.

$$\begin{aligned}\mathcal{L}_{tv} = & \sum_{(i,j,j+1)\in M} \|I_{comp}^{i,j+1} - I_{comp}^{i,j}\|_1 \\ & + \sum_{(i,j,i+1)\in M} \|I_{comp}^{i+1,j} - I_{comp}^{i,j}\|_1, \end{aligned} \tag{9}$$

where $M$ is the hole region in $\hat{I}_{mask}$.

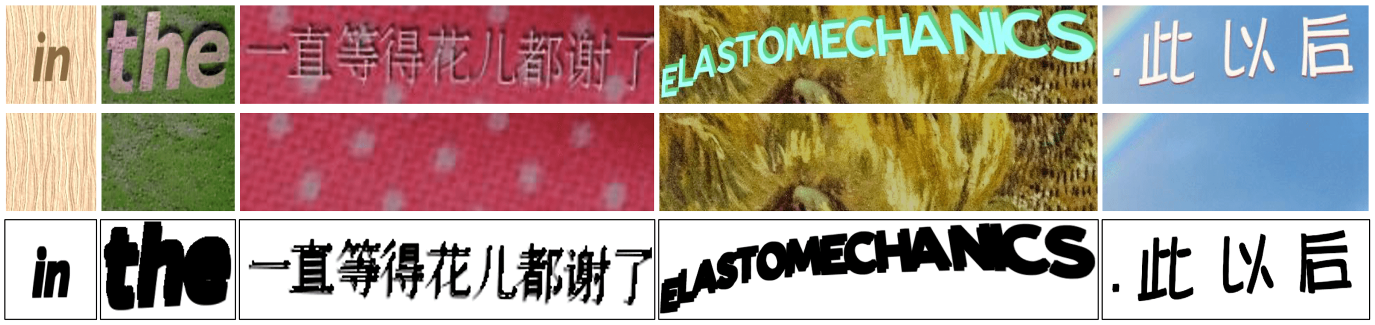


Fig. 5. Some training image samples generated by our enhanced synthesis text engine. Input (top), final ground truth (middle), text mask ground truth (bottom).

Finally, the loss function of the entire network is expressed as follows:

$$\mathcal{L} = 10\mathcal{L}_{SMPM} + \mathcal{L}_{pixel} + \lambda_1\mathcal{L}_{per} + \lambda_2\mathcal{L}_{style} + \lambda_3\mathcal{L}_{tv}, \quad (10)$$

where $\lambda_1$, $\lambda_2$, and $\lambda_3$ are set to 0.05, 100, and 0.1, respectively, according to Liu *et al.* [38]. We slightly changed the weight of the style loss according to our own training loss curve.

## IV. Experiment

### A. Implementation Details

Our implementation is based on PyTorch. In the training process, we generated one million synthetic text images and corresponding text mask images as training data from background images that did not contain text. The input size of our network was 128 × 640, and the height of the training images was first resized to 128 while maintaining the aspect ratio. Then, if the width of the image was insufficient, the remaining pixels were padded with 0 on the right side of the image; otherwise, it was resized to 640. The training batch size was 8 on a single 1080Ti GPU. We used Adam [58] to optimize the entire network with $\beta = (0.9, 0.999)$ and set the weight decay to 0. The learning rate started with 0.0002 and decayed to nine-tenth after each epoch in the training phase. The network was trained in an end-to-end manner, and we followed the fine-tuning strategy [38], which freezes the batch normalization parameters in the encoder of the background inpainting module after approximately 10 epochs.

In the inference process, we first expanded the text bounding box to include more background information and cropped the expanded text region from the image. Then, we resized and padded the cropped text image following the preprocessing strategy of training and fed it to our proposed network for text erasing. Subsequently, we cut off the padding part of the network output and resized the remaining part back to its original size and ratio. Finally, part of the output, which was inside the original bounding box, was pasted back into the source image. The text in an arbitrary quadrilateral annotation was transformed by perspective transformation, and the text in the curved annotation was transformed by thin-plate-spline into rectangular text images before being fed into our network. The network output was transformed back to its original shape and copied to the original position to obtain the final text-erased image.

### B. Dataset and Evaluation Metrics

#### 1) Synthetic Dataset:

- **Improved Synth-text image** We used over 1,500 English and Chinese fonts and 10,000 background images without text to generate a total of one million images for our model training using our enhanced synthesis text engine, which is improved from Synth-text technology [5]. The training dataset contains original background images as the final ground truth, synthetic text in background images as input, and mask images of synthetic text as the ground truth of the text mask.
  Compared with the vanilla synth-text method, we made several improvements to make our generated data share more similarity with real-world data. 1) There is a 50% possibility that the text instance is composed directly on the background instead of being generated by Poisson blending, in which case, some background information behind the text instance would be provided. 2) Some other effects were added to the text instance, such as Gaussian blur to simulate out-of-focus, text shift, text with a 3D structure, and more shadow parameters to make the shadow of the text more realistic, among others. 3) We compressed and saved the image in JPEG format with different compression qualities to handle images of varying quality. 4) A dilation mask of a text instance, including all of its effects, was generated in the mask image to reduce the effect of JPEG artifacts around the text edge. Some samples of the generated images are shown in Fig. 5.
- **SCUT-Syn** [3] was created by Synthesis text engine [5], which contains 8,000 images for training and 800 images for testing. The background images of this dataset were collected from ICDAR 2013 [11] and ICDAR MLT-2017 [59], and the text instances in the background images were manually removed. Most test images were from the training set, and the training and testing sets were generated from the same background images, although the synthesized text instances were different. We evaluated our method using only test images.

#### 2) Real-World Dataset:

- **ICDAR 2013** [11] is a widely used scene text images dataset that includes 229 training images and 223 testing images. All text instances were in English and were

TABLE III

ABLATION STUDY AND QUALITATIVE COMPARISON BETWEEN DIFFERENT CONFIGURATIONS OF OUR PROPOSED NETWORK ON SCUT-SYN AND SCUT-ENSTEXT DATASETS. SMPM: STROKE MASK PREDICTION MODULE, BIPM: BACKGROUND INPAINTING MODULE, PCONV: PARTIAL CONVOLUTIONS, SC: SKIP CONNECTION BETWEEN TWO MODULES, SA: SELF-ATTENTION BLOCK, DICE: DICE LOSS, L1: L1 LOSS

| Method | SCUT-Syn | | | SCUT-EnsText | | |
|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM(%)↑ | MSE↓ | PSNR↑ | SSIM(%)↑ | MSE↓ |
| BIPM (w/o PConv) | 34.78 | 96.67 | 0.00061 | 34.62 | 95.79 | 0.00114 |
| SMPM + BIPM | 37.74 | 97.31 | 0.00034 | 36.11 | 96.31 | 0.00077 |
| SMPM + BIPM + SC | 38.29 | 97.48 | 0.00030 | 36.36 | 96.39 | 0.00070 |
| SMPM + BIPM (w/o PConv) + SC + SA | **38.70** | **97.67** | 0.00025 | 36.49 | 96.39 | 0.00073 |
| SMPM + BIPM + SC + SA w/o Dice | 38.58 | 97.59 | 0.00026 | 36.64 | 96.42 | 0.00067 |
| SMPM + BIPM + SC + SA w/o L1 | 38.60 | 97.61 | 0.00026 | 36.82 | 96.50 | 0.00062 |
| SMPM + BIPM + SC + SA (all) | 38.60 | 97.55 | **0.00024** | **37.08** | **96.54** | **0.00054** |

well-focused. In this study, only the test set was used for the evaluation.

- **SCUT-EnsText** [4] is a comprehensive and challenging scene text removal dataset, containing 2,749 training images and 813 testing images, which are collected from ICDAR2013 [11], ICDAR-2015 [60], MS COCO-Text [61], SVT [62], MLT-2019 [63], and ArTs [64]. The text instances of this dataset are in Chinese or English with diverse shapes, such as horizontal text, arbitrary quadrilateral text, and curved text. All text instances were carefully erased by annotators with good visual quality. By providing original text annotation and text-erased ground-truth, this dataset can be comprehensively used for both qualitative and quantitative evaluations. In this study, we used this test set to evaluate the performance of the proposed method.

*3) Evaluation Metrics:*

- **Quantitative Evaluation** To quantify the text erasure ability of a model, we followed [2]–[4], [7], [8] and utilized a baseline scene text detection model to detect the texts in the text-erased images and evaluated how low are the recall of the detection results. A lower recall indicated that less text was detected, and more text was erased by the model. To make a fair comparison with previous studies, scene text detector EAST and ICDAR 2013 evaluation [22] protocols were used in the ICDAR 2013 dataset [11], text detector CRAFT [30], and ICDAR 2015 [60] protocols were adopted in the evaluation of SCUT-EnsText [4].
- **Qualitative Evaluation** We followed the previous image inpainting works by reporting metrics including peak signal-to-noise ratio (PSNR), the structural similarity index (SSIM) [65] and mean squared error (MSE). Higher SSIM, PSNR, and lower MSE values indicate better image restoration quality. Qualitative evaluations were conducted on both the SCUT-Syn [3] and SCUT-EnsText [4] datasets.

## C. Ablation Study

In this section, we investigate the effectiveness of the different settings of the proposed model. The stroke mask prediction module (SMPM), skip connection (SC) between two modules, self-attention block (SA), partial convolutions (PConv), L1 loss, and dice loss are the focus of this study. The qualitative evaluation results on the SCUT-Syn and SCUT-EnsText datasets are presented in Table III, and some text-erasing samples are shown in Fig. 6.

- **Stroke Mask Prediction Module** SMPM aims to provide the pixel-level information of text region as the hole for background inpainting Module (BIPM), so that the network can learn more from the valid features of the non-text region and suppress the text residue. The qualitative results are presented in Table III. Using the text mask can significantly improve the text erasing performance. It should be noted that the partial convolutional layers in the background inpainting module, function as normal convolutional layers without the mask image from the SMPM.
- **Skip Connection** Skip connection links and concatenates the low-resolution feature maps of the two modules to provide the features inside the text region for the decoder of the BIPM and to improve the accuracy and stability of text mask prediction. Table III implies that the skip connection between the two modules can improve the erasing quality of the image in both the SCUT-Syn and SCUT-EnsText datasets.
- **Self-Attention Block** GCblock adds channel-wise weights to the input feature maps taking into consideration the correspondences between feature maps and non-local features. To confirm the importance of the self-attention block, we trained our network without the SA block. In Table III, we see that the performance of our network decreases when the self-attention block is missing.
- **Partial Convolutions** To evaluate the advantages of the partial convolutional layers, we also re-implemented our method without these layers. Table III lists the qualitative performance of the SCUT-Syn and SCUT-EnsText datasets. We observed that, compared to a network with partial convolutional layers, the network with partial convolutional layers performs better on SCUT-EnsText but worse on SCUT-Syn datasets. We believe that the discrepancies between synthetic data and real-world data are the cause for this difference in performance. As mentioned before, the reason for improving the Synth-text engine is that Poisson image editing retains some texture information of the background image when it blends the foreground text instances into a background image. However, most scene text instances in real-world images are not transparent. For the erasure result on SCUT-Syn

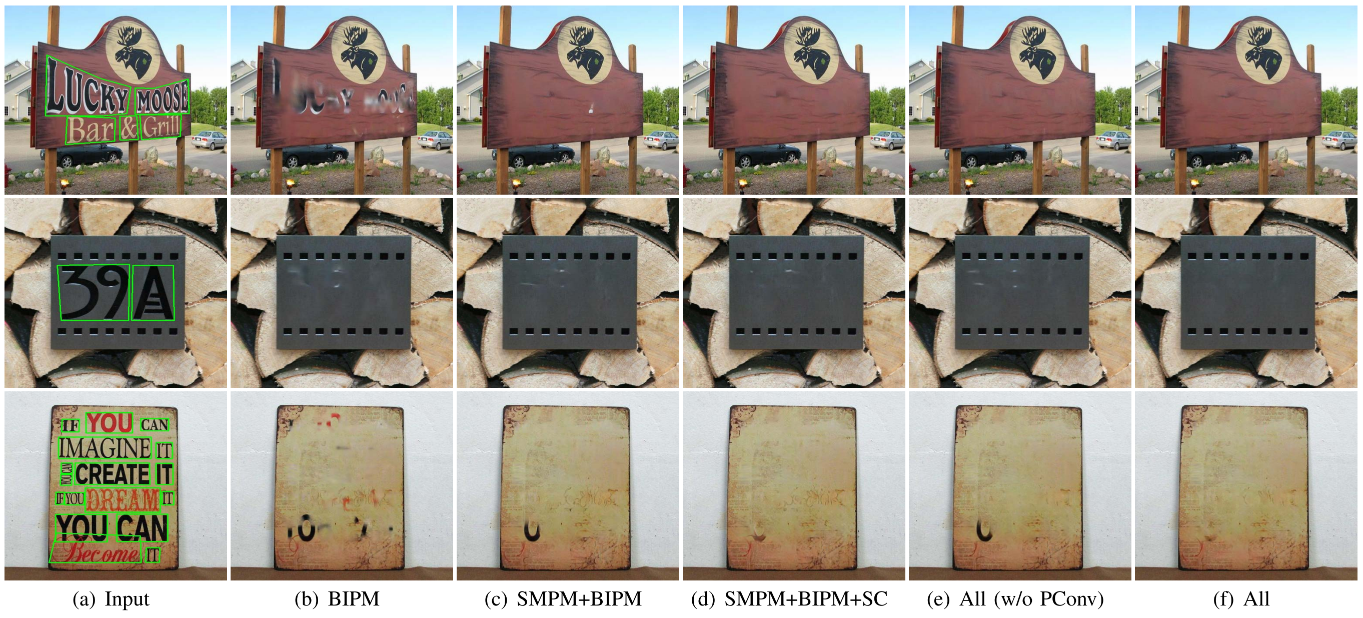


Fig. 6. Visual quality results of ablation study on SCUT-EnsText dataset. From left to right: input images, output of BIPM, output of SMPM+BIPM, output of SMPM+BIPM+SC, output of all (w/o PConv), and output of all. SMPM: Stroke mask prediction module, BIPM: Background inpainting module, PConv: Partial convolutions, SC: Skip connection between two modules, SA: Self-attention block, All: SMPM + BIPM + SC + SA.

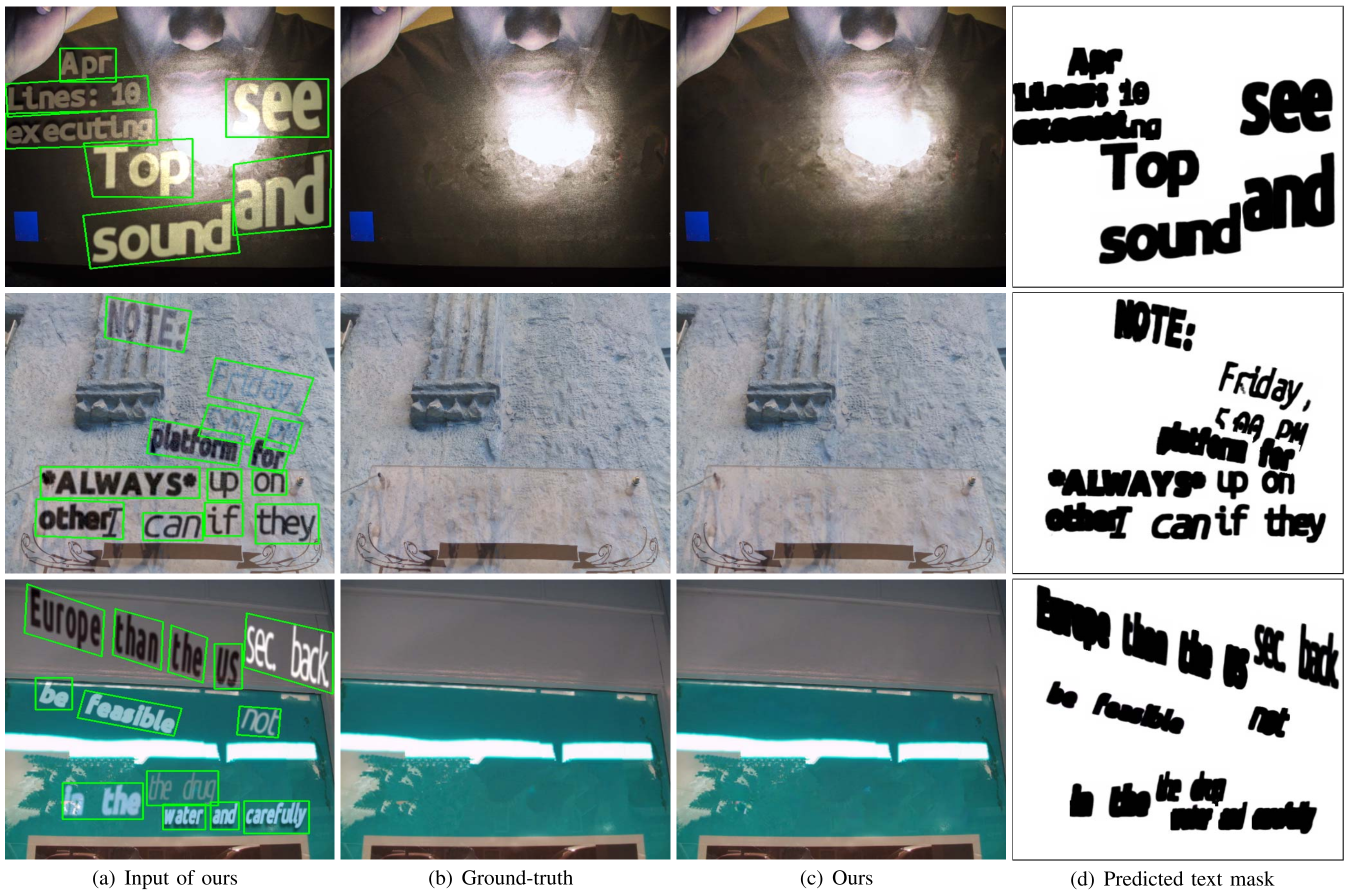
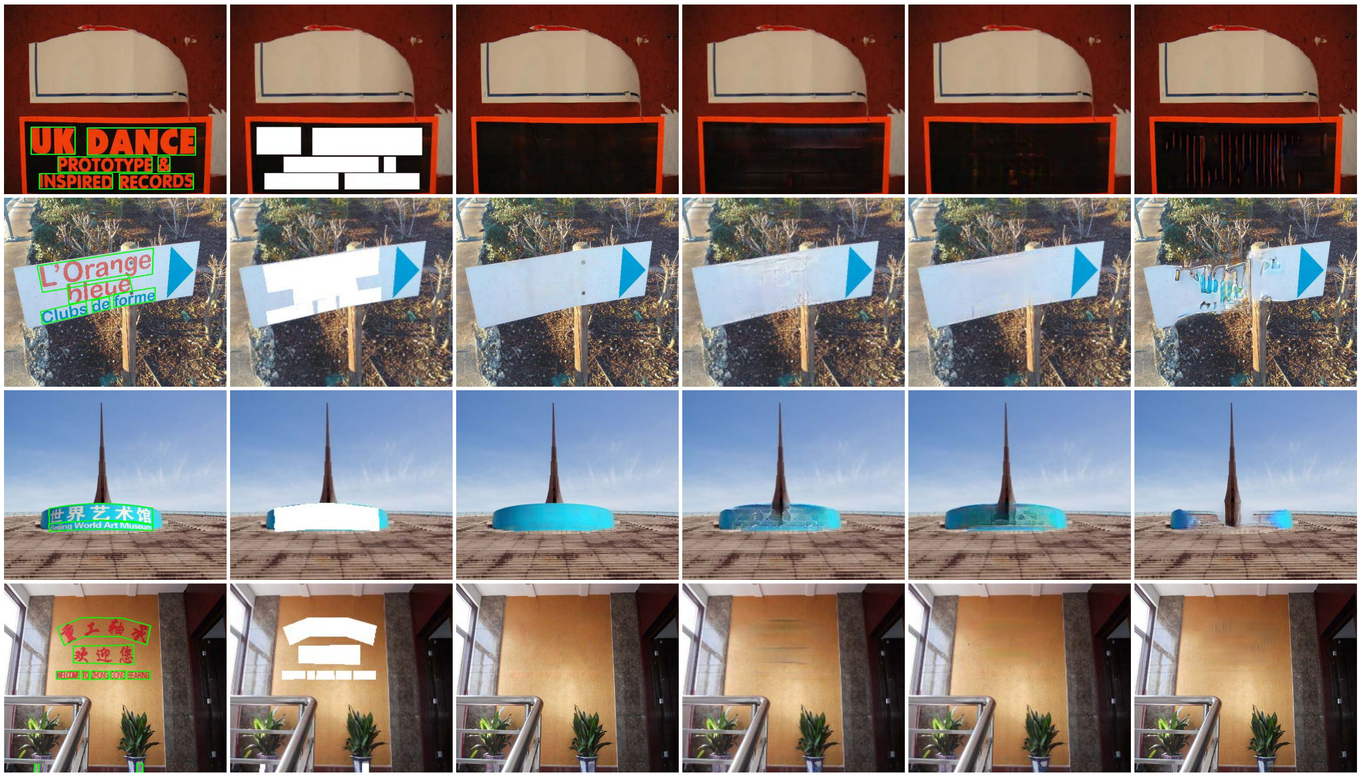


Fig. 7. Qualitative results of our method on the SCUT-Syn dataset. From left to right: input image and text bounding boxes, ground truth, output of our method, and predicted text mask.

datasets, compared with directly extracting features using normal convolution, using partial convolution and $F_m$ feature concatenation to split the features inside and outside the text region is a relatively inefficient approach for transparent text erasure. However, when facing the real-world data such as SCUT-EnsText, using partial



(a) Input of ours (b) Input of inpainting methods (c) Ours (d) LBAM (e) RFR-Net (f) HiFill

Fig. 8. Visual qualitative comparison between our method and state-of-the-art image inpainting methods on the SCUT-EnsText dataset. From left to right: input of our method, input of inpainting methods, output of our method, output of LBAM, output of RFR-Net, and output of HiFiII.

TABLE IV
COMPARISON BETWEEN PREVIOUS SCENE TEXT-ERASING STUDIES AND OUR PROPOSED METHOD ON THE SCUT-SYN AND ICDAR2013 DATASETS

| Method | SCUT-Syn | | | ICDAR2013 | Parameters | Inference speed | Input |
|---|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM(%)↑ | MSE↓ | R↓ | | | |
| Original images | - | - | - | 70.83 | - | - | - |
| SceneTextEraser [1] | 14.68 | 46.13 | 0.7148 | 10.08 | - | - | Image |
| Pix2Pix [49] | 25.60 | 89.86 | 0.2465 | 10.19 | 54.4M | **17ms** | Image |
| EnsNet [3] | 37.36 | 96.44 | 0.0021 | 5.66 | 12.4M | 24ms | Image |
| MTRNet [7] | 29.71 | 94.43 | 0.0001 | 0.18 | 54.4M | - | Image(256×256) + Text Mask |
| Weak Supervision [8] | 37.44 | 93.69 | - | 2.47 | 28.7+6.0M | 57+39ms | Image(256×256) |
| MTRNet++ [2] | 34.55 | **98.45** | 0.0004 | - | 18.7M | 37ms | Image(256×256) |
| EraseNet [4] | 38.32 | 97.67 | **0.0002** | - | 19.7M | 34ms | Image |
| EAST [22] + Ours | 31.18 | 95.93 | 0.002 | 0.73 | 24.1+9.9M | 18+23∼ms | Image |
| Ours | **38.60** | 97.55 | **0.0002** | **0** | **9.9M** | 23∼ms | Image + BBox |

convolution can achieve better performance than normal convolution.

- **L1 Loss and Dice Loss** To confirm the contribution of these two losses in SMPM, we trained our whole network with single L1 loss or dice loss and compared the final erasure result with all. In Table III, we observe that our network with the L1 loss and dice loss combination achieved better results on the real-world dataset.

## D. Comparison With State-of-the-Art Methods

To evaluate the performance of our proposed method, we compared it with recent state-of-the-art methods on the SCUT-Syn, ICDAR2013, and SCUT-EnsText datasets. For the SCUT-Syn and ICDAR2013 datasets, the results of SceneTextEraser, Pix2Pix, and EnsNet were implemented and reported by Zhang *et al.* [3]. The results of MTRNet [7], weak supervision [8], MTRNet++ [2], and EraseNet [4] were collected from official reports. If there is no specific description, the resolution of the input image is 512 × 512. Table IV displays the results for the SCUT-Syn [3] and ICDAR2013 [11] datasets. Our proposed method achieves the highest PSNR on the SCUT-Syn dataset and the lowest recall on the ICDAR2013 dataset when the bounding boxes are provided. Some text-erasing examples on the SCUT-Syn dataset are shown in Fig. 7. Our method achieves lower recall on the ICDAR2013 dataset when the detection result of EAST [22] is used to provide text location. Here, R represents recall, which is the detection result of the EAST under the ICDAR2013 evaluation protocol. We believe the reason why MTRNet++, EraseNet, and EnsNet could generate higher SSIM images on

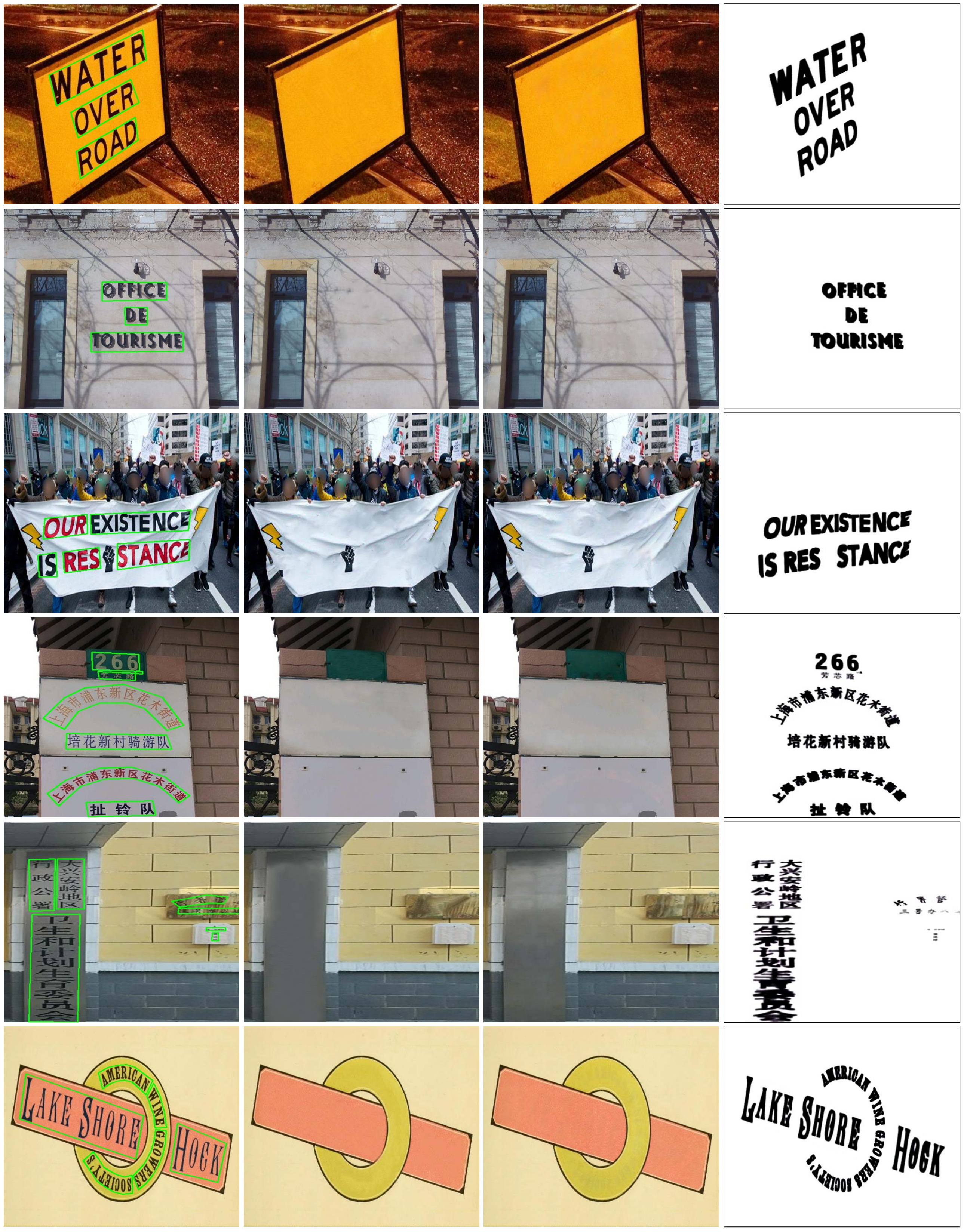


Fig. 9. Qualitative results of our method on SCUT-EnsText dataset. From left to right: input, ground truth, output of our method, and predicted text mask.

the SCUT-Syn dataset is because most test images are included in the training set, and share the same background images with training images when they are generated. The EAST and our model were not trained on the SCUT-Syn training set,

TABLE V
COMPARISON BETWEEN STATE-OF-THE-ART INPAINTING METHODS AND PROPOSED METHOD ON THE SCUT-ENSTEXT DATASET

| Method | SCUT-EnsText | | | Parameters | Inference speed | Input | Training dataset |
|---|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM(%)↑ | MSE↓ | | | | |
| LBAM [39] | 36.21 | 95.58 | 0.0007 | 68.3M | **11ms** | Image(256×256) + Text Mask | Paris Street View [9] |
| RFR-Net [40] | 36.95 | 96.12 | 0.0006 | 31.2M | 90ms | Image(256×256) + Text Mask | Paris Street View [9] |
| HiFill [43] | 31.48 | 94.17 | 0.0021 | **2.7M** | 28ms | Image + Text Mask | Places2 [66] |
| Ours | **37.89** | **97.02** | **0.0004** | 9.9M | 23∼ms | Image(256×256) + BBox | Improved Synth text |
| Ours | **37.08** | **96.54** | **0.0005** | 9.9M | 23∼ms | Image + BBox | Improved Synth text |

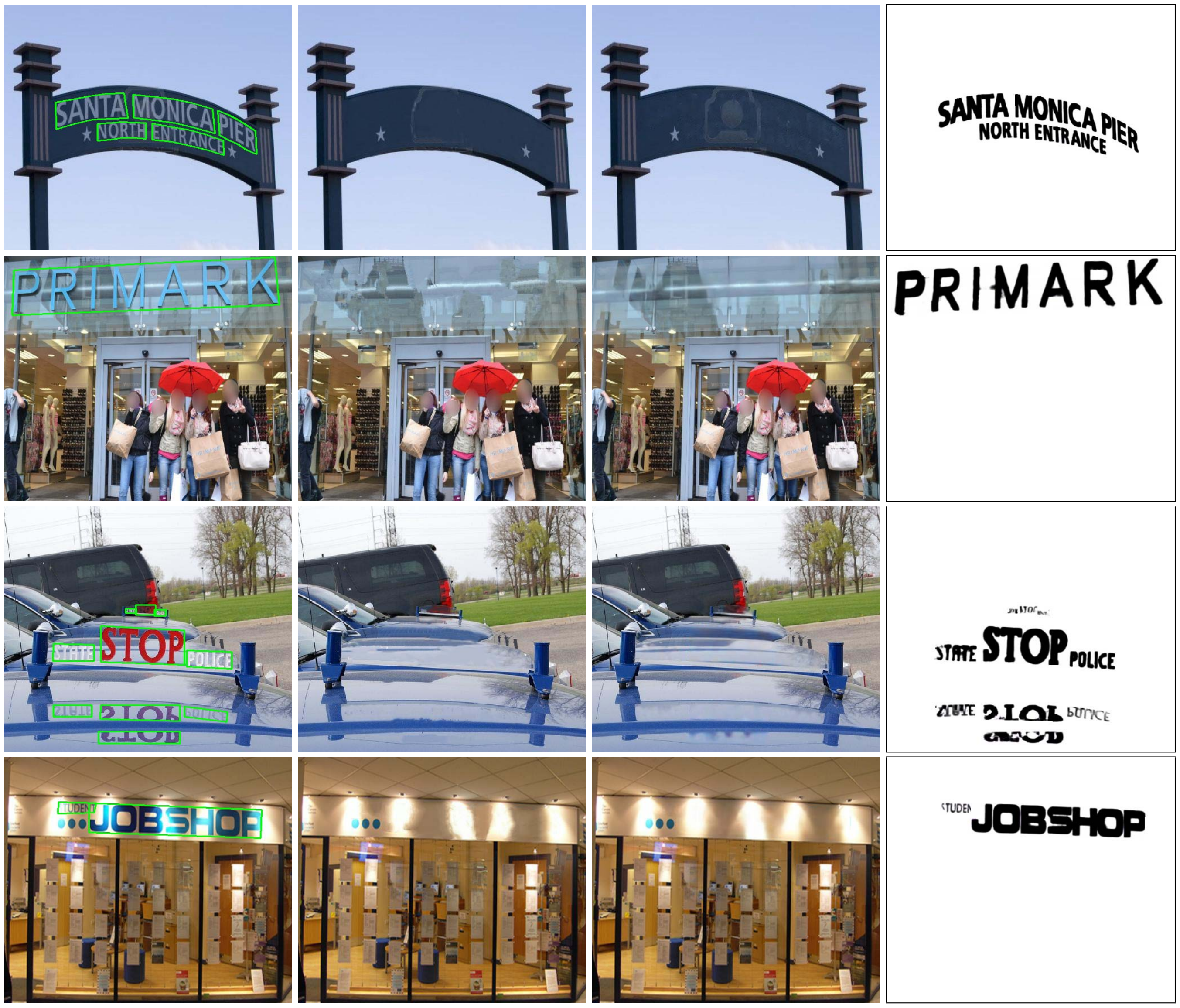
TABLE VI
COMPARISON BETWEEN STATE-OF-THE-ART SCENE TEXT-ERASING METHODS AND PROPOSED METHOD ON THE SCUT-ENSTEXT DATASET. RE: WE RE-IMPLEMENTED THIS METHOD

| Method | Qualitative eval | | | Quantitative eval | Parameters | Inference speed | Input |
|---|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM(%)↑ | MSE↓ | R↓ | | | |
| Original images | - | - | - | 69.5 | - | - | - |
| SceneTextEraser [1] | 25.47 | 90.14 | 0.0047 | 5.9 | - | - | Image |
| EnsNet [3] | 29.54 | 92.74 | 0.0024 | 32.8 | **12.4M** | **24ms** | Image |
| EraseNet [4] | 32.30 | 95.42 | 0.0015 | 4.6 | 19.7M | 34ms | Image |
| EraseNet [4] (Re) | 32.64 | 95.13 | 0.0017 | 5.4 | 19.7M | 34ms | Image |
| DB-ResNet-50 [29] + EraseNet (Re) | 32.70 | 95.18 | 0.0021 | 12.3 | 26.1+19.7M | 37+34ms | Image |
| CRAFT [30] + EraseNet (Re) | 34.15 | 95.66 | 0.0015 | 7.3 | 20.7+19.7M | 120+34ms | Image |
| DB-ResNet-18 [29] + Ours | 33.17 | 95.44 | 0.0020 | 10.3 | 12.6+9.9M | 17+27∼ms | Image |
| DB-ResNet-50 [29] + Ours | 33.54 | 95.57 | 0.0018 | 10.5 | 26.1+9.9M | 37+27∼ms | Image |
| CRAFT [30] + Ours | **35.34** | **96.24** | **0.0009** | **3.6** | 20.7+9.9M | 120+27∼ms | Image |

thereby resulting in a lower PSNR and SSIM. We believe that this dataset cannot fully reflect the generalization ability of a network when it is used for training. Our text-erasing network is lightweight with only 9.9 million trainable parameters. For a fair comparison of the inference speed, we tested all methods using a single 1080ti GPU and an AMD Ryzen7 3700X @ 3.6GHz CPU with the original input size of the networks. The inference time of our method consisted of the time cost of the network forward, pre-processing, and post-processing. The time cost of pre-processing and post-processing was approximately 4 ms in the case of using perspective transformation and 76 ms in the case of using off-the-shell thin-plate-spline function in OpenCV.

For the SCUT-EnsText dataset, we compared our method with state-of-the-art image inpainting methods and scene text erasing methods. The results of the comparison with the inpainting methods of the SCUT-EnsText dataset are shown in Table V. Our scene text erasing method achieves excellent results in image quality when text bounding box information is provided. We made some revisions to the original text location annotation because we observed some unmatched cases between the location of the erased text and the text bounding box of the ground truth. We selected three state-of-the-art image inpainting methods: LBAM [39], RFR-Net [40], and HiFill [43]. The LBAM [39] and RFR-Net [40] models were pretrained on the Paris Street View dataset [9], and the HiFill [43] model was pretrained on the Places2 dataset [66]. For a fair comparison with the pretrained inpainting methods, we generated the hole mask directly from our revised text location annotations and resized the input images to the same size because some pretrained inpainting models only work at a resolution of $256 \times 256$. Our method generated images with higher quality than that of the state-of-the-art image inpainting methods under the same conditions. Some samples are presented for visual quality comparison in Fig. 8. We observed that there are certain unusual textures or artifacts in the text-erased images inpainted via pretrained image inpainting methods, causing unnatural erasure results. We also found the inpainting logic of HiFill [43], a preference for restoring the hole region with further background information is displayed, leading to worse results than those obtained using the other two methods. Although this model is pretrained on the Places2 [66] dataset, which contains many indoor and urban views, it still presents a serious domain shift problem when facing scene-text-erasing tasks.

In addition, we used our method with a pretrained scene text detector as a two-step automatic scene text eraser. In our experiment, we used DB [29] and CRAFT [30] for scene text detection and produced arbitrary quadrilateral bounding boxes as the input for our method. DB-ResNet-18 and DB-ResNet-50 were pretrained in the SynthText and ICDAR 2015 datasets with the box threshold set at 0.3. CRAFT was pretrained on the SynthText, ICDAR 2013, and MLT 2017 datasets, and the text threshold was set to 0.6. We compared our proposed methods with previous scene text erasing methods on the SCUT-EnsText dataset. For a fair comparison with one-step methods, such as EraseNet, we carefully re-implemented EraseNet and replaced the background regions of the output of EraseNet with the original images according to the detection results of the scene text detector. The results are shown in Table VI and imply that the results of the first detection followed by inpainting via our method are significantly superior to those of existing state-of-the-art one-step methods in both qualitative and quantitative evaluations. Here, R denotes the recall, which is the detection result of the CRAFT [30] under the ICDAR2015 evaluation [60] protocol. Overall, compared with a single EraseNet, using an auxiliary scene text detector with the one-step method could help increase the quality of



(a) Input (b) Ground-truth (c) Ours (d) Predicted text mask

Fig. 10. Our method can retain more detailed background information and restore the background texture. From left to right: input image and text bounding boxes, ground truth, output of our method, and predicted text mask.

the output images by replacing excessive erasure of text-free areas with content from original images. On the other hand, this operation could also restore some text-erased instances to their original status because of the limited detection ability of the text detector, resulting in an increase in recall. Some qualitative results of our method on the SCUT-EnsText dataset are shown in Fig. 9. Our method can clearly remove the text region, regardless of whether the text instances have varying shapes, fonts, or illumination conditions. Nonetheless, compared with one-step end-to-end scene text erasing methods, the inference speed of our two-step cropped-images-based pipeline is relatively slow, as shown in Table VI.

## *E. Discussion*

From the experimental results, we observe that the model trained using our improved synthetic text dataset displays a different inpainting logic than that of the annotations of the real-world scene-text removal dataset, which was manually edited using Photoshop. As shown in Fig. 10, our method can erase text while retaining more detailed background information of the image. In addition, owing to the model design and a large amount of training data, our method was able to reconstruct some background texture for better visual perception. However, because of the limited text style of the synthetic text engine, the method failed during text erasing in case of text oriented in special shape or the stereoscopic text under complicated illumination conditions, as depicted in Fig. 11. Because we propose erasing word-level text in cropped images, our method encountered some difficulty in erasing small text instances in the images. As there are always JPEG boundary artifacts surrounding the text edge, our model cannot significantly discriminate whether the pixel near the

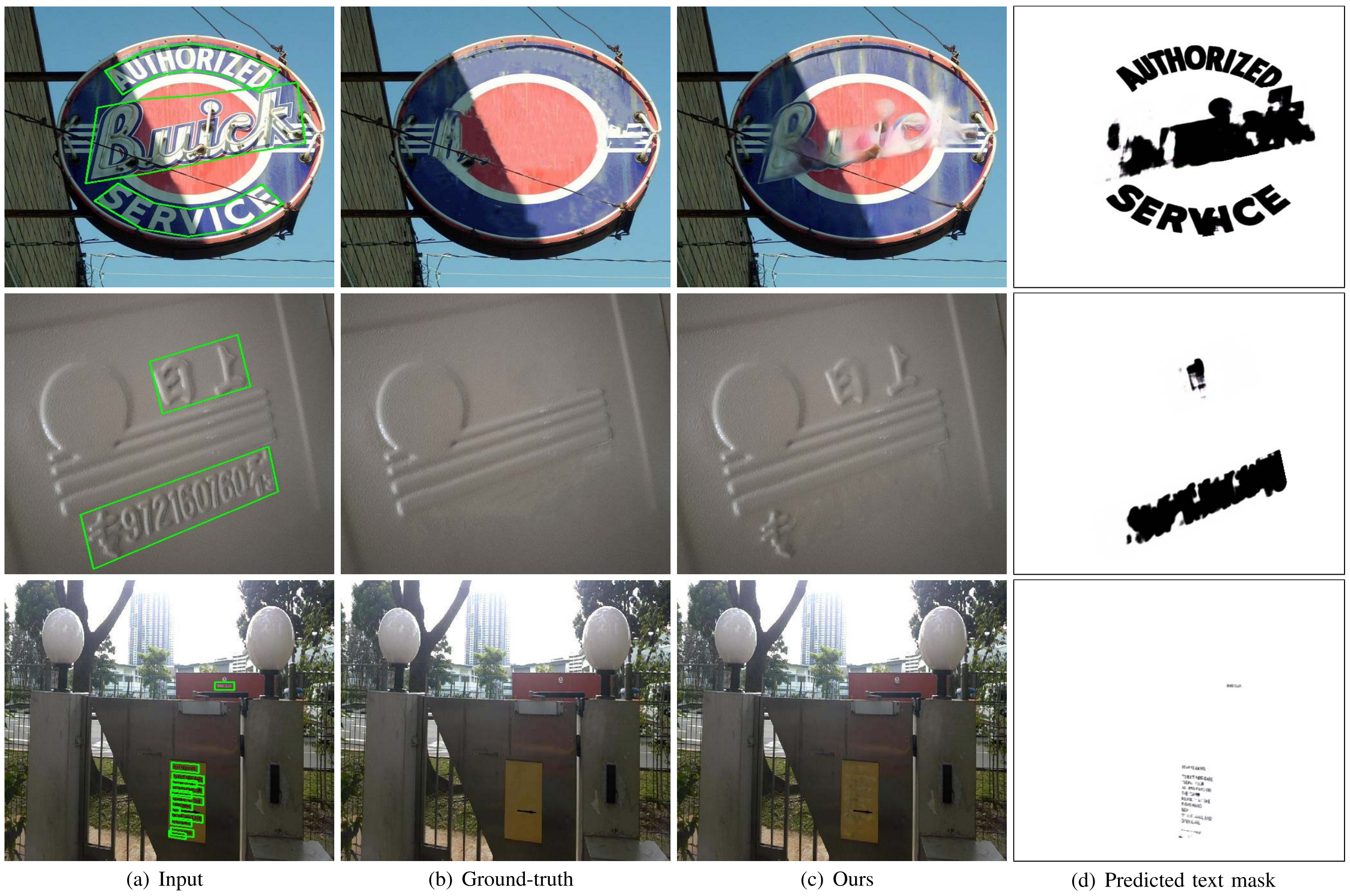


Fig. 11. Some failure cases of our method. From left to right: input image and text bounding boxes, ground truth, output of our method, and predicted text mask.

text edge belongs to the background or to the artifact. Borrowing features from artifact regions can result in a text region being inpainted by strange colors, yielding poor inpainting results.

When using scene text detector working with our method, the quality of the generated text bounding boxes also affects the quality of text erasure. Text instances partially covered by bounding boxes can result in residues of text outside boxes, because in our experimental setting, only the region inside the text bounding boxes are replaced by the output of our network. Moreover, curved text, bounded by poor quality polygons, will be heavily distorted after TPS transformation, which makes the prediction of stroke mask more difficult, resulting in poor text erasure. In contrast, curved text, with good polygon annotation, usually leads to high-quality erasing. In our evaluation, we found that using perspective transformation and quadrilateral bounding boxes is the safest way to maintain high-quality erasure on average.

## V. Conclusion

In this study, we propose a novel scene text erasing method that addresses the weak text location problem of one-step methods and the domain shift problem of using inpainting models pretrained on street view or Places datasets. To this end, our model was trained using only our improved synthetic text dataset. The model inpaints the text region based on a predicted text stroke mask derived from cropped text images, whereby more background information can be preserved. By utilizing a stroke mask prediction module, partial convolution layers, an attention block in the background inpainting module, and skip connection between two modules, our method can reasonably erase scene text with texture restoration. Using a pretrained scene text detector to provide text location information, our model can function as an automatic scene text eraser to remove text from the wild. Overall, our experimental results show that our method performs better than existing state-of-the-art methods.

## References

[1] T. Nakamura, A. Zhu, K. Yanai, and S. Uchida, "Scene text eraser," in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Nov. 2017, pp. 832–837.
[2] O. Tursun, S. Denman, R. Zeng, S. Sivapalan, S. Sridharan, and C. Fookes, "MTRNet++: One-stage mask-based scene text eraser," *Comput. Vis. Image Understand.*, vol. 201, Dec. 2020, Art. no. 103066.
[3] S. Zhang, Y. Liu, L. Jin, Y. Huang, and S. Lai, "EnsNet: Ensconce text in the wild," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 801–808.
[4] C. Liu, Y. Liu, L. Jin, S. Zhang, C. Luo, and Y. Wang, "EraseNet: End-to-end text removal in the wild," *IEEE Trans. Image Process.*, vol. 29, pp. 8760–8775, 2020.
[5] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2315–2324.

[6] X. Bian, C. Wang, W. Quan, J. Ye, X. Zhang, and D.-M. Yan, "Scene text removal via cascaded text stroke detection and erasing," 2020, *arXiv:2011.09768*.

[7] O. Tursun, R. Zeng, S. Denman, S. Sivapalan, S. Sridharan, and C. Fookes, "MTRNet: A generic scene text eraser," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 39–44.

[8] J. Zdenek and H. Nakayama, "Erasing scene text with weak supervision," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2020, pp. 2227–2235.

[9] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. Efros, "What makes Paris look like Paris?" *ACM Trans. Graph.*, vol. 31, no. 4, pp. 1–9, 2012.

[10] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[11] D. Karatzas *et al.*, "ICDAR 2013 robust reading competition," in *Proc. 12th Int. Conf. Document Anal. Recognit.*, Aug. 2013, pp. 1484–1493.

[12] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jun. 2004, pp. 366–373.

[13] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images," in *Proc. Asian Conf. Comput. Vis.*, vol. 6494, 2011, pp. 770–783.

[14] A. Jamil, I. Siddiqi, F. Arif, and A. Raza, "Edge-based features for localization of artificial Urdu text in video images," in *Proc. Int. Conf. Document Anal. Recognit.*, Sep. 2011, pp. 1120–1124.

[15] A. Mosleh, N. Bouguila, and A. B. Hamza, "Automatic inpainting scheme for video text detection and removal," *IEEE Trans. Image Process.*, vol. 22, no. 11, pp. 4460–4472, Nov. 2013.

[16] W. Huang, Z. Lin, J. Yang, and J. Wang, "Text localization in natural images using stroke feature transform and text covariance descriptors," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1241–1248.

[17] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "TextBoxes: A fast text detector with a single deep neural network," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 4161–4167.

[18] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, vol. 9905, 2016, pp. 21–37.

[19] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *Proc. Eur. Conf. Comput. Vis.*, vol. 9912, 2016, pp. 56–72.

[20] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[21] J. Ma *et al.*, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Trans. Multimedia*, vol. 20, no. 11, pp. 3111–3122, Nov. 2018.

[22] X. Zhou *et al.*, "EAST: An efficient and accurate scene text detector," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2642–2651.

[23] C. Zhang *et al.*, "Look more than once: An accurate detector for text of arbitrary shapes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10544–10553.

[24] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, "Multi-oriented text detection with fully convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4159–4167.

[25] P. Lyu, M. Liao, C. Yao, W. Wu, and X. Bai, "Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes," in *Proc. Eur. Conf. Comput. Vis.*, vol. 11218, Sep. 2018, pp. 71–88.

[26] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," in *IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[27] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao, "TextSnake: A flexible representation for detecting text of arbitrary shapes," in *Proc. Eur. Conf. Comput. Vis.*, vol. 11206, 2018, pp. 19–35.

[28] Y. Li *et al.*, "PSENet: Psoriasis severity evaluation network," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 1, pp. 800–807.

[29] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, "Real-time scene text detection with differentiable binarization," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 7, pp. 11474–11481.

[30] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9357–9366.

[31] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proc. 28th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*, 2001, pp. 341–346.

[32] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patch-Match," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 1–11, 2009.

[33] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen, "Image melding: Combining inconsistent images using patch-based synthesis," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 1–10, Aug. 2012.

[34] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. ACM SIGGRAPH Conf. Comput. Graph.*, 2000, pp. 417–424.

[35] M. M. Oliveira, B. Bowen, R. McKenna, and Y.-S. Chang, "Fast digital image inpainting," in *Proc. Int. Conf. Vis. Imag. Image Process.*, 2001, pp. 261–266.

[36] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2536–2544.

[37] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–14, 2017.

[38] G. Liu, A. F. Reda, J. K. Shih, T. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proc. ECCV*, vol. 11215, Sep. 2018, pp. 89–105.

[39] C. Xie *et al.*, "Image inpainting with learnable bidirectional attention maps," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8857–8866.

[40] J. Li, N. Wang, L. Zhang, B. Du, and D. Tao, "Recurrent feature reasoning for image inpainting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7757–7765.

[41] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5505–5514.

[42] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. Huang, "Free-form image inpainting with gated convolution," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4470–4479.

[43] Z. Yi, Q. Tang, S. Azizi, D. Jang, and Z. Xu, "Contextual residual aggregation for ultra high-resolution image inpainting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7505–7514.

[44] C. W. Lee, K. Jung, and H. J. Kim, "Automatic text detection and removal in video sequences," *Pattern Recognit. Lett.*, vol. 24, no. 15, pp. 2607–2623, Nov. 2003.

[45] E. A. Pnevmatikakis and P. Maragos, "An inpainting system for automatic image structure—Texture restoration with text removal," in *Proc. 15th IEEE Int. Conf. Image Process.*, Oct. 2008, pp. 2616–2619.

[46] A. Mosleh, N. Bouguila, and A. B. Hamza, "Image text detection using a bandlet-based edge detector and stroke width transform," in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 63.1–63.12.

[47] M. Khodadadi and A. Behrad, "Text localization, extraction and inpainting in color images," in *Proc. 20th Iranian Conf. Electr. Eng. (ICEE)*, May 2012, pp. 1035–1040.

[48] P. D. Wagh and D. R. Patil, "Text detection and removal from image using inpainting with smoothing," in *Proc. Int. Conf. Pervas. Comput. (ICPC)*, Jan. 2015, pp. 1–4.

[49] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5967–5976.

[50] S. Qin, J. Wei, and R. Manduchi, "Automatic semantic content removal by learning to neglect," in *Proc. Brit. Mach. Vis. Conf.*, 2018, pp. 1–12.

[51] C. Zheng, T.-J. Cham, and J. Cai, "Pluralistic image completion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1438–1447.

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[53] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-Net: Fully convolutional neural networks for volumetric medical image segmentation," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 565–571.

[54] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "GCNet: Non-local networks meet squeeze-excitation networks and beyond," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 1971–1980.

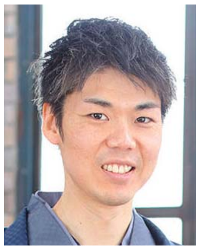
[55] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proc. Eur. Conf. Comput. Vis.*, vol. 9906, 2016, pp. 694–711.

[56] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2414–2423.

[57] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015, pp. 1–14.

[58] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015, pp. 1–15.
[59] N. Nayef *et al.*, "ICDAR2017 robust reading challenge on multi-lingual scene text detection and script identification—RRC-MLT," in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Nov. 2017, pp. 1454–1459.
[60] D. Karatzas *et al.*, "ICDAR 2015 competition on robust reading," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 1156–1160.
[61] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie, "COCO-text: Dataset and benchmark for text detection and recognition in natural images," 2016, *arXiv:1601.07140*.
[62] K. Wang and S. Belongie, "Word spotting in the wild," in *Proc. Eur. Conf. Comput. Vis.*, vol. 6311, 2010, pp. 591–604.
[63] N. Nayef *et al.*, "ICDAR2019 robust reading challenge on multi-lingual scene text detection and recognition—RRC-MLT-2019," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 1582–1587.
[64] C. K. Chng *et al.*, "ICDAR2019 robust reading challenge on arbitrary-shaped text-RRC-ArT," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 1571–1576.
[65] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[66] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1452–1464, Jun. 2018.

**Zhengmi Tang** received the B.E. degree from Xidian University, Shaanxi, China, in 2017, and the M.E. degree in cybernetics engineering from Hiroshima University, Japan, in 2020. He is currently pursuing the Ph.D. degree in communication engineering with the IIC-Laboratory, Tohoku University, Japan. His current research interests include computer vision, scene-text detection, and data synthesis.



**Tomo Miyazaki** (Member, IEEE) received the B.E. degree from Yamagata University in 2006 and the Ph.D. degree from Tohoku University in 2011. From 2011 to 2012, he worked on the geographic information system at Hitachi, Ltd. From 2013 to 2014, he worked at Tohoku University as a Postdoctoral Researcher, where he has been an Assistant Professor since 2015. His research interests include pattern recognition and image processing.

**Yoshihiro Sugaya** (Member, IEEE) received the B.E., M.E., and Ph.D. degrees from Tohoku University, Sendai, Japan, in 1995, 1997, and 2002, respectively. He is currently an Associate Professor at the Graduate School of Engineering, Tohoku University. His research interests include computer vision, pattern recognition, image processing, parallel processing, and distributed computing. He is a member of the Institute of Electronics, Information and Communication Engineers (IEICE) and the Information Processing Society of Japan.

**Shinichiro Omachi** (Senior Member, IEEE) received the B.E., M.E., and Ph.D. degrees in information engineering from Tohoku University, Japan, in 1988, 1990, and 1993, respectively. He worked as an Assistant Professor at the Education Center for Information Processing, Tohoku University, from 1993 to 1996. Since 1996, he has been affiliated with the Graduate School of Engineering, Tohoku University, where he is currently a Professor. From 2000 to 2001, he was a Visiting Associate Professor at Brown University. His research interests include pattern recognition, computer vision, image processing, image coding, and parallel processing. He is a member of the Institute of Electronics, Information and Communication Engineers, the Information Processing Society of Japan, among others. He received the IAPR/ICDAR Best Paper Award in 2007, the Best Paper Method Award of the 33rd Annual Conference of the GfKl in 2010, the ICFHR Best Paper Award in 2010, and the IEICE Best Paper Award in 2012. He is currently the Vice Chair of the IEEE Sendai Section. He served as the Editor-in-Chief for *IEICE Transactions on Information and Systems* from 2013 to 2015.