Computational Visual Media
https://doi.org/10.1007/s41095-021-0242-8 Vol. 8, No. 2, June 2022, 273–287

Research Article

Scene text removal via cascaded text stroke detection and erasing

Xuewei Bian[1,2,*], Chaoqun Wang[1,2,*], Weize Quan[1,2] (✉), Juntao Ye[1,2], Xiaopeng Zhang[1,2], and Dong-Ming Yan[1,2]

Abstract Recent learning-based approaches show promising performance improvement for the scene text removal task but usually leave several remnants of text and provide visually unpleasant results. In this work, a novel end-to-end framework is proposed based on accurate text stroke detection. Specifically, the text removal problem is decoupled into text stroke detection and stroke removal; we design separate networks to solve these two subproblems, the latter being a generative network. These two networks are combined as a processing unit, which is cascaded to obtain our final model for text removal. Experimental results demonstrate that the proposed method substantially outperforms the state-of-the-art for locating and erasing scene text. A new large-scale real-world dataset with 12,120 images has been constructed and is being made available to facilitate research, as current publicly available datasets are mainly synthetic so cannot properly measure the performance of different methods.

Keywords scene text removal; text stroke detection; generative adversarial networks; cascaded network design; real-world dataset

1 Introduction

Text is an important information carrier which often appears in various scenes. The problem of scene text removal can be stated as follows: given an image with a certain amount of text (as in Fig. 1(a)), the goal is to remove the text in this image (see Fig. 1(d)). This task has many applications in daily life, for example in personal information protection (e.g., hiding telephone numbers or a home address in public photos) and in text translation (removing the original text and inserting new translated results) [1].

Several approaches have been proposed to erase graphical text (e.g., subtitles) from color images [2–4]. In challenging cases of scene text removal, with complex backgrounds and text in various fonts and sizes, these methods often produce results with visual artifacts. Inspired by the notable success of deep learning in image transformation [5–7], recent works have introduced deep-learning-based approaches to solve this problem with promising results [8–12]. Learning-based methods can be classified into two main categories, depending on whether a mask is used. Methods without masking simply take the given image as input and remove all text from it. Such methods often leave noticeable remnants of text or incorrectly distort non-text areas, and cannot remove text locally. Other methods use a region mask, i.e., a rectangular or polygonal mask approximately indicating the text region (see Fig. 1(b)), as an additional input to facilitate text removal.

A recent mask-based text removal network (MTRNet) [10] achieved a noticeable improvement over prior works for scene text removal by focusing on text regions using an auxiliary binary mask. Its pipeline is similar to that of general image inpainting tasks [13, 14]. However, there is a key difference: the pixel values of the original input image in the text regions indicated by an auxiliary mask are

* Xuewei Bian and Chaoqun Wang contributed equally to this work.

1 National Laboratory of Pattern Recognition, Institute of Automation, Beijing 100049, China. E-mail: X. Bian, bianxuewei2017@ia.ac.cn; C. Wang, wangchaoqun2018@ia.ac.cn; W. Quan, qweizework@gmail.com (✉); J. Ye, yejuntao@gmail.com; X. Zhang, xiaopeng.zhang@ia.ac.cn; D.-M. Yan, yandongming@gmail.com.

2 School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100084, China.

Manuscript received: 2021-02-22; accepted: 2021-05-26

(a) (b) (c) (d)

Fig. 1 Example results using our proposed scene text removal method: (a) input image, (b) region mask, (c) text stroke mask obtained by our TSDNet, and (d) final result.

known for text removal, whereas the corresponding values are missing (or corrupted) for general image inpainting. Generally, as the regions to be processed (indicated by the mask) become larger, filling or removing text becomes more difficult for not only image inpainting but also text removal. In addition, in the scene text removal problem, regions not covered by text strokes do not need to be removed. The mask used by MTRNet covers several unnecessary or redundant regions (non-stroke areas), especially when text strokes are scattered sparsely. A better result could be achieved if the exact text strokes could be extracted, allowing the original contents of the input image to be preserved as much as possible.

In this paper, a novel end-to-end framework is proposed based on a *generative adversarial network* (GAN) to address this problem. The key idea of our approach is first to extract the text strokes as accurately as possible and then improve the text removal. These two processes can be further enhanced via a simple cascade. Our idea is similar to the very recent MTRNet++ [11], an extended version of MTRNet [10]. However, two key differences exist between our proposed method and MTRNet++. Firstly, our proposed method only uses the detected stroke mask (the output of our text stroke detection network) as the additional information (the region mask is also taken as the input to the text removal generative network), whereas MTRNet++ depends on the exact predicted stroke mask in their so-called fine-inpainting branch. However, determining an exact stroke mask is difficult. Thus, the result of MTRNet++ is more sensitive to errors in the predicted stroke mask. Secondly, our proposed method is conducted in a simple cascaded manner, so our detected stroke mask can be further refined and

used in the text removal generative network twice, whereas MTRNet++ only uses the stroke mask once. Therefore, our method is more efficient, as shown in Section 4.

In addition, current public datasets for scene text removal are mainly synthetic, which affects the generalization ability of the trained models. To facilitate this research and approximate the real-world setting, we have constructed a new, high-quality, large-scale dataset.

The main contributions of our work are as follows:

- a text stroke detection network (TSDNet), which can effectively distinguish text strokes from non-text areas,
- a text removal generative network,
- their combination to construct a processing unit, which is cascaded to obtain our final network which demonstrates superior performance,
- a weighted-patch-based discriminator (WD) to pay more attention to text areas of input images, making generating more realistic images easier for the generator, and
- a high-quality real-world dataset for benchmarking the scene text removal task and other related tasks.

The remainder of this paper is organized as follows. Section 2 reviews existing work. Section 3 provides motivation and details the networks in our method. Section 4 evaluates the performance of our method and provides comparisons with existing methods. Section 5 draws conclusions and discusses future work.

2 Related work

2.1 Scene text detection

Scene text detection is a fundamental step in scene understanding and is widely studied in the field of computer vision [15]. Deep learning has considerably improved the performance of scene text detection frameworks, surpassing traditional methods by large margins. Shi et al. [16] decomposed text into two locally detectable elements of segments and links, which are simultaneously detected by a fully convolutional network. Liu et al. [17] collected a curved text dataset called CTW1500 to facilitate curved text detection and proposed a method integrating transverse and longitudinal sequence connections. Chen et al. [18] proposed the concept of a weighted text border and introduced an attention module to boost detection performance. To improve detection, multiscale pyramid input is widely used but it requires much more computation. He et al. [19] achieved a remarkable speedup via a novel two-stage framework including a scale-based region proposal network and a fully convolutional network. CRAFT [20] effectively detected arbitrary text areas by exploring each character and the affinities between characters. In this work, this method is adopted as a tool to measure the performance of scene text removal (see Section 4.2).

2.2 Text/non-text image classification

Another relevant research area is text/non-text image classification, which determines whether an image block contains text. Zhang et al. [21] first proposed an effective method for text image discrimination, by combining maximally stable extremal regions [22], convolutional neural networks (CNN), and bag of words (BoW) [23]. Bai et al. [24] proposed a multiscale spatial partition network to solve this task efficiently by predicting all image blocks simultaneously in a single forward propagation. Zhao et al. [25] investigated this task from two perspectives of speed and accuracy. They used a small, shallow CNN for speed and applied knowledge distillation to improve accuracy. Very recently, Gupta and Jalal [26] combined a text detector, EAST [27], and a classification subnetwork for text/non-text image classification. Unlike these previous works, to facilitate text removal, our method aims to capture the exact positions of text strokes, at the pixel level, instead of image blocks or patches with text.

2.3 Scene text removal

Existing approaches of scene text removal can be classified into two major categories: traditional non-learning methods and deep-learning-based methods.

Traditional approaches typically use color-histograms or threshold-based methods to extract text areas, and then propagate information from non-text regions to text regions depending on pixel or patch similarity [2–4]. These methods are suitable for simple cases, with clean, well-focused text, but work less well in complex scenarios, such as images with perspective distortion and complicated backgrounds.

Recent learning-based approaches try to solve this problem with the powerful learning capacity of deep

neural networks. Nakamura et al. [8] first proposed a scene text erasure method (ST Eraser) based on a CNN and conducted text erasure patch by patch. This patch-based processing fails to localize text having a complex shape and inevitably damages the consistency and continuity of the erased result. Zhang et al. [9] designed an end-to-end trainable framework (EnsNet) with a conditional GAN to remove text from natural images. Unlike Ref. [8] which erases text in an image patch by patch, EnsNet can erase scene text on the entire image in an end-to-end manner. These two works do not use masks and need to localize and remove text simultaneously. Such methods often suffer from inaccurate text localization and incomplete text removal. To solve this problem, Tursun et al. [10] developed MTRNet. An auxiliary mask is used to provide information on where the text is, enabling MTRNet to better focus on text removal. The additional information provided by the mask is the main reason why MTRNet outperforms previous methods. MTRNet also supports partial or local text removal by user mask control. Very recently, Tursun et al. [11] proposed an extended version, MTRNet++, which introduces a mask refinement branch to turn coarse region masks into pixel-level masks. The latter are used as input to a fine-inpainting branch to provide additional text information. Based on EnsNet [9], Liu et al. [12] introduced an additional generator to construct a two-stage coarse-to-refine network like Ref. [28]. Moreover, they collected a new dataset, SCUT-EnsText, containing 3562 images. All these existing approaches often leave several text strokes unchanged or generate unpleasant results because they do not appropriately and exactly pay attention to the text strokes. Another shortcoming of the current methods is that their training datasets are mainly synthetic because the collection of real-world datasets is difficult and time consuming.

3 Method

3.1 Network architecture

We combine a text stroke detection network with a text removal generative network to construct a processing unit. The final network is obtained by cascading this unit and combining it with a weighted-patch-based discriminator.

3.1.1 Cascaded generator

The proposed generator has two purposes, firstly to detect text strokes in the input image accurately, and secondly to inpaint the detected text strokes with proper content. To achieve the first goal, a text stroke detection network (TSDNet) is constructed. For the second goal, a text removal generative network (TRGNet) is proposed. The whole generator is obtained by cascading the group of TSDNet and TRGNet, as shown in Fig. 2. The parameters in these four networks are not shared. Technically, TSDNet and TRGNet employ a U-Net-like architecture [29] because, compared to a simple encoder–decoder framework, the U-Net architecture with skip connections helps recover the structure and texture details of unmasked areas from input images

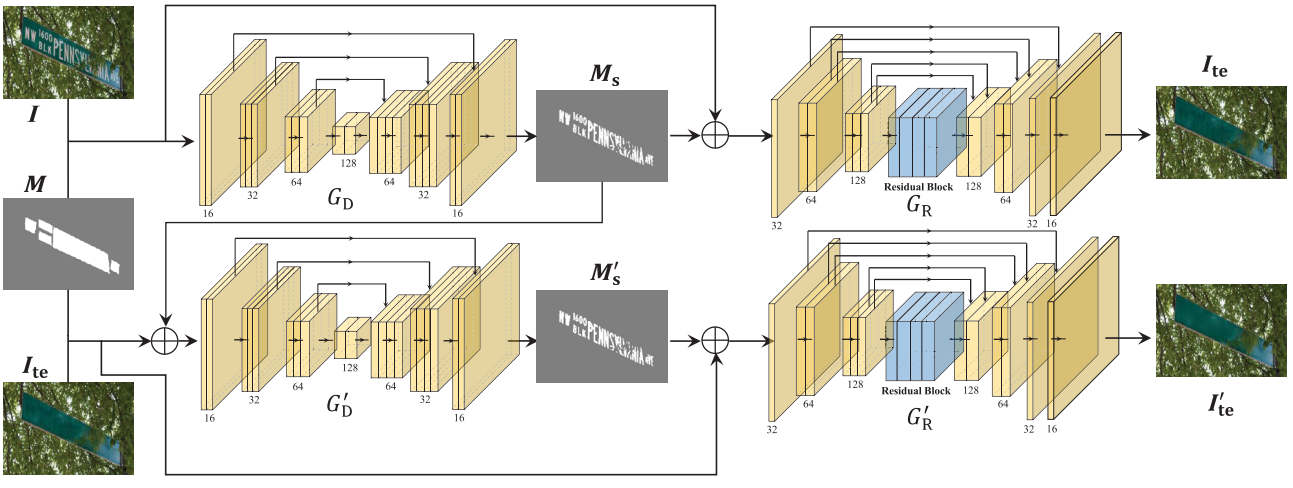


Fig. 2 Overall structure of the proposed generator, using cascaded text stroke detection and text removal generation. ⊕ indicates concatenation of image, region mask, and stroke mask. The convolutional kernel size of the first layer of G_R and G'_R is 5×5, and the remaining kernel size is 3×3.

as well as avoiding over-smoothing and undesired artifacts.

The inputs to TSDNet (denoted G_{D}) are a text image $\boldsymbol{I}$ and a binary mask $\boldsymbol{M}$ (indicating the text regions). The output is a matrix $\boldsymbol{M}_{\mathrm{s}}$ of the same size as $\boldsymbol{M}$; it contains floats in [0, 1], larger values indicating a higher confidence that the corresponding position in image $\boldsymbol{I}$ is covered by a text stroke:

$$\boldsymbol{M}_{\mathrm{s}} = G_{\mathrm{D}}(\boldsymbol{I}, \boldsymbol{M}) \tag{1}$$

The ground truth of text stroke distribution is a binary mask $\boldsymbol{M}_{\mathrm{gt}}$, in which 1 means the corresponding position in $\boldsymbol{I}$ is covered by a text stroke. Unlike $\boldsymbol{M}$ which only specifies the approximate region of a certain text element, $\boldsymbol{M}_{\mathrm{gt}}$ is a pixel-level annotation of text strokes. In practice, $\boldsymbol{M}_{\mathrm{gt}}$ is obtained by binarizing the difference between the paired text image $\boldsymbol{I}$ and text-free image $\boldsymbol{I}_{\mathrm{gt}}$ (more details are given in Section 4.1), and this stroke annotation is only used in the training stage as supervisory information to train TSDNet.

After obtaining stroke mask $\boldsymbol{M}_{\mathrm{s}}$, TRGNet G_{R} is then applied to erase text from input image $\boldsymbol{I}$. G_{R} takes three items as input: text image $\boldsymbol{I}$, binary mask $\boldsymbol{M}$, and the stroke mask $\boldsymbol{M}_{\mathrm{s}}$ obtained from G_{D}; it outputs text-erased image $\boldsymbol{I}_{\mathrm{te}}$:

$$\boldsymbol{I}_{\mathrm{te}} = G_{\mathrm{R}}(\boldsymbol{I}, \boldsymbol{M}, \boldsymbol{M}_{\mathrm{s}}) \tag{2}$$

A TSDNet followed by a TRGNet (top row in Fig. 2) can already detect and erase text effectively, but the resulting images ($\boldsymbol{I}_{\mathrm{te}}$) sometimes contain artifacts and slight remnants of text. We observe that a simple cascade can eliminate such artifacts and bring substantial visual improvement. Thus, in our architecture, the second TSDNet G'_{D} takes $\boldsymbol{I}_{\mathrm{te}}$, $\boldsymbol{M}$, and $\boldsymbol{M}_{\mathrm{s}}$ as input, and outputs $\boldsymbol{M}'_{\mathrm{s}}$. Then, the second TRGNet G'_{R} takes $\boldsymbol{I}_{\mathrm{te}}$, $\boldsymbol{M}$, and $\boldsymbol{M}'_{\mathrm{s}}$ as input, and outputs the final text-erased result $\boldsymbol{I}'_{\mathrm{te}}$. By combining the previous outputs, G'_{D} acquires a more accurate text stroke distribution in an incremental manner. Thus, G'_{R} can effectively reduce artifacts and inconsistency.

Previous studies such as EnsNet and ST Eraser, which simply take an image or patch as input and try to erase text without prior information, showed relatively limited performance. In this work, we use a binary mask specifying the text region as additional information to decrease the difficulty of detecting and erasing text at the same time, and design a TSDNet to provide more accurate instruction as to which area should be removed. By doing so, text removal is successfully decoupled into text stroke detection and stroke removal, leading to a proposed framework giving an effective solution to solve these decoupled problems.

3.1.2 Weighted-patch-based discriminator

A patch-based discriminator (see Fig. 3) is useful to concentrate effort on altered areas as text removal only needs to change parts of the input image. In this work, the discriminator proposed in SN-PatchGAN [28, 30] is used to discriminate the text-erased image patch by patch. The original discriminator is further improved by attaching an additional convolutional branch D_M to discriminator D to assign various weights to different patches according to mask $\boldsymbol{M}$. D_M has the same architecture as D, but each layer only has one channel, and weights in the convolutional kernel are fixed to 1. By doing so, the patches covered by more text will be given more attention.

3.2 Training loss

We now present our loss functions for the generator and discriminator. To verify that our proposed method is valid, a relatively simple loss function is used when training our network. For TSDNet G_{D} and G'_{D}, simple l_1 loss is used:

$$\mathcal{L}_{\mathrm{TSD}} = \boldsymbol{E}\left[\|\boldsymbol{M}_{\mathrm{s}} - \boldsymbol{M}_{\mathrm{gt}}\|_1 + \lambda_t \|\boldsymbol{M}'_{\mathrm{s}} - \boldsymbol{M}_{\mathrm{gt}}\|_1\right] \tag{3}$$

where λ_t balances the l_1 losses of G_{D} and G'_{D}. $\lambda_t = 10$ in all our experiments, because most text strokes have been detected by G_{D}.

For the scene text removal task, our main goal is to remove text and preserve non-text regions. Therefore,

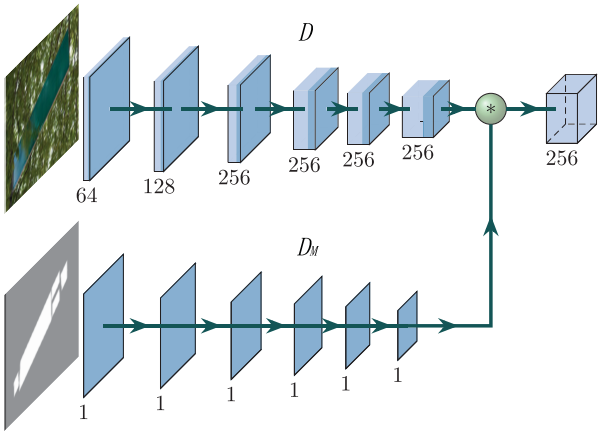


Fig. 3 Architecture of our weighted-patch-based discriminator. $*$ denotes element-wise multiplication between two branches with broadcasting. A 5×5 convolutional kernel is used.

more attention is paid to masked areas (indicated by $\boldsymbol{M}$), especially the detected stroke areas (indicated by $\boldsymbol{M}_{\mathrm{s}}/\boldsymbol{M}'_{\mathrm{s}}$). More precisely, the corresponding weight matrices $\boldsymbol{M}_{\mathrm{w}}$ and $\boldsymbol{M}'_{\mathrm{w}}$ for G_{R} and G'_{R} are defined as

$$\begin{aligned}\boldsymbol{M}_{\mathrm{w}} &= \mathbb{1} + \lambda_{\mathrm{m}}\boldsymbol{M} + \lambda_{\mathrm{s}}\boldsymbol{M}_{\mathrm{s}}\\ \boldsymbol{M}'_{\mathrm{w}} &= \mathbb{1} + \lambda_{\mathrm{m}}\boldsymbol{M} + \lambda_{\mathrm{s}}\boldsymbol{M}'_{\mathrm{s}}\end{aligned} \tag{4}$$

where $\mathbb{1}$ has all elements 1, and the same shape as $\boldsymbol{M}$. Then, the total loss for G_{R} and G'_{R} is defined as

$$\begin{aligned}\mathcal{L}_{\mathrm{TRG}} = \boldsymbol{E}\Big[&\|\boldsymbol{I}_{\mathrm{te}} \odot \boldsymbol{M}_{\mathrm{w}} - \boldsymbol{I}_{\mathrm{gt}} \odot \boldsymbol{M}_{\mathrm{w}}\|_1 + \\ &\lambda_{\mathrm{r}}\|\boldsymbol{I}'_{\mathrm{te}} \odot \boldsymbol{M}'_{\mathrm{w}} - \boldsymbol{I}_{\mathrm{gt}} \odot \boldsymbol{M}'_{\mathrm{w}}\|_1\Big]\end{aligned} \tag{5}$$

where $\odot$ is the element-wise product, and λ_{r} is a balancing parameter. In all our experiments, we set $\lambda_{\mathrm{m}} = 5$, $\lambda_{\mathrm{s}} = 5$, and $\lambda_{\mathrm{r}} = 10$.

For the objective function of patch-based GAN, the hinge version of adversarial loss [31, 32] is used. The corresponding loss functions for the generator and discriminator are defined respectively as

$$\mathcal{L}_{\mathrm{G}}^{\mathrm{sn}} = -\boldsymbol{E}\Big[D_M(\boldsymbol{M}) \odot D(\boldsymbol{I}'_{\mathrm{te}})\Big] \tag{6}$$

$$\begin{aligned}\mathcal{L}_{\mathrm{D}}^{\mathrm{sn}} = \boldsymbol{E}\Big[&ReLU(1 - D_M(\boldsymbol{M}) \odot D(\boldsymbol{I}_{\mathrm{gt}}))\Big] + \\ \boldsymbol{E}\Big[&ReLU(1 + D_M(\boldsymbol{M}) \odot D(\boldsymbol{I}'_{\mathrm{te}}))\Big]\end{aligned} \tag{7}$$

where $\odot$ means element-wise product with broadcasting in terms of depth.

In summary, the total loss for our cascaded generator combines Eqs. (3), (5), and (6):

$$\mathcal{L}_{\mathrm{G}} = \mathcal{L}_{\mathrm{TSD}} + \mathcal{L}_{\mathrm{TRG}} + \mathcal{L}_{\mathrm{G}}^{\mathrm{sn}} \tag{8}$$

We note that perceptual loss [5] and style loss [33] provide no noticeable improvements for our task. One reason is that scene text is usually located in a relatively flat area. Total variation loss [34] has no apparent effect on the erased result either, so it is not used in our method.

4 Experimental results

To evaluate our proposed method quantitatively and qualitatively, we compare it with recent state-of-the-art text removal methods. An ablation study is also conducted to evaluate different components of our network.

4.1 Dataset

To train a deep model for text removal, paired images with and without text are required. However, obtaining such data for real-world scenes is difficult, which is why synthetic datasets are mainly used in most existing approaches. Two public synthetic datasets are available: the Oxford synthetic scene text detection dataset [35] and the SCUT synthetic text removal dataset [9]. These two datasets adopt the same synthesis technology proposed by Ref. [35] and share the same drawback: given a text-free image as background, multiple images with text are synthesized. For instance, the Oxford dataset synthesized 800,000 images using only 8000 text-free images, so 100 text images share each background image, leading to insufficient background diversity. Such repetition negatively affects the generalization ability of models. Very recently, Liu et al [12]. collected a real-world dataset, SCUT-EnsText, but it only has 3562 images, with text in two languages (Chinese and English). Thus, we have constructed a larger real-world dataset with multilingual text for text removal.

To construct our dataset, 5070 images with text were first collected from the ICDAR2017 MLT dataset [36], and 1970 images captured from supermarkets and streets. These were then processed to obtain corresponding text-free images, region masks, and text stroke masks. The text from the collected images was manually removed using the inpainting tools in Photoshop to provide text-free images as ground truth. Region masks were annotated using the VGG Image Annotator tool [37]. To get the ground truth stroke masks, the difference between paired images with and without text is first computed and then turned into a binary image. To enrich the diversity of our dataset, synthesis is also used, and 4000 images with high realism were manually selected. A total of 11,040 images was obtained as the training set (Train_rw). Several samples of our dataset are shown in Fig. 4. To construct the testing set (Test_rw), 1080 further real-world images were collected, and the above post processing is applied to obtain text-free images, region masks, and text stroke masks. Our collected real-world dataset contains 12,120 images and has text in multiple languages as reported in Table 1.

For the public synthetic dataset, we use the Oxford dataset [35], which is much larger than the SCUT dataset [9]. 75% of the Oxford dataset was randomly selected as the training set (Train_ox), and 2000 images were randomly selected from the remainder as testing set (Test_ox). For the SCUT-EnsText dataset, the original split is followed as reported in Ref. [12].

Fig. 4 Examples from our dataset. Left to right: images with text, text-free images, region masks, and text stroke masks.

Table 1 Language statistics for our dataset; one image may contain multiple languages

Latin	Arabic	Chinese	Japanese	Korean	Bangla
8604	765	3807	819	848	519

4.2 Evaluation metrics

Performance is evaluated according to two different criteria: Can the method completely remove text from an image? Is the text area replaced with appropriate content? An accurate text detector is often used to assess the former; we use the state-of-the-art detector CRAFT [20] and DetEval protocol [38] for evaluation via recall, precision, and f-measure. For the second criterion, general image inpainting metrics are adopted: mean absolute error (MAE), peak signal-to-noise ratio (PSNR), and structural similarity index (SSIM).

4.3 Implementation details

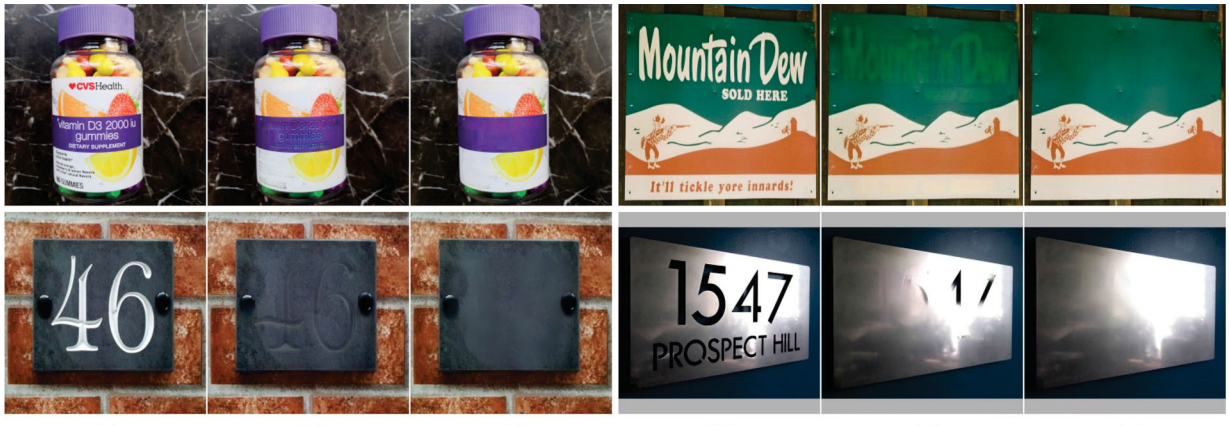
Our network was implemented using TensorFlow 1.13 on an nVidia TITAN RTX GPU. Input images were resized to 256×256. The Adam optimizer [39] was used with a mini-batch size of 16 to train our network, with $\beta_1 = 0.5$ and $\beta_2 = 0.9$. The initial learning rate was set to 0.0001.

4.4 Dataset comparison

Figure 5 compares example results using the Oxford synthetic dataset and our real-world dataset for training, using two different networks (MTRNet and our proposed network). Each group of 3 images shows the input image, result when using the Oxford dataset got training, and result using our dataset for training. In each case, using our dataset, the text is better removed, especially for our network, with no noticeable text remnants and better preservation of the original image details, such as lighting effects. This demonstrates that our dataset is more suitable for training scene text removal networks, even though the Oxford dataset has more images.

4.5 Comparison with the state-of-the-art methods

In this subsection, our method is quantitatively and qualitatively compared with recent state-of-the-art text removal methods: ST Eraser [8], EnsNet [9],



(a) (b) (c) (a) (b) (c)

Fig. 5 Effects on generalization ability of using the Oxford dataset and our dataset for training. Above: results from MTRNet. Below: results from our method. Each group of 3 images: (a) input image, (b) text removal result trained on the Oxford dataset, and (c) text removal result trained on our dataset.

MTRNet [10], MTRNet++ [11], and EraseNet [12]. The official implementation of EnsNet and EraseNet is used, and ST Eraser, MTRNet, and MTRNet++ are reimplemented.

Table 2 quantitatively compares these six methods on the Oxford dataset and our dataset. Our method is superior to the other methods by a large margin for all metrics. When training on Train_rw and testing on Test_ox, our method achieves the best performance, and this phenomenon is repeated when training on Train_ox and testing on Test_rw. This further indicates that our network has better generalization ability. For cross-dataset validation, the results of training on Train_ox and testing on Test_ox are relatively similar to those of training on Train_rw and testing on Test_ox (see Table 2, columns 9–14). However, the performance when training on Train_rw and testing on Test_rw is evidently better than when training on Train_ox and testing on Test_rw (see Table 2, columns 3–8, e.g., the PSNR and SSIM of EnsNet are improved from 26.41 to 33.78 and from 87.30% to 95.43%, respectively. These results indicate that our dataset is more suitable for this scene text removal task, especially for real-world applications.

Figure 6 shows text-erased images using all six methods. Three groups of three rows separately correspond to using the Oxford dataset [35], the SCUT-EnsText dataset [12], and our dataset. Compared to other text removal methods, our method more effectively erases text and inpaints the text area with proper content. In the first row of Fig. 6, our result preserves texture details more consistent with the original non-text areas, while other methods result in noticeable text remnants or visual inconsistency. The last row of Fig. 6 similarly shows that our method can preserve the wrinkles of the T-shirt. Our method can also better process oblique text (see rows 5, 6). In the penultimate row, our result has no text remnants and preserves the original light transition well.

In addition, in Table 3 we compare the model complexity in terms of the number of learnable parameters and FLOPs required. Our method is intermediate for both measures. The relatively small number of FLOPs required indicates that the speed of our method is competitive.

4.6 Multilingual and selective text removal

In this subsection, further results are provided in Fig. 7 showing multilingual text removal (top 3 rows) and selective text removal (bottom 3 rows. MTRNet [10], MTRNet++ [11], and our method were trained on our real-world dataset.

Compared to MTRNet and MTRNet++, our

Table 3 Comparison of model complexity in terms of the number of learnable parameters and FLOPs needed. The largest and smallest values are indicated in red and blue respectively

Method	Parameters (M)	FLOPs (G)
ST Eraser	0.27	100.86
MTRNet	0.34	66.79
MTRNet++	3.76	62.51
EraseNet	19.74	14.64
Ours	3.79	57.48

Table 2 Quantitative comparison of our method to the state-of-the-art methods. All methods were trained and tested on the Oxford dataset and our dataset separately. For PSNR and SSIM, higher is better; for MAE, R (recall), P (precision), and F (f-measure), lower is better

	Testing set	Test_rw						Test_ox					
Training set	Method	MAE	PSNR	SSIM (%)	R (%)	P (%)	F (%)	MAE	PSNR	SSIM (%)	R (%)	P (%)	F (%)
	Original image	—	—	—	43.25	40.68	41.93	—	—	—	48.24	67.93	56.42
Train_rw	ST Eraser	2.52	27.20	91.13	6.23	20.55	9.56	2.67	27.94	90.24	14.42	49.27	22.31
	EnsNet	1.22	33.78	95.43	1.94	20.18	3.53	1.89	31.37	93.03	7.25	49.84	12.65
	MTRNet	1.62	34.31	96.34	0.55	17.14	1.06	2.31	31.81	92.36	0.49	37.27	0.96
	MTRNet++	0.97	36.02	96.28	0.99	15.87	1.86	1.71	32.56	93.60	0.76	39.53	1.49
	EraseNet	1.16	34.10	95.55	1.58	19.34	2.92	1.77	32.12	93.87	5.91	44.32	10.43
	Ours	**0.75**	**39.44**	**97.56**	**0.35**	**10.23**	**0.68**	**1.63**	**34.40**	**93.97**	**0.05**	**15.38**	**0.10**
Train_ox	ST Eraser	4.26	21.52	82.20	28.34	36.17	31.78	5.77	20.92	77.05	21.56	48.60	29.87
	EnsNet	2.55	26.41	87.30	23.77	34.56	28.17	1.75	32.76	93.20	4.07	46.09	7.48
	MTRNet	1.89	32.53	92.68	11.07	34.95	16.82	2.24	31.79	92.06	1.79	42.75	3.43
	MTRNet++	1.31	35.12	96.02	12.21	33.18	17.85	1.77	33.97	93.80	2.19	44.37	4.17
	EraseNet	1.64	33.85	93.11	15.66	18.29	16.87	1.85	33.25	93.55	3.82	45.51	7.05
	Ours	**1.23**	**36.23**	**96.64**	**0.33**	**10.89**	**0.64**	**1.72**	**34.48**	**94.47**	**0.00**	**0.00**	**0.00**

Fig. 6 Qualitative comparison of our method to ST Eraser, EnsNet, MTRNet, MTRNet++, and EraseNet. Top to bottom, in groups of 3 rows: using Oxford dataset, SCUT-EnsText dataset, and our dataset.

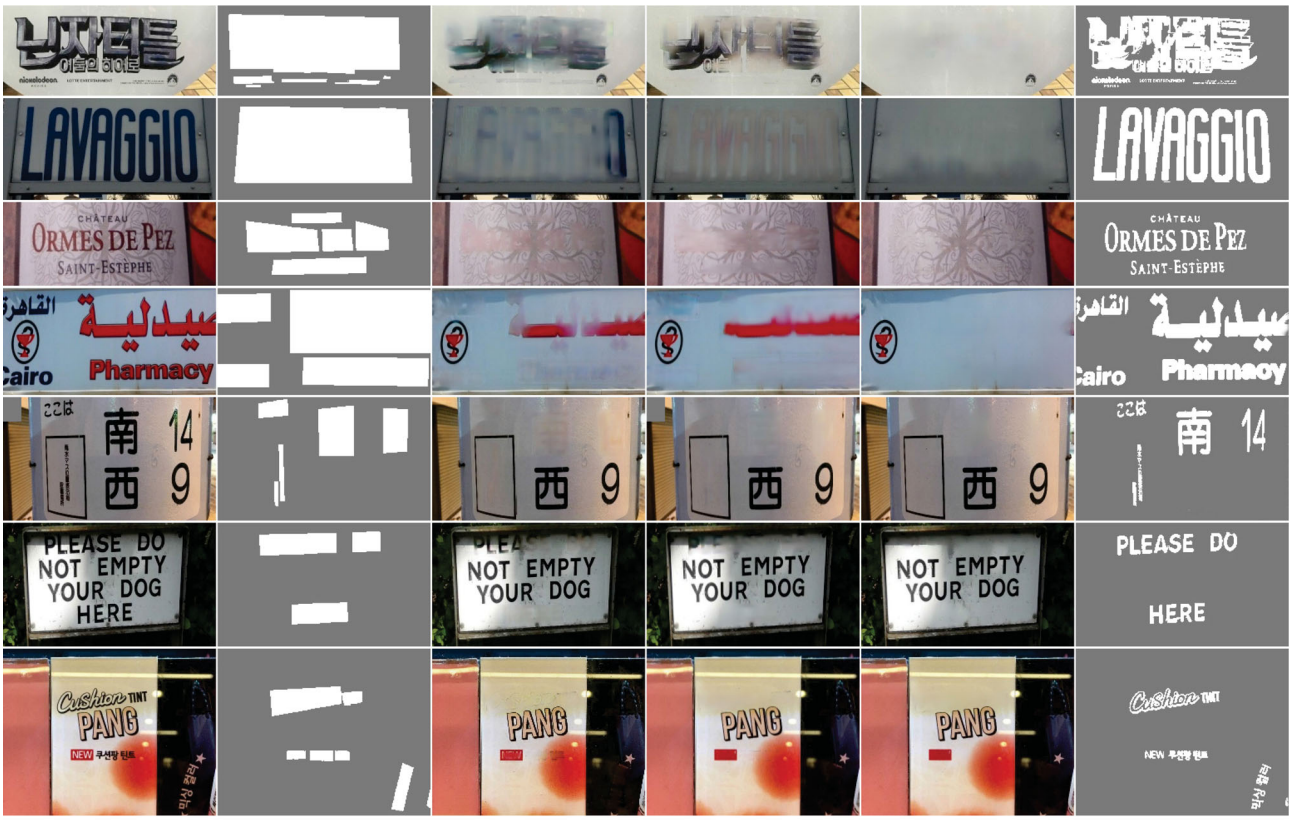


Fig. 7 Multilingual text removal (top 3 rows) and selective text removal (bottom 3 rows). Left to right: original image, input mask, MTRNet output, MTRNet++ output, our output, and stroke mask detected by our method.

method can more successfully remove text in multiple languages because our text stroke detection network more efficiently focuses on the text and even learns the differences between various languages. Thus, it provides more useful information in the form of accurate stroke masks for the consecutive text removal generative network than a region mask.

Our method can also accomplish selective text removal. Given an auxiliary mask, where the text to be removed is indicated by a user-provided polygonal mask, our method can remove the desired text without affecting other text.

4.7 Ablation study

Next, the effects of different components of our network are considered.

4.7.1 Baseline

For the baseline model, a single TRGNet G_R (shown in Fig. 2) is used as the generator, and the discriminator proposed in SN-PatchGAN [28] is used as the discriminator D in Fig. 3. The inputs to TRGNet here are a text image $\boldsymbol{I}$ and a binary mask $\boldsymbol{M}$. Comparing Tables 2 and 4 shows that this baseline model already has similar performance to previous text removal methods, including EnsNet which does not use an auxiliary mask and MTRNet which uses a region mask. The visual results are also good, as shown in the second column of Fig. 8.

4.7.2 Weighted-patch-based discriminator

The original discriminator proposed in SN-PatchGAN treats all patches equally. Our work focuses on masked regions, and a weighted discriminator, which can pay more attention to the masked area by assigning a higher weight, is used. Comparing Base-

Table 4 Ablation study. Models are trained on Train_rw and tested on Test_rw. tMAE is the mean absolute error between the detected stroke mask and ground truth stroke mask

Method	MAE	PSNR	SSIM (%)	tMAE (%)
Baseline	1.59	35.00	95.42	—
WD	1.00	38.31	97.22	—
TSDNet	0.98	38.17	97.33	7.63
WD + TSDNet	0.92	38.47	97.48	4.85
Cascade	**0.75**	**39.44**	**97.56**	**4.73**

Fig. 8 Qualitative results of ablation study. The last column gives the best result.

line and WD rows in Table 4 shows that our proposed weighted discriminator can significantly improve performance over that of the baseline model. The first row in Fig. 8 further shows that our proposed WD can help maintain structural consistency in a given image.

4.7.3 Text stroke detection network

A TSDNet is added to the baseline model to demonstrate the effectiveness of accurate text stroke extraction. Table 4 shows that a baseline model with TSDNet achieves much higher PSNR and SSIM, and a much lower MAE (see rows Baseline and TSDNet). Our proposed TSDNet can effectively distinguish whether the given area is a text stroke, information which can help TRGNet to remove masked areas more purposefully. Combining WD and TSDNet (row WD+TSDNet in Table 4) shows a further performance improvement. Results in the second row of Fig. 8 shows how our TSDNet can help to completely remove text from an image (see the character T).

4.7.4 Cascaded TSDNet and TRGNet

Cascading TSDNet and TRGNet can help fix minor mistakes and slight text remnants left in the results of the first unit of TSDNet and TRGNet. This includes completing partially detected text strokes, removing residual text, and fixing visual artifacts. We also experimented with using three cascaded

units, but the text-erased results were slightly blurred, possibly because some high-frequency information is lost during cascading.

4.7.5 Stroke detection

Text stroke detection is an important component of our generic framework. Here, the effects of stroke detection performance on final text removal are further discussed. Inserting TSDNet into the baseline clearly improves performance validating our design for TSDNet. Furthermore, text stroke detection performance is enhanced via the cascaded design (tMAE for Cascade is substantially smaller than for TSDNet in Table 4), resulting in better text removal. This finding further illustrates that improved stroke detection can enhance the final text removal result. In addition, tMAE of MTRNet++ is significantly worse than that of our method. Figure 9 shows several examples of text stroke detection: our results are clearer and more exact. This improvement in stroke detection may be the main reason for the limited performance of MTRNet++, and why our method is more effective than MTRNet++.

Fig. 9 Comparison of text stroke detection methods.

5 Conclusions

In this work, a novel GAN-based framework is proposed for the scene text removal task by decoupling text stroke detection and stroke removal. A text stroke detection network and a text removal generative network are designed and implemented, and the final model is constructed by cascading the combination of the two networks. Quantitative and qualitative studies illustrate the superior results of our proposed network. Our study implies that knowing the positions of text strokes is beneficial to the scene text removal problem. A versatile real-world dataset, including text images, ground truth text-free images, and auxiliary masks, which can be used to benchmark the text removal methods, has been constructed. Moreover, our approach can be used to quickly construct a large scale text-free image dataset from images with text, and pixel-wise text stroke annotations can be obtained by binarizing the difference between paired text image and text-free image. Such datasets will provide better, more fine-grained supervised information to improve the performance of scene text detection and recognition tasks. Our source code is available at `https://github.com/wcq19941215/SceneTextRemoval`.

Our method can generate implausible results if the text area is too large. We believe that a larger dataset with more diverse data could help to mitigate existing shortcomings. In future, we plan to collect more real-world text images and construct a larger, richer dataset that can be used for the text removal task and other related research, e.g., realistic text synthesis. In this work, text region masks are used in an offline manner, considering the imperfect performance of current automatic text detectors and the requirements of partial text removal applications. We hope to design a more complete framework combining automatic text detection, which supports refinement of possible detection errors and selection of specific text regions with simple user guidance. Studying semi-supervised text removal would also be engaging.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (62102418 and 62172415), the National Key R&D Program of China (2019YFB2204104), and the Open Research Fund Program of State key Laboratory of Hydroscience and Engineering, Tsinghua University (sklhse-2020-D-07). We would like to thank Shiyu Hou and Shuo Liu for helping collect the dataset.

References

[1] Wu, L.; Zhang, C. Q.; Liu, J. M.; Han, J. Y.; Liu, J. T.; Ding, E. R.; Bai, X. Editing text in the wild. In: Proceedings of the 27th ACM International Conference on Multimedia, 1500–1508, 2019.

[2] Khodadadi, M.; Behrad, A. Text localization, extraction and inpainting in color images. In: Proceedings of the 20th Iranian Conference on Electrical Engineering, 1035–1040, 2012.

[3] Modha, U.; Dave, P. Image inpainting-automatic detection and removal of text from images. *International Journal of Engineering Research and Applications* Vol. 2, No. 2, 930–932, 2012.

[4] Wagh, P. D.; Patil, D. R. Text detection and removal from image using inpainting with smoothing. In: Proceedings of the International Conference on Pervasive Computing, 1–4, 2015.

[5] Johnson, J.; Alahi, A.; Li, F. F. Perceptual losses for real-time style transfer and super-resolution. In: *Computer Vision – ECCV 2016. Lecture Notes in Computer Science, Vol. 9906*. Leibe, B.; Matas, J.; Sebe, N.; Welling, M. Eds. Springer Cham, 694–711, 2016.

[6] Isola, P.; Zhu, J. Y.; Zhou, T. H.; Efros, A. A. Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 5967–5976, 2017.

[7] Zhu, J.-Y.; Park, T.; Isola, P.; Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision, 2242–2251, 2017.

[8] Nakamura, T.; Zhu, A.; Yanai, K.; Uchida, S. Scene text eraser. In: Proceedings of the International Conference on Document Analysis and Recognition, 832–837, 2017.

[9] Zhang, S.; Liu, Y.; Jin, L.; Huang, Y.; Lai, S. EnsNet: Ensconce text in the wild. In: Proceedings of the AAAI Conference on Artificial Intelligence, 801–808, 2019.

[10] Tursun, O.; Zeng, R.; Denman, S.; Sivapalan, S.; Sridharan, S.; Fookes, C. MTRNet: A generic scene text eraser. In: Proceedings of the International Conference on Document Analysis and Recognition, 2019.

[11] Tursun, O.; Denman, S.; Zeng, R.; Sivapalan, S.; Sridharan, S.; Fookes, C. MTRNet++: One-stage mask-based scene text eraser. *Computer Vision and Image Understanding* Vol. 201, 103066, 2020.

[12] Liu, C. Y.; Liu, Y. L.; Jin, L. W.; Zhang, S. T.; Luo, C. J.; Wang, Y. P. EraseNet: End-to-end text removal in the wild. *IEEE Transactions on Image Processing* Vol. 29, 8760–8775, 2020.

[13] Iizuka, S.; Simo-Serra, E.; Ishikawa, H. Globally and locally consistent image completion. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 107, 2017.

[14] Yu, J. H.; Lin, Z.; Yang, J. M.; Shen, X. H.; Lu, X.; Huang, T. S. Generative image inpainting with contextual attention. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5505–5514, 2018.

[15] Ye, Q. X.; Doermann, D. Text detection and recognition in imagery: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 37, No. 7, 1480–1500, 2015.

[16] Shi, B. G.; Bai, X.; Belongie, S. Detecting oriented text in natural images by linking segments. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3482–3490, 2017.

[17] Liu, Y. L.; Jin, L. W.; Zhang, S. T.; Luo, C. J.; Zhang, S. Curved scene text detection via transverse and longitudinal sequence connection. *Pattern Recognition* Vol. 90, 337–345, 2019.

[18] Chen, J.; Lian, Z. H.; Wang, Y. Z.; Tang, Y. M.; Xiao, J. G. Irregular scene text detection via attention guided border labeling. *Science China Information Sciences* Vol. 62, No. 12, 220103, 2019.

[19] He, W. H.; Zhang, X. Y.; Yin, F.; Luo, Z. B.; Ogier, J. M.; Liu, C. L. Realtime multi-scale scene text detection with scale-based region proposal network. *Pattern Recognition* Vol. 98, 107026, 2020.

[20] Baek, Y.; Lee, B.; Han, D.; Yun, S.; Lee, H. Character region awareness for text detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 9357–9366, 2019.

[21] Zhang, C.; Yao, C.; Shi, B.; Bai, X. Automatic discrimination of text and non-text natural images. In: Proceedings of the 13th International Conference on Document Analysis and Recognition, 886–890, 2015.

[22] Matas, J.; Chum, O.; Urban, M.; Pajdla, T. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing* Vol. 22, No. 10, 761–767, 2004.

[23] Joachims, T. Text categorization with Support Vector Machines: Learning with many relevant features. In: *Machine Learning: ECML-98. Lecture Notes in Computer Science (Lecture Notes in Artificial*

Intelligence), Vol. 1398. Nédellec, C.; Rouveirol, C. Eds. Springer Berlin Heidelberg, 137–142, 1998.

[24] Bai, X.; Shi, B. G.; Zhang, C. Q.; Cai, X.; Qi, L. Text/non-text image classification in the wild with convolutional neural networks. *Pattern Recognition* Vol. 66, 437–446, 2017.

[25] Zhao, M.; Wang, R.-Q.; Yin, F.; Zhang, X.-Y.; Huang, L.-L.; Ogier, J.-M. Fast text/non-text image classification with knowledge distillation. In: Proceedings of the International Conference on Document Analysis and Recognition, 1458–1463, 2019.

[26] Gupta, N.; Jalal, A. S. Text or non-text image classification using fully convolution network (FCN). In: Proceedings of the International Conference on Contemporary Computing and Applications, 150–153, 2020.

[27] Zhou, X.; Yao, C.; Wen, H.; Wang, Y.; Zhou, S.; He, W.; Liang, J. EAST: An efficient and accurate scene text detector. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2642–2651, 2017.

[28] Yu, J. H.; Lin, Z.; Yang, J. M.; Shen, X. H.; Lu, X.; Huang, T. Free-form image inpainting with gated convolution. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 4470–4479, 2019.

[29] Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. Lecture Notes in Computer Science, Vol. 9351*. Navab, N.; Hornegger, J.; Wells, W.; Frangi, A. Eds. Springer Cham, 234–241, 2015.

[30] Miyato, T.; Kataoka, T.; Koyama, M.; Yoshida, Y. Spectral normalization for generative adversarial networks. In: Proceedings of the International Conference on Learning Representations, 2018.

[31] Tran, D.; Ranganath, R.; Blei, D. Hierarchical implicit models and likelihood-free variational inference. In: Proceedings of the Advances in Neural Information Processing Systems, 5523–5533, 2017.

[32] Zhang, H.; Goodfellow, I.; Metaxas, D.; Odena, A. Self-attention generative adversarial networks. In: Proceedings of the 36th International Conference on Machine Learning, 7354–7363, 2019.

[33] Gatys, L. A.; Ecker, A. S.; Bethge, M. Image style transfer using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2414–2423, 2016.

[34] Aly, H. A.; Dubois, E. Image up-sampling using total-variation regularization with a new observation model. *IEEE Transactions on Image Processing* Vol. 14, No.10, 1647–1659, 2005.

[35] Gupta, A.; Vedaldi, A.; Zisserman, A. Synthetic data for text localisation in natural images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2315–2324, 2016.

[36] Nayef, N.; Yin, F.; Bizid, I.; Choi, H.; Feng, Y.; Karatzas, D.; Luo, Z.; Pal, U.; Rigaud, C.; Chazalon, J. et al. ICDAR2017 robust reading challenge on multi-lingual scene text detection and script identification-RRC-MLT. In: Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition, 1454–1459, 2017.

[37] Dutta, A.; Zisserman, A. The VIA annotation software for images, audio and video. In: Proceedings of the 27th ACM International Conference on Multimedia, 2276–2279, 2019.

[38] Wolf, C.; Jolion, J.-M. Object count/area graphs for the evaluation of object detection and segmentation algorithms. *International Journal on Document Analysis and Recognition* Vol. 8, No. 4, 280–296, 2006.

[39] Kingma, D. P.; Ba, J. Adam: A method for stochastic optimization. In: Proceedings of the International Conference on Learning Representations, 2015.

Xuewei Bian received his M.S. degree in computer technology from the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing, in 2020. He received his B.S. degree in automation from Beijing University of Chemical Technology, in 2017. His research interests include image processing and computer vision.

Chaoqun Wang is pursuing an M.S. degree in the School of Artificial Intelligence, Chinese Academy of Sciences, Beijing. He received his B.S. degree in computer network from Jianghan University, Wuhan, in 2018. His research interests include image processing and computer vision.

Weize Quan received his Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences and the Université Grenoble Alpes, France, in 2020, and his bachelor degree from Wuhan University of Technology in 2014. He is currently an assistant professor at the NLPR. His research interests include image processing and computer graphics.

Juntao Ye obtained his B.Eng. degree from Harbin Engineering University in 1994, his M.Sc. degree from the Institute of Computational Mathematics and Scientific/Engineering Computing, Chinese Academy of Sciences, in 2000, and his Ph.D. in computer science from The University of Western Ontario, Canada, in 2005. His research interests include graphics, particularly physically-based simulation of cloth and fluid, as well as image/video processing.

Xiaopeng Zhang is a professor in the NLPR. He received his Ph.D. degree in computer science from the Institute of Software, Chinese Academy of Sciences, in 1999. He received a National Scientific and Technological Progress Prize (second class) in 2004. His main research interests include computer graphics and image processing.

Dong-Ming Yan is a professor in the NLPR. He received his Ph.D. degree in computer science from Hong Kong University in 2010, and his master and bachelor degrees in computer science and technology from Tsinghua University in 2005 and 2002. His research interests include computer graphics, geometry processing, and visualization.