** Supplementary material **
On Manipulating Scene Text in the Wild with Diffusion Models

In this supplementary material, we provide (i) more details about our implementation setting and the SynText dataset used in our experiments (e.g., dataset creation), (ii) additional experiments on one-shot ablation studies, and (iii) additional qualitative results on ICDAR, COCO-Text, SVT, IIIT5K, and HierText datasets as well as Youtube video.

1. Implementation Setting

We began by setting up the pre-trained LDM model, which uses text-to-image technology [10], and incorporated the ABINet text recognition model [2]. Our training process involved utilizing the Syntext dataset for 500k iterations, with a learning rate of $1e^{-6}$ and a batch size of 1. It's important to note that we only trained the diffusion model while keeping the language model unchanged from its original state. To manipulate the text in the input image, we fine-tuned the diffusion model using a given image for 1500 iterations, with a learning rate of $1e^{-6}$. Next, we optimized the target embedding using cross-entropy loss over 1000 iterations, with a learning rate of $1e^{-4}$. We employed an ADAM optimizer and performed on a single RTX 3090.

2. Synthesized Dataset

Our main paper briefly explains that our synthesized scene texts are generated using SynText [3]. Below, we provide a list of steps to build the synthesized text pairs, which include both the source and target texts:

i. We prepare source texts, target texts, and background images from SynText [3].

ii. We randomize the augmentation parameters such as font type & size, text geometry, color augmentation, and text effects as shown in Table. 1 as well as the desired image resolution.

iii. Both texts are rendered from the text space to the image space using the Pygame [1] library, and we apply various augmentations such as font type, font size, curve text rate, underline text rate, strong text rate, and oblique rate. During this step, we apply a standard color to the text, such as black.

iv. The rendered text is additionally augmented with geometry augmentation, such as zoom rate, shear rate, rotation rate, and perspective rate, to supplement the text location and perspective.

v. The background is randomly cropped according to the desired resolution, and then several augmentations are applied, including brightness, contrast, color, and additional padding adjustments.

vi. The rendered text and pre-processed background are then integrated. During this process, we apply text effect augmentation *e.g.*, shadow effects and text colors. Lastly, we combine text and background using the Poisson blending algorithm [8].

We slightly modified the augmentation parameters for the evaluation data by setting the minimum size to 128 for each width and height to ensure high-quality images. We also added more font types, such as DecaySans and Chickenic. Finally, we adjusted the contrast and brightness values to validate that the recognition model could recognize the text correctly.

3. Additional Experiments

In this section, we provide more analysis to ablate our framework using one-shot style adaptation. Moreover, we also show the trade-off analysis between style preservation and text editing. We present more qualitative results on various datasets *e.g.*, ICDAR2013 [5], ICDAR2015 [4], COCO-Text [11], SVT [9], IIIT5K [7], HierText [6], and Youtube videos.

One-shot style adaptation. In Table. 2, we show the importance of one-shot style adaptation. We use the diffusion model ϵ_θ fine-tuned on our created synthetic scene texts. Although the OCR score is slightly lower, the image quality score significantly drops. Without one-shot style adaptation, the scores plummet 1.47, 0.15, and 0.25 for PSNR, SSIM, and LPIPS scores, respectively, compared to our complete framework. It is clearly shown in Fig. 1 where the style differs entirely from the source image. Even though

Types	Apply	Aug. Method	Values						
			Min	Max	Rate	Scale	Grid size	Mag.	Desc.
Color	Background	Color	0.7	1.3	0.8	-	-	-	-
		Brightness	0.7	1.5	0.8	-	-	-	-
		Contrast	0.7	1.3	0.3	-	-	-	-
		Resolution	64	-	-	-	-	-	-
	Text	Color	0.7	1.3	0.8	-	-	-	-
Font	Text	Size	25	60	-	-	-	-	-
		Type	-	-	-	-	-	-	Arial & OpenSans
		Underline	-	-	0.01	-	-	-	-
		Strong	-	-	0.07	-	-	-	-
		Oblique	-	-	0.02	-	-	-	-
Geometry	Text	Zoom	-	-	-	0.1	-	-	-
		Rotate	-	-	-	1	-	-	-
		Shear	-	-	-	2	-	-	-
		Perspective	-	-	-	0.0005	-	-	-
		Elastic	-	-	0.001	-	4	2	-
		Curve	-	-	-	0.05	0.1	-	-
Effects	Text	Border	-	-	0.02	-	-	-	-
		Shadow	-	-	0.02	-	-	-	-

Table 1. The list of augmentations for generating the SynText dataset. ('Types': type of augmentation, 'Apply': target augmentation, 'Aug. Method': specific augmentation method, 'Min': minimum value, 'Min': maximum value, 'Rate': the possibility the augmentation is applied, 'Scale': the scale value, 'Grid size': the grid size for elastic augmentation, 'Mag': the magnitude value, and 'Desc.': is the description.

One-shot	Text	PSNR (↑)	SSIM (↑)	LPIPS (↓)	OCR Acc. (% ↑)	
					char	word
✗	✗	28.62	0.39	0.54	92.50	77.50
✗	✓	28.67	0.37	0.55	**96.79**	**90.50**
✓	✗	**30.09**	**0.54**	0.30	84.25	60.16
✓	✓	**30.09**	**0.54**	**0.29**	94.58	84.83

Table 2. Ablation studies on the SynText dataset. The best score is denoted by **bold** text. The column 'One-shot' and 'Text' denote the utilization of one-shot style adaptation method and text recognition guidance, respectively.

we apply text recognition guidance, it only revises the text content, not the overall style.

Preserving style trade-off. We observe the trade-off between preserving style and editability in Table 2. The first row represents the results without a one-shot step. The OCR score is high, but the image assessment score is significantly lower. In contrast, the third row shows a considerable increase in image assessment when we apply the one-shot approach, but at the cost of a considerably lower OCR score.

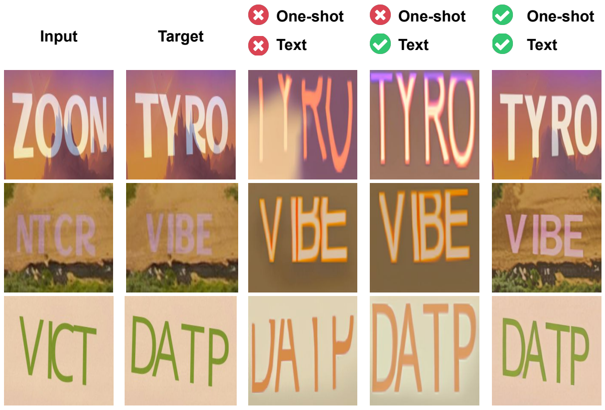

Figure 1. Ablation result of our method on Syntext dataset. The label 'One-shot' and 'Text' denote the utilization of SynText dataset and text recognition guidance, respectively.

Qualitative results We present additional qualitative results for our method in Fig. 2, Fig. 3 on benchmark datasets, including ICDAR2013 [5], ICDAR2015 [4], COCO-Text [11], SVT [9], and IIIT5K [7]. In Fig. 2, we add more examples where the target text is more varied in terms of length of characters. Furthermore, in Fig. 3, we show that our method is able to replace the source scene text in the original image with the edited version while preserving the visual characteristics of the image, such as its geometry and style. We also demonstrate our result on the recent scene text dataset namely HierText [6] in Fig. 4 and more real-case such as Youtube videos in Fig. 5. For the YouTube video collection, we focused on travel-related content. We extracted frames from these videos using FFmpeg and selected frames that contained scene text. Note that these collected frames posed additional challenges:(1) The quality of the frames is uncontrollable (*e.g.* it can be noisy and/or blurry due to the recording device and motion blur). (2) Since we specifically chose travel videos, the size of the scene text is typically quite small. Despite these challenges, our method successfully replaces the source scene text while preserving the characteristics, as demonstrated in Fig. 5.

4. Societal Impact

Our project aims to solve scene text manipulation for text translation and obscure sensitive information. However, we recognize that our work may also have unintended consequences, such as creating and disseminating false information. To address this issue, we are developing a fake scene text detection system that can help identify and prevent the creation of fake documents that use manipulated text.

References

[1] Pygame library. https://www.pygame.org/news. 1

[2] Shancheng Fang, Hongtao Xie, Yuxin Wang, Zhendong Mao, and Yongdong Zhang. Read like humans: Autonomous, bidirectional and iterative language modeling for scene text recognition. In *CVPR*, pages 7098–7107, 2021. 1

[3] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *CVPR*, pages 2315–2324, 2016. 1

[4] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman K. Ghosh, Andrew D. Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, Faisal Shafait, Seiichi Uchida, and Ernest Valveny. ICDAR 2015 robust reading competition. In *ICDAR*, pages 1156–1160, 2015. 1, 2, 4

[5] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernández Mota, Jon Almazán, and Lluís-Pere de las Heras. ICDAR 2013 robust reading competition. In *ICDAR*, pages 1484–1493, 2013. 1, 2, 4

[6] Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. Towards end-to-end unified scene text detection and layout analysis. In *CVPR*, pages 1039–1049, 2022. 1, 3, 5

[7] Anand Mishra, Karteek Alahari, and C. V. Jawahar. Top-down and bottom-up cues for scene text recognition. In *CVPR*, pages 2687–2694, 2012. 1, 2

[8] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM Trans. Graph.*, 22:313–318, 2003. 1

[9] Trung Quy Phan, Palaiahnakote Shivakumara, Shangxuan Tian, and Chew Lim Tan. Recognizing text with perspective distortion in natural scenes. In *ICCV*, pages 569–576, 2013. 1, 2, 4

[10] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10674–10685, 2022. 1

[11] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge J. Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. In *ArXiv*, 2016. 1, 2, 4

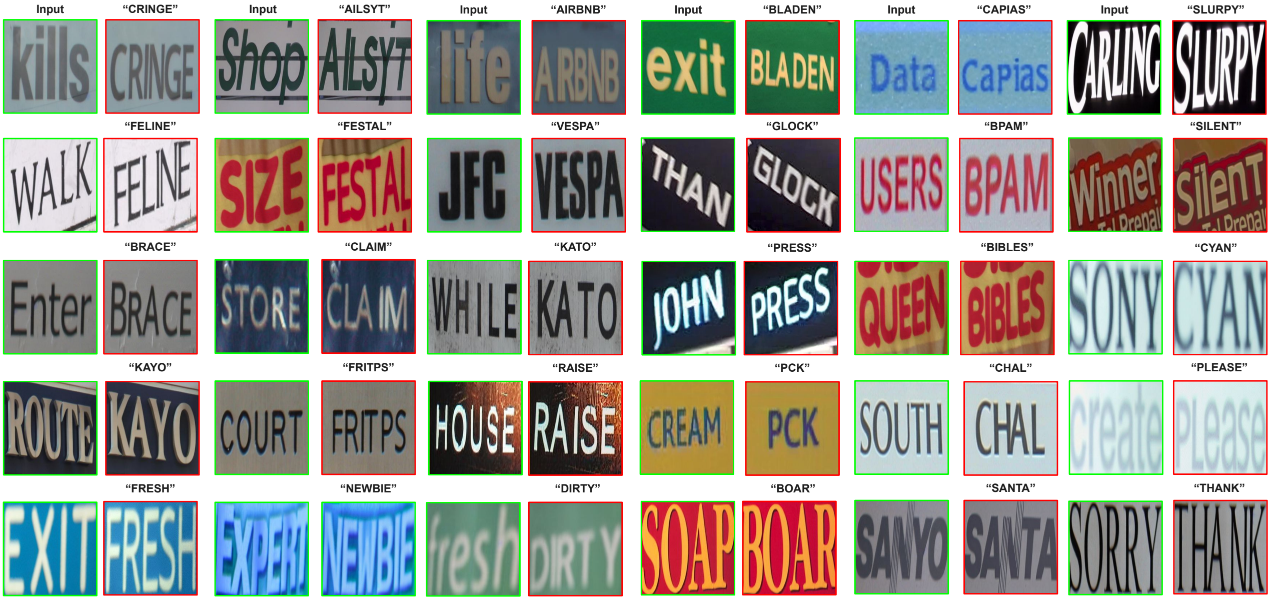

Figure 2. Given a single word *green box* from in-the-wild images and the desired text, our method successfully edits the text with the desired text in the image *red box* in mixed datasets such as ICDAR2013 [5], ICDAR2015 [4], COCO-Text [11], and SVT [9].

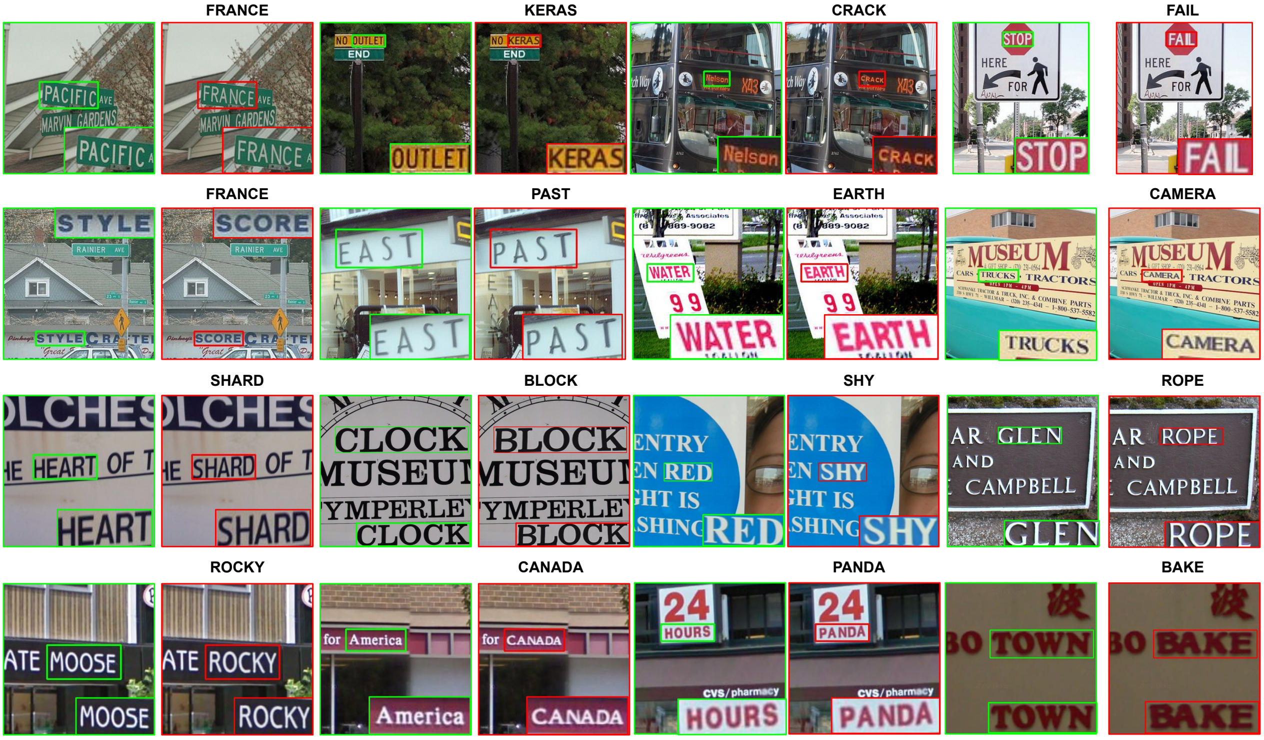

Figure 3. Given a cropped single word from in-the-wild images using a bounding box marked by a *green box*, and specifying the desired text, our method successfully edits the text to match the desired text and can replace the original word in the original image, as shown by *red box*.

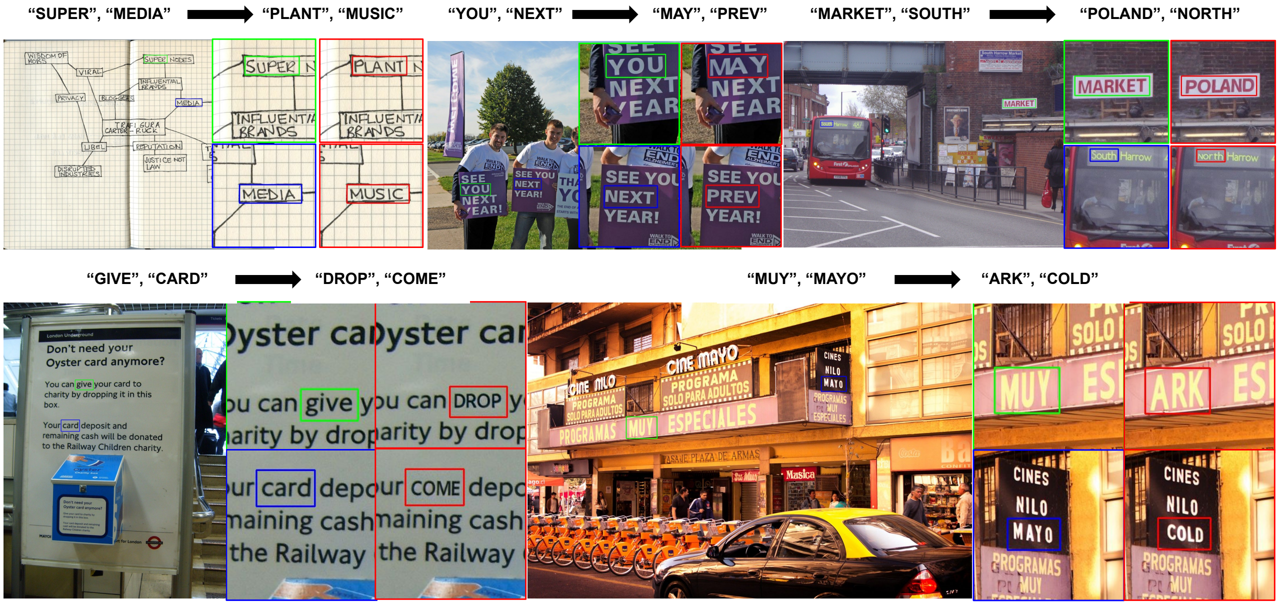

Figure 4. Given a cropped single word from HierTextt [6] images using a bounding box marked by *green box* and *blue box*, and specifying the desired text, our method successfully edits the text to match the desired text and can replace the original word in the original image, as shown by *red box*.

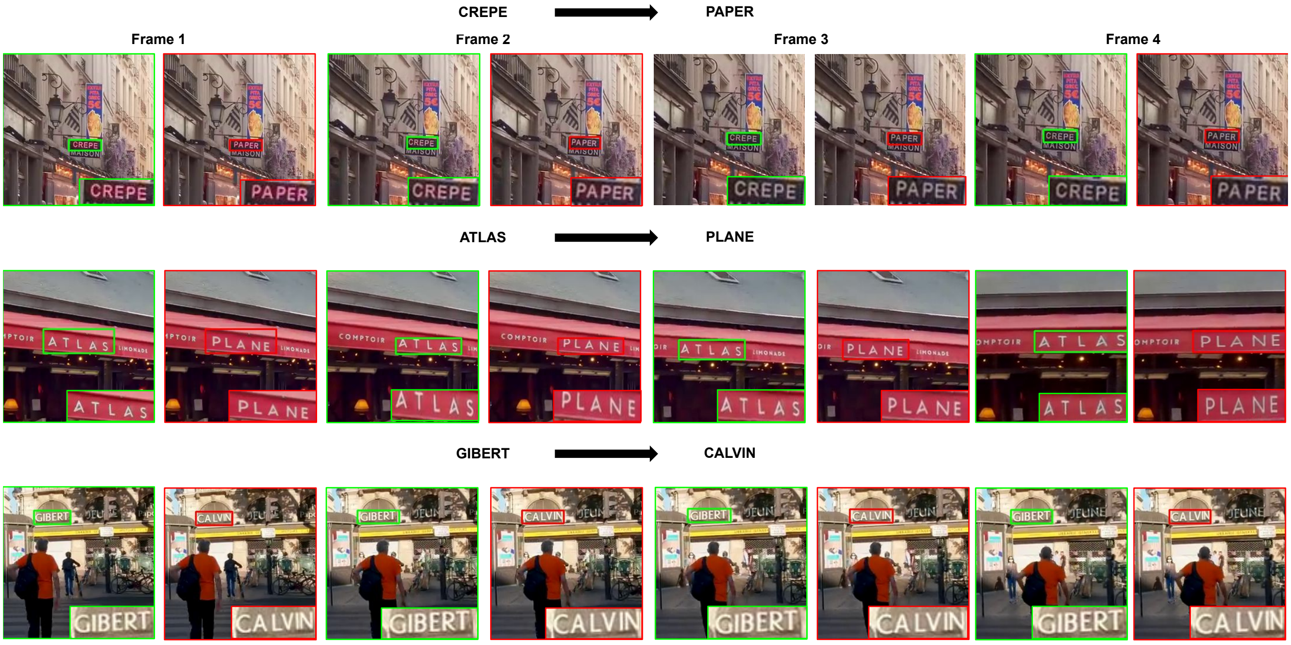

Figure 5. Given a cropped single word from Youtube frames using a bounding box marked by a *green box*, and specifying the desired text, our method successfully edits the text to match the desired text and can replace the original word in the original image, as shown by *red box*.