StyleTextGen: Style-Conditioned Multilingual Scene Text Generation

Supplementary Material

1. Text-Oriented Visual Encoder Pretraining

Our Text-Oriented Visual Encoder is pretrained with a style-preserving text segmentation objective. Instead of predicting a binary foreground mask, the pretraining task requires reconstructing the RGB appearance of the text regions while suppressing the background. This forces the encoder to focus not only on the presence of text, but also on local visual styles such as font, color, and stroke width.

We use an InternViT[1] backbone as our visual encoder, initialized from the publicly released TokenFD[2] vision encoder weights. Building on this encoder, we attach a lightweight segmentation head composed of residual self-attention blocks followed by convolution layers to produce the style-preserving text segmentation outputs.

The original TokenFD vision encoder is pretrained on the large-scale bilingual (Chinese–English) TokenIT dataset with a token-level image-as-text alignment objective: for each BPE token, the pooled visual feature within its text-region mask is aligned with the embedding of the same token in the text branch. This dense supervision over bilingual text produces features that are text-sensitive, making the encoder pretrained with TokenFD well suited to our style-preserving text segmentation setting.

We pretrain the Text-Oriented Visual Encoder on the public real-world English text segmentation dataset TextSeg, as well as on our synthetic bilingual style dataset. Several representative examples from our synthetic dataset are shown in Figure 3.

2. Details of Inference-time Style Injection Pipeline

This section provides additional details of the full inference pipeline when applying inference-time style injection. We restate the attention-based style injection in a self-contained form and summarize the overall sampling process in Alg. 1.

Algorithm 1 Inference-time Style Injection Pipeline

Input: style image I_{style}, style text P_s, target text P, initial noise z_T, injection steps T_{inj}.

Output: stylized image $\hat{I}_{\text{out}}$ corresponding to P and I_{style}.

$z_T^{(1)} \leftarrow z_T$;
for $t = T, T-1, \ldots, 1$ **do**
  $\epsilon_t^{(1)} \leftarrow \epsilon_\theta(z_t^{(1)}, P, I_{\text{style}}, t)$;
  $z_{t-1}^{(1)} \leftarrow \text{Sample}(z_t^{(1)}, \epsilon_t^{(1)})$;
end for
$\hat{I}_{\text{gen}} \leftarrow \text{Decode}(z_0^{(1)})$;
$M_{\text{gen}} \leftarrow \text{Seg}(\hat{I}_{\text{gen}})$; $M_{\text{style}} \leftarrow \text{Seg}(I_{\text{style}})$;
$z_{\text{style}} \leftarrow \mathcal{E}_{\text{style}}(I_{\text{style}})$;
$K_s, V_s \leftarrow \text{Invert}(z_{\text{style}}, P_s, M_{\text{style}})$;
$z_T^{(2)} \leftarrow z_T$; {reuse the same initial noise}
for $t = T, T-1, \ldots, 1$ **do**
  $\epsilon_t, \{Q, K, V\} \leftarrow \epsilon_\theta(z_t^{(2)}, P, I_{\text{style}}, t)$;
  if $t \le T_{\text{inj}}$ **then**
    $f_t \leftarrow \text{StyleInject}(Q, K, V,\ K_s, V_s,\ M_{\text{gen}}, M_{\text{style}})$;
    {Eqs. (2)–(5), updating the self-attention output}
    $\epsilon_t \leftarrow \epsilon_\theta(z_t^{(2)}, P, I_{\text{style}}, t;\ f_t)$;
  end if
  $z_{t-1}^{(2)} \leftarrow \text{Sample}(z_t^{(2)}, \epsilon_t)$;
end for
$\hat{I}_{\text{out}} \leftarrow \text{Decode}(z_0^{(2)})$;

Return $\hat{I}_{\text{out}}$

Style Injection. Given a generated image $\hat{I}_{\text{gen}}$ and a style reference image I_{style}, we employ a bilingual text segmentation model to obtain text-region masks M_{gen} and M_{style} for the generated and style images, respectively. These masks localize the subsequent feature modulation to text regions.

We first extract a style latent from the reference image I_{style} using the dual-branch style encoder $\mathcal{E}_{\text{style}}$ in our framework:

$$z_{\text{style}} = \mathcal{E}_{\text{style}}(I_{\text{style}}). \tag{1}$$

We then invert z_{style} through the DiT backbone using a flow-matching-based inversion [5, 6] to recover internal key–value tensors (K_s, V_s), where the mask M_{style} ensures that only text regions contribute to the extracted style representation.

During generation, let Q, K, V denote the query, key and value tensors of a self-attention layer. We first obtain style-masked tensors and apply AdaIN[3] to derive style-adapted keys and values:

$$\tilde{K}, \tilde{V} = \text{AdaIN}\left(K, V;\ K_s \odot M_{\text{style}}, V_s \odot M_{\text{style}}\right). \tag{2}$$

and then blend them with the original tensors inside the target text regions:

$$\begin{aligned} K' &= (1 - M_{\text{gen}}) \odot K + M_{\text{gen}} \odot \tilde{K}, \\ V' &= (1 - M_{\text{gen}}) \odot V + M_{\text{gen}} \odot \tilde{V}. \end{aligned} \tag{3}$$

Under these updated tensors, we compute the base and style attention responses as

$$\begin{aligned} f_{\text{base}} &= \text{Attention}(Q, K', V'), \\ f_{\text{style}} &= \text{Attention}(Q, K_s, V_s), \end{aligned} \tag{4}$$

and obtain the final output via mask-guided blending:

$$f_{\text{out}} = (1 - M_{\text{gen}}) \odot f_{\text{base}} + M_{\text{gen}} \odot \text{AdaIN}\left(f_{\text{base}};\ f_{\text{style}} \odot M_{\text{style}}\right). \tag{5}$$

We apply this modulation only during the first T_{inj} denoising steps (T_{inj}=10 in all experiments) to preserve scene-text fidelity and background coherence.

Inference pipeline. Let P be the target text prompt, I_{style} the style reference image, and P_s the style text associated with I_{style}. We denote by ϵ_θ the DiT noise-prediction model and by `Sample` the flow-matching sampler update. The complete inference pipeline with style injection is summarized in Alg. 1.

3. More Results

3.1. Ablation Study on the Text Style Consistency Loss Weight λ_{tsc}

We further study the impact of the loss weight λ_{tsc} of the Text Style Consistency Loss. To isolate the effect of this loss term, we disable the inference-time style injection and vary λ_{tsc} while keeping all other settings fixed. The results are summarized in Tab. 1.

When $\lambda_{\text{tsc}} = 0$, the model reduces to a conditional flow-matching baseline, leading to poorer recognition and perceptual quality. Setting $\lambda_{\text{tsc}} = 1$ brings negligible gains. In contrast, increasing it to 5 and 10 consistently improves Sen.Acc and NED while also reducing FID and LPIPS. Although $\mathcal{L}_{\text{tsc}}$ is introduced as a style loss, we also observe improvements in text accuracy metrics. We attribute this to the fact that enforcing text style consistency encourages the model to decouple text regions from the background and to generate more stable character contours, which in turn benefits recognition. The best trade-off is achieved at $\lambda_{\text{tsc}} = 10$, which we adopt as the default. Increasing the weight to $\lambda_{\text{tsc}} = 20$ again degrades the metrics, indicating that overly strong style supervision harms generation fidelity.

3.2. Qualitative Results of Bilingual Text Segmentation Model

Hi-SAM [7] is a hierarchical text segmentation model built on SAM [4]. It consists of two components. The first component, SAM-TS, uses the image encoder from SAM together with a self-prompting module and a decoder to generate pixel-level text masks. The second component takes these masks as input and groups them into segmentation results at the levels of words, text lines, and paragraphs.

Since Hi-SAM is only trained on English text segmentation datasets, it shows limited performance when directly applied to Chinese text segmentation. To obtain a model that can handle both English and Chinese, we follow the original configuration of SAM-TS, initialize it with the released weights, and fine-tune it on our synthetic bilingual dataset. The resulting model is referred to as our Bilingual Text Segmentation Model, which is used in our framework both to compute the Text Style Consistency Loss and to evaluate the StyleText-CE metric. Figure 1 presents qualitative comparisons between the original SAM-TS and our bilingual model on Chinese text images. The original SAM-TS often produces incomplete masks for Chinese characters. In contrast, our fine-tuned model yields more accurate and continuous masks for Chinese text regions.

Table 1. Ablation study on the loss weight λ_{tsc} of the Text Style Consistency Loss. The best result in each column is highlighted in bold.

λ_{tsc}	Sen.Acc ↑	NED ↑	FID ↓	LPIPS ↓
0	0.625	0.785	129.00	0.510
1	0.627	0.786	128.45	0.509
5	0.664	0.803	116.37	0.496
10	**0.669**	**0.807**	**115.21**	**0.493**
20	0.619	0.787	127.92	0.511

Figure 1. Qualitative comparison of Chinese text segmentation. From left to right: source images, segmentation masks predicted by SAM-TS, and masks produced by our bilingual text segmentation model.

3.3. Additional Style Similarity Evaluation

In addition to the segmentation-based evaluation reported in the main paper, we further perform style similarity evaluation using cropped text bounding boxes. Specifically, text regions are cropped according to the annotated bounding boxes, and FID together with GPT-Score are computed on the cropped regions. FID is computed between deep features of generated and reference text regions, while GPT-Score is obtained by prompting ChatGPT to assess style similarity on a 5-point scale.

This additional evaluation avoids reliance on the text segmentation model and provides a robust assessment of style similarity. As shown in Table 2, our method consistently outperforms Calligrapher across all settings, further supporting the conclusions in the main paper.

Table 2. Style similarity evaluation computed on text bounding boxes (FID↓/GPT-Score↑).

Model	Self Style Reference		External Style Reference	
	Mono	Cross	Mono	Cross
Callig.	108.4/3.85	172.6/3.59	153.9/3.62	181.2/3.27
Ours	**97.8/4.12**	**156.1/3.88**	**132.5/4.01**	**158.9/3.74**

4. Discussion and Comparison with Frontier Models

To better contextualize StyleTextGen among recent generative models, we provide additional qualitative comparisons with Nano Bananan and GPT-Image in Fig. 2. These examples illustrate the distinction between general-purpose image generators and a model designed specifically for style-conditioned scene text synthesis.

In the first example, our method captures the reference font style more faithfully. While the other models generate visually coherent text, they fail to reproduce the characteristic stroke shapes and typographic details present in the reference. In contrast, StyleTextGen preserves the font appearance accurately. The second example further demonstrates that our method transfers color and rendering style more effectively. The generated text aligns more closely with the reference in both hue composition and overall visual appearance, yielding stronger style consistency.

While state-of-the-art image generation models exhibit strong general-purpose visual synthesis capabilities, they are not specifically tailored for granular control over scene text style. In contrast, StyleTextGen explicitly models text-specific style features and enforces style consistency, thereby offering more reliable control over the appearance of rendered text.

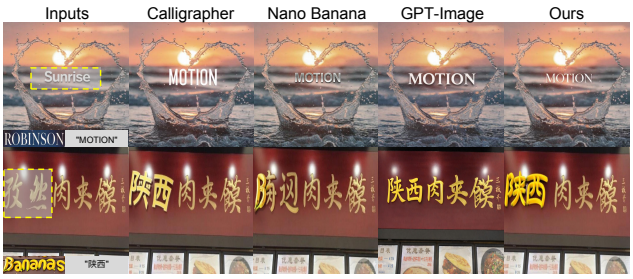

Figure 2. Qualitative comparison with frontier models.

References

[1] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 24185–24198, 2024. 1

[2] Tongkun Guan, Zining Wang, Pei Fu, Zhengtao Guo, Wei Shen, Kai Zhou, Tiezhu Yue, Chen Duan, Hao Sun, Qianyi Jiang, et al. A token-level text image foundation model for document understanding. *arXiv preprint arXiv:2503.02304*, 2025. 1

[3] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017. 1

[4] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 2

[5] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 2023. 1

[6] Litu Rout, Yujia Chen, Nataniel Ruiz, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Semantic image inversion and editing using rectified stochastic differential equations. *arXiv preprint arXiv:2410.10792*, 2024. 1

[7] Maoyuan Ye, Jing Zhang, Juhua Liu, Chenyu Liu, Baocai Yin, Cong Liu, Bo Du, and Dacheng Tao. Hi-sam: Marrying segment anything model for hierarchical text segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–16, 2024. 2

Figure 3. Examples of synthetic data used for pretraining our Text-Oriented Visual Encoder. For each example, we show the input text image (Input) and the corresponding style-preserving text segmentation ground truth (GT).