TEXT-BASED VECTOR SKETCH EDITING WITH IMAGE EDITING DIFFUSION PRIOR

Haoran Mo, Xusheng Lin, Chengying Gao[*] and Ruomei Wang

Sun Yat-sen University
{mohaor,linxsh8}@mail2.sysu.edu.cn, {mcsgcy,isswrm}@mail.sysu.edu.cn

ABSTRACT

We present a framework for text-based vector sketch editing to improve the efficiency of graphic design. The key idea behind the approach is to transfer the prior information from raster-level diffusion models, especially those from image editing methods, into the vector sketch-oriented task. The framework presents three editing modes and allows iterative editing. To meet the editing requirement of modifying the intended parts only while avoiding changing the other strokes, we introduce a stroke-level local editing scheme that automatically produces an editing mask reflecting locally editable regions and modifies strokes within the regions only. Comparisons with existing methods demonstrate the superiority of our approach.

Index Terms— vector sketch, image editing, diffusion model

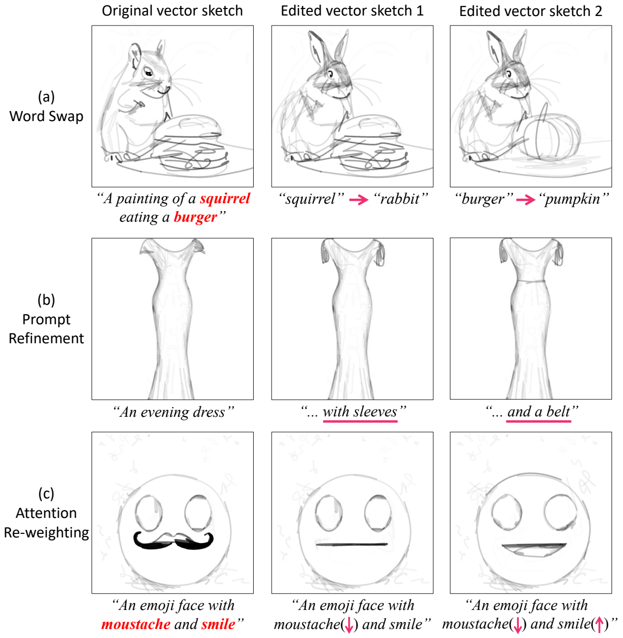
Original vector sketch
Edited vector sketch 1
Edited vector sketch 2
(a) Word Swap
"A painting of a squirrel eating a burger"
"squirrel" → "rabbit"
"burger" → "pumpkin"
(b) Prompt Refinement
"An evening dress"
"... with sleeves"
"... and a belt"
(c) Attention Re-weighting
"An emoji face with moustache and smile"
"An emoji face with moustache(↓) and smile"
"An emoji face with moustache(↓) and smile(↑)"

Fig. 1. Our text-based vector sketch editing framework presents three modes of controlling the edits. It also allows iterative editing.

1. INTRODUCTION

Textual prompts have emerged to be the most popular and intuitive interactive medium recently due to their convenience and user-friendliness. Editing of vector line drawings plays a fundamental role in graphic design [1], while it is largely limited to manual expert workflows where considerable user efforts are required. In this work, we combine the two together for text-based vector graphics editing to improve the editing efficiency. While there exist several works on text-based vector graphics synthesis, the editing is relatively under-explored as it is more challenging to realize the editing intentions, *i.e.,* identifying and modifying target graphics following the prompts. We make the first attempt by focusing on vector sketches in a free-hand style, and aim at a scenario for making creative and fast design where users first generate a vector sketch from an original prompt, and then intend to edit the result by modifying the prompt, such as adding refinement or changing the contents. Examples in Fig. 1 show this process.

Pre-trained large models, *e.g.,* diffusion models (DMs) [2], are broadly employed in text-based tasks, because such models trained with giant-scale datasets provide good prior knowledge for various generative scenarios. However, such kind of pre-trained models are still rare for vector graphics, due to the lack of large-scale datasets with cross-modality annotations (*e.g.,* text). To this end, we propose to transfer *prior knowledge* from raster-level pre-trained DMs, especially those from image editing methods, into our text-based vector sketch editing task. This alleviates the need for ample text-vector sketch pairs for training. We introduce a simple yet effective method to combine a text-based image editing technique with vector sketch synthesis approaches, by utilizing the DM-based edited images as guidance when generating the edited sketches. The powerful prior information helps to produce high-quality vector sketches.

As shown in Fig. 1, we present three editing modes: (a) *Word Swap* that swaps the words in original prompt, (b) *Prompt Refinement* that adds new words to the prompt, and

[*] The corresponding author is Chengying Gao. This work was supported by the Guangdong Basic and Applied Basic Research Foundation (Grant No.2022A1515011425).

(c) *Attention Re-weighting* that strengthens or weakens some words (*e.g.,* less "moustache"). Furthermore, our approach allows iterative editing by sequentially changing the prompts.

The editing requirement of vector line drawing is that only intended regions should be modified while preserving the others, which is crucial in graphic design. To address this problem, we introduce a *stroke-level local editing scheme*, which automatically extracts an editing mask from the pre-trained DM according to the original and edited prompts. The mask correctly reflects the locally editable regions, and constrains the edits to be applied only to strokes within those regions. Thus, the non-target strokes remain unchanged.

We evaluate our approach through qualitative and quantitative comparisons with text-based vector graphics synthesis algorithms and a text-based image editing method. The results corroborate the superior performance of our framework in producing visually appealing vector sketches conforming to the editing prompts while preserving the unedited parts[1].

The main contributions are summarized as follows:

- A text-based vector sketch editing framework using prior knowledge from a raster-level image editing diffusion model. It presents three modes of controlling the edits and allows iterative editing.
- A stroke-level local editing scheme that remains vector strokes in the unedited regions unchanged while modifying the rest strokes following the prompts.
- Comprehensive comparisons with existing approaches that demonstrate the effectiveness of our approach.

2. RELATED WORK

2.1. Text-based Vector Sketch Synthesis and Editing

The rapid development of diffusion models has induced a large body of text-based image synthesis methods [2]. Given different representations between raster and vector images, some works bridge the gap through a differentiable renderer [3] that converts vector sketches into bitmaps. This allows direct optimization of the parametrized sketches in a raster-level supervision manner. CLIPDraw [4] optimizes the stroke parameters based on cosine distance in CLIP space [5] between the rendered sketch and the input text. VectorFusion [6] inputs the rendered image and the text to latent diffusion model (LDM) [2] for a score distillation sampling (SDS) loss reflecting text-image alignment. Based on the SDS loss, DiffSketcher [7] further uses the LDM-generated image and the rendered sketch to calculate a semantic and perceptual loss to improve visual quality of the sketches. These methods address the problem of text-based vector image synthesis, while struggling with editing task as they tend to fail in reproducing the unedited parts of the original vector images. In contrast, our approach with the proposed stroke-level local editing scheme is able to modify the intended contents while remaining the other strokes unchanged.

Text-based vector graphics editing is a relatively under-explored area with limited researches. IconShop [8] employs an autoregressive Transformer for vector icon editing and relies on a large-scale dataset with text annotations for training. It is not suitable to our task due to the shortage of abundant text-vector sketch pairs. In comparison, our method with the pre-trained diffusion model as a prior is independent of such a training dataset. SVGCustomization [9] uses a customized text-to-image diffusion model for raster-level editing, followed by a vectorization process to generate vector graphics. It relies heavily on color information for shape alignment, which is not applicable to monochromatic sketches.

2.2. Text-based Image Editing

An innate property of this task is to change the contents specified by the editing prompt while preserving the unedited parts. Some works require users to draw an input mask to specify the editable regions [10], while the others automatically identity editable areas to reduce the user effort [11, 12, 13]. Prompt-to-Prompt (P2P) [11] uses cross-attention maps from LDM [2] which indicate the attended regions of each word token. During editing, the cross-attention maps are injected according to the difference between original and editing prompts, so as to produce edited images with local changes. Pix2pix-zero [12] adopts cross-attention maps similarly, but uses them as an explicit supervision for training. DiffEdit [13] proposes to predict an editing mask based on difference between two predicted noises from the original and editing prompts. While the editing approaches above are designed for raster images, we exploit the prior information from their models and transfer into the vector sketch editing task.

3. METHOD

3.1. Preliminary

We integrate the idea of a text-based image editing approach Prompt-to-Prompt (P2P) [11] built on a pre-trained latent diffusion model (*a.k.a.* Stable Diffusion) [2] into our framework, so we first provide a concise overview. Given a textual prompt P and a seed, P2P generates an image I first and then an edited image I^* according to an edited prompt P^*. To meet the editing requirement of preserving structure and contents with respect to the source image I, P2P exploits cross-attention maps which have been shown to control the composition of the images from the diffusion model.

The latent diffusion model employs a U-Net to predict a noise ϵ from a noisy image z_t and text embedding $\psi\left(P\right)$ at each diffusion step t, and the two modalities are fused in cross-attention layers. Formally, three linear projections ℓ_Q, ℓ_K, ℓ_V are utilized to project feature maps of the noisy image $\phi\left(z_t\right)$ and $\psi\left(P\right)$ to a query matrix $Q = \ell_Q\left(\phi\left(z_t\right)\right)$, a key

[1] The source code can be found at https://github.com/MarkMoHR/DiffSketchEdit.

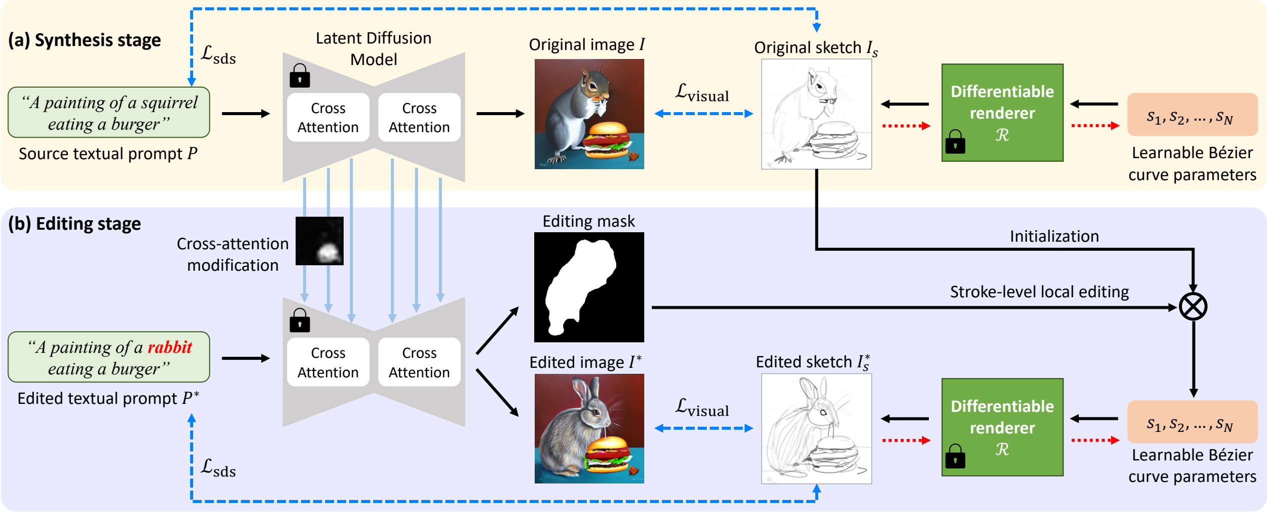
(a) Synthesis stage
$\mathcal{L}_{\text{sds}}$
"A painting of a squirrel eating a burger"
Source textual prompt P
Latent Diffusion Model
Cross Attention
Cross Attention
Original image I
$\mathcal{L}_{\text{visual}}$
Original sketch I_s
Differentiable renderer $\mathcal{R}$
$s_1, s_2, ..., s_N$
Learnable Bézier curve parameters
(b) Editing stage
Cross-attention modification
"A painting of a rabbit eating a burger"
Edited textual prompt P^*
$\mathcal{L}_{\text{sds}}$
Cross Attention
Cross Attention
Editing mask
Edited image I^*
$\mathcal{L}_{\text{visual}}$
Edited sketch I_s^*
Differentiable renderer $\mathcal{R}$
$s_1, s_2, ..., s_N$
Learnable Bézier curve parameters
Initialization
Stroke-level local editing

Fig. 2. Our framework for text-based vector sketch editing incorporates the prior knowledge from a pre-trained text-based image editing diffusion model. The edited image is used in a visual loss for optimizing the vector stroke parameters. Blue arrows mean the losses and red arrows indicate the gradient propagation direction.

matrix $K = \ell_K(\psi(P))$ and a value matrix $V = \ell_V(\psi(P))$, respectively. The cross-attention maps are calculated as $M = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)$, where M_i is an attention map reflecting the attended image region of the i-th token.

Based on this observation, P2P proposes to inject the cross-attention maps of the source image I into the generation process for I^* with the edited prompt P^* to maintain the image structure. Three controlling modes are introduced: (1) **Word Swap**, which means swapping tokens in P with others to form P^*. The cross-attention maps of all tokens from P^* in the generation of I^* are directly replaced with those from P. (2) **Prompt Refinement**, which allows adding new tokens to P. It replaces the attention maps of common tokens only, and maintains those for I^* corresponding to new tokens in P^*. (3) **Attention Re-weighting**, which strengthens or weakens the effect of some tokens. The cross-attention maps of specified tokens are scaled by a re-weight parameter.

3.2. Text-based Vector Sketch Editing Framework

The framework allows users to edit a prompt-generated vector sketch quickly by modifying the original textual prompt. As shown in Fig 1, we present three editing modes akin to Prompt-to-Prompt (P2P) [11], namely Word Swap, Prompt Refinement and Attention Re-weighting. The key idea behind our approach is to transfer powerful *prior knowledge* from text-based image editing models into the editing of vector sketches. Formally, the original and edited images from a pre-trained latent diffusion model are used as visual supervision for the synthesized sketches, which promotes their visual quality and consistency to the prompts.

Our framework is shown in Fig. 2, which consists of an initial synthesis stage based on a source textual prompt P and a subsequent editing stage with an edited prompt P^*. We represent a vector sketch with N learnable cubic Bézier curves (or strokes) $\{s_1, s_2, ..., s_N\}$. In each stage, the vector sketch is generated via direct optimization on the parameters of the strokes. Each stroke $s_i = (\{(x_i, y_i)^j\}_{j=1}^4, o_i)$ is made up of four control points $(x_i, y_i)^j$ and an opacity parameter o_i mimicking the pen pressure. A differentiable renderer Diffvg [3] renders the strokes into a raster sketch that enables calculation of raster-level losses. The derived gradient can be propagated through the renderer back to the stroke parameters.

Initial Synthesis Stage. An original vector sketch I_s is generated according to the prompt P in this stage. We employ the pipeline of DiffSketcher [7], where a pre-trained latent diffusion model (LDM) [2] is incorporated during the optimization. An original image I is produced by the LDM with prompt P, which is used to calculate a joint visual semantic and perceptual loss $\mathcal{L}_{\text{visual}}$ with the rendered sketch I_s. A score distillation sampling loss $\mathcal{L}_{\text{sds}}$ in DiffSketcher [7] that measures the text-image alignment is also adopted. The two losses are combined for the optimization of stroke parameters.

Editing Stage. We incorporate the Prompt-to-Prompt image editing method [11] into this stage to make full use of its powerful raster prior for the vector stroke-level editing task. Given the edited prompt P^* in one of the three editing modes, the latent diffusion model generates the edited image I^* with the cross-attention modification operations introduced in Section 3.1. The I^* conforms to the edited prompt while maintaining the structure and necessary contents of the original image I.

Afterwards, we use the optimized strokes in the first synthesis stage as an initialization for subsequent editing. Similar

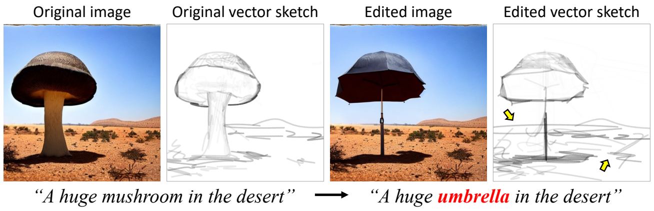

Fig. 3. An example of edit without stroke-level local editing scheme. Messy strokes are added into the background although it is not the target editing region.

to the first stage, the visual loss and the score distillation sampling loss are jointly used to optimize the stroke parameters of the edited sketch I_s^*. The edited image I^* serves as a good prior to control the visual appearance of the edited sketch.

3.3. Stroke-level Local Editing Scheme

The requirement of vector image editing is to modify the target object or region while avoiding introducing unnecessary changes in other parts. This allows the editing to be performed in a moving-forward way, which is critical in graphic design. Albeit with an edited image restoring the unedited parts as a guidance, plus the original sketch as an initialization, undesired changes of strokes in those non-target regions (*e.g.,* the background in Fig. 3) still exist. This is probably because vector sketches form sparse abstractions of the images, which can be non-unique and diverse in human perception. The optimization process using semantic and perceptual similarity as criteria suffers from uncertainty and induces variations of strokes even in non-target parts.

To overcome the issue and constrain the edits to be applied only to strokes in the intended regions, we propose a *stroke-level local editing scheme*. The main idea behind this scheme is to optimize the parameters of those strokes only, such that the others remain unchanged. We first find editable regions following the idea of pixel-level local editing in Prompt-to-Prompt [11], by automatically extracting an editing mask from the cross-attention maps in the pre-trained diffusion model, as shown in Fig. 2. We leverage the cross-attention maps that correspond to the words specifying the editing regions, and average them across all denoising steps. The averaged map is binarized with a threshold $k = 0.3$ and form the local editing mask for the target parts.

Given the original vector sketch in the first synthesis stage as an initialization, we identify the strokes lying within the local regions as the editable ones, as shown in Fig. 4. Considering that the mask is in pixel space and the strokes in vector one, we bridge the gap through the renderer Diffvg [3]. The mask is first resized to a pre-defined image size. Then, each stroke is rendered into a raster image of the same size. We treat a stroke as editable if its intersection with the editable regions is more than half its own area. The optimization is performed on the parameters of the editable strokes, resulting in an edited sketch modifying only the target regions.

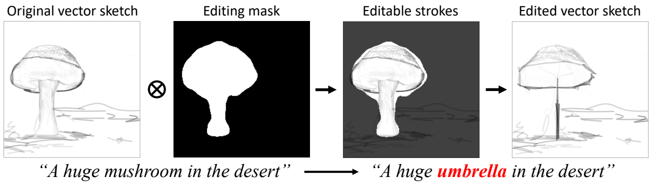

Fig. 4. Illustration of stroke-level local editing scheme with an editing mask extracted from the pre-trained latent diffusion model. The background remains unchanged then.

3.4. Iterative Editing

As shown in Fig. 1, our approach allows iterative editing to obtain an incremental effect. For example, we add "sleeves" to the dress to produce an edited result, followed by a second addition of a "belt". This is different from Prompt-to-Prompt [11] that is only able to apply multiple edits to the initially generated image. We achieve this by modifying the cross-attention maps of the prompts in a sequential manner. Please refer to the supplemental document for details.

4. EXPERIMENTS

4.1. Comparisons with Existing Approaches

Due to the lack of existing methods for text-based vector sketch editing, we compare our framework with text-based vector graphics synthesis approaches and an image-level text-based editing algorithm. For the former, CLIPDraw [4], VectorFusion [6] and DiffSketcher [7] are used as baseline methods. For fair comparisons, we utilize the generated vector sketch from the original prompt in our first stage as an initialization, and produce the edited vector sketch according to the edited prompt with these methods. Note that the comparisons are done in editing modes Word Swap and Prompt Refinement excluding Attention Re-weighting to which the baseline approaches are not applicable as the original and edited prompts share the same word tokens. In terms of the text-based image editing counterpart, we compare to Prompt-to-Prompt [11], although it produces a raster image pair that does not meet the demand of our task. To make it generate free-hand style sketches, a prefix *"a monochromatic free-hand line sketch of"* is added to the prompts.

Qualitative Results. The comparisons are shown in Fig. 5. CLIPDraw produces more abstract sketches due to its loss function based on high-level semantic similarity between texts and sketches. VectorFusion fails to work on the Prompt Refinement mode in most cases, probably due to the lack of visual supervision. DiffSketcher is able to generate sketches corresponding to the prompts, although it tends to change the

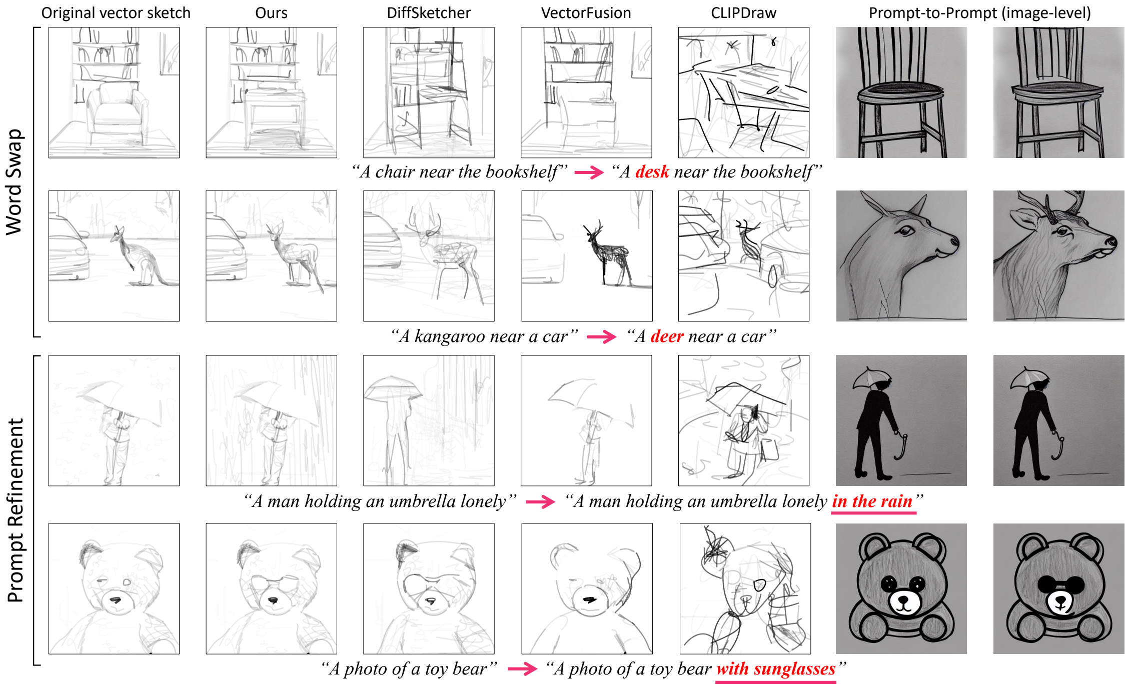
Original vector sketch
Ours
DiffSketcher
VectorFusion
CLIPDraw
Prompt-to-Prompt (image-level)
Word Swap
Prompt Refinement
"A chair near the bookshelf" → "A desk near the bookshelf"
"A kangaroo near a car" → "A deer near a car"
"A man holding an umbrella lonely" → "A man holding an umbrella lonely in the rain"
"A photo of a toy bear" → "A photo of a toy bear with sunglasses"

Fig. 5. Comparisons with baseline methods involving replacing objects, changing the background and modifying local components. Each image pair of Prompt-to-Prompt denotes the initially synthesized image (left) and the edited one (right).

layout and the non-target regions (*e.g.,* the car and the man). In contrast, with the editing prior and the proposed stroke-level local editing scheme, our results demonstrate superior quality in terms of consistency with the editing prompts and preservation of unedited parts. Prompt-to-Prompt synthesizes images in a different style, and fails to generate all objects specified in the prompts in most cases.

Our approach is also able to produce promising results in Attention Re-weighting mode (see Fig. 1 and Fig. 8). Please refer to supplemental materials for more results.

Table 1. Quantitative comparisons with baseline methods. "Text-Image" denotes Text-image consistency and "Image CLIP" indicates Image-level similarity with CLIP score. The 1^{st} and 2^{nd} best results are highlighted with bold type and underline, respectively.

	Text-Image(↑)	Image CLIP(↑)	LPIPS(↑)
CLIPDraw [4]	44.01	77.54	56.58
DiffSketcher [7]	46.30	89.39	66.51
VectorFusion [6]	**47.01**	90.52	72.35
Ours (w/o local)	46.46	<u>93.85</u>	<u>79.01</u>
Ours	<u>46.64</u>	**96.64**	**89.15**

Quantitative Results. We also quantitatively compare with the baseline methods of vector-level editing that share the same output format as ours. Three metrics are used: text-image consistency based on CLIP score [11, 7], and image-level similarity measured by LPIPS score and CLIP score [11]. The image-level similarity reflects the preservation of unedited parts in the absence of ground-truth data. We generate a large number of examples with random prompts for the Word Swap and Prompt Refinement modes, and average the scores across all the testing examples.

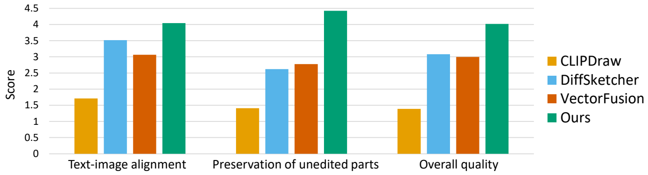

Fig. 6. Results of user study.

As shown in Table 1, CLIPDraw shows the worst performance. DiffSketcher performs worse than ours, especially in image-level similarity, indicating its weakness in preservation of unedited regions. VectorFusion is slightly better than ours in text-image consistency, probably because of its loss function designed for text-image alignment primarily. While its performance in local preservation is noticeably worse. On the whole, our approach performs the best, surpassing the baseline methods by a large margin in image-level similarity.

User Study. We additionally conduct a user study for a subjective evaluation. We invite 28 participants who have no prior knowledge of this project and assign each user 30 groups of random examples. They are asked to score 1 (worst) to 5 (best) regarding text-image alignment, preservation of

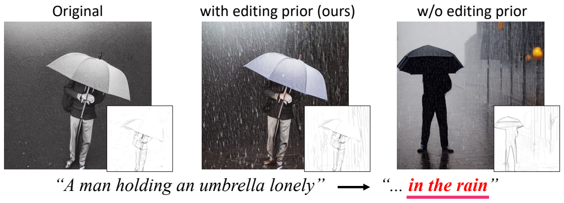
Original
with editing prior (ours)
w/o editing prior
“A man holding an umbrella lonely” → “... in the rain”

Fig. 7. Effectiveness of text-based image editing prior.

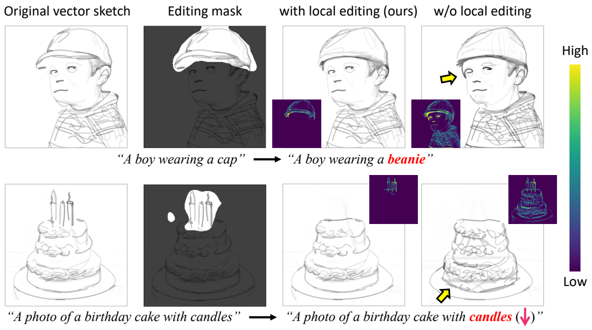
Original vector sketch
Editing mask
with local editing (ours)
w/o local editing
High
Low
“A boy wearing a cap” → “A boy wearing a beanie”
“A photo of a birthday cake with candles” → “A photo of a birthday cake with candles (↓)”

Fig. 8. Comparisons between methods with and without stroke-level local editing scheme. Sub-figures are the absolute difference from the original sketches.

unedited parts and overall quality. The results shown in Fig. 6 are in line with those in the quantitative evaluation, where our approach still outperforms the baseline methods.

4.2. Ablation Study

Image Editing Prior. In the absence of the editing prior from a raster-level diffusion model, the framework produces two images independently from the original and the edited prompts through the pre-trained latent diffusion model, which exhibit varying structure and contents albeit with the same seed, as shown in Fig. 7. Consequently, the edited sketch guided by the image has similar variations compared to the original one. On the contrary, the text-based image editing technique provides an edited image with a consistent composition and layout (*e.g.,* the man and the umbrella), serving as a feasible prior for guiding subsequent vector sketch editing.

Stroke-level Local Editing Scheme. We show quantitative results of method without this scheme in Table 1-(w/o local). Examples in Fig. 8 corroborate its effectiveness of facilitating local edits in the target regions. The framework without this scheme brings about unnecessary stroke variations in the non-target regions (see yellow arrows and absolute difference).

5. CONCLUSION

We present a text-based vector sketch editing framework that extracts prior information from an image editing method based on pre-trained latent diffusion models. Our introduced stroke-level local editing scheme identifies editable regions and constrains the edits within those regions. While showing superior performance in both text-image consistency and preservation of unintended regions, our editing approach still has a limitation that it is only applicable to prompt-generated vector sketches rather than existing ones. Using sketch-to-image translation diffusion models to form the raster prior could be considered to address the issue.

6. REFERENCES

[1] Haoran Mo, Edgar Simo-Serra, Chengying Gao, Changqing Zou, and Ruomei Wang, “General virtual sketching framework for vector line art,” *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–14, 2021.

[2] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, et al., “High-resolution image synthesis with latent diffusion models,” in *CVPR*, 2022.

[3] Tzu-Mao Li, Michal Lukáč, Michaël Gharbi, and Jonathan Ragan-Kelley, “Differentiable vector graphics rasterization for editing and learning,” *ACM Transactions on Graphics (TOG)*, 2020.

[4] Kevin Frans, Lisa Soros, and Olaf Witkowski, “Clipdraw: Exploring text-to-drawing synthesis through language-image encoders,” *NeurIPS*, 2022.

[5] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, et al., “Learning transferable visual models from natural language supervision,” in *ICML*, 2021.

[6] Ajay Jain, Amber Xie, and Pieter Abbeel, “Vectorfusion: Text-to-svg by abstracting pixel-based diffusion models,” in *CVPR*, 2023.

[7] Ximing Xing, Chuang Wang, Haitao Zhou, Jing Zhang, et al., “Diffsketcher: Text guided vector sketch synthesis through latent diffusion models,” *NeurIPS*, 2023.

[8] Ronghuan Wu, Wanchao Su, Kede Ma, and Jing Liao, “Iconshop: Text-based vector icon synthesis with autoregressive transformers,” *ACM Transactions on Graphics (TOG)*, 2023.

[9] Peiying Zhang, Nanxuan Zhao, and Jing Liao, “Text-guided vector graphics customization,” in *SIGGRAPH Asia 2023 Conference Papers*, 2023.

[10] Omri Avrahami, Dani Lischinski, and Ohad Fried, “Blended diffusion for text-driven editing of natural images,” in *CVPR*, 2022.

[11] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, et al., “Prompt-to-prompt image editing with cross-attention control,” in *ICLR*, 2023.

[12] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu, “Zero-shot image-to-image translation,” in *ACM SIGGRAPH 2023 Conference Proceedings*, 2023.

[13] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord, “Diffedit: Diffusion-based semantic image editing with mask guidance,” in *ICLR*, 2023.