FireEdit: Fine-grained Instruction-based Image Editing via Region-aware Vision Language Model

Supplementary Material

In supplementary material, we provide the following contents:

- Implementation details of FireEdit;
- More visual results on Emu Edit test set;
- Additional ablation studies
- Human evaluation;
- Societal impact.

1. Implementation Details of FireEdit

Training Protocol. Our complete training process is divided into three stages. In the first stage, we train the Q-Former module to align the output of the Vision-Language Model (VLM) with the CLIP text encoder. In the second stage, we use the pre-trained DINOv2 [12] as the image encoder for the VLM. Region proposals are derived using Deformable DETR [20], and ROIAlign [13] is used as the region encoder, with weights loaded from [11]. The parameters of DINOv2, Deformable DETR, and ROIAlign are frozen. We only train an additional Adapter, which maps visual features to the text space of the Large Language Model (LLM). To ensure efficient training, the parameters of the LLM are also frozen, and we use LoRA [4] to fine-tune it. The editing representations aligned by Q-Former are directly used to guide the learning of the diffusion model. In this stage, the parameters to be trained include the Adapter, LoRA, Q-Former, and UNet [14] in the diffusion model. In the third stage, we load the model weights from the previous stage and introduce the proposed HVCA and TATI. In this stage, we optimize the parameters of LoRA, Q-Former, HVCA, TATI, and Unet.

Implementation Details. During the first stage of training, we use the AdamW optimizer [10] with a learning rate of 2e-4 and a weight decay parameter of 0. The training objectives at this stage are the combination of the mse loss between the output of VLM and CLIP text encoder, and the language model loss. The weights of both losses are 1. We train the Q-Former for 120000 steps. In the second stage, we also adopt the AdamW optimizer. The values of learning rate, weight decay, and warm-up ratio were set to 1e-5, 0, and 0.001, respectively. In this stage, the loss function is composed of the language model loss and the diffusion loss, both of the weights are set to 1. We train the model for 5000 steps in this stage. The third stage employs the same training settings as the second stage, and we optimize FireEdit for 25000 steps. We perform all experiments on 16 NVIDIA A100 GPUs.

Training Dataset. Our training is divided into three stages. In the first stage, our primary training data source is CC12M [3]. The next two stages use the same training data, which is divided into three categories: (1) segmentation dataset, which comes from COCOStuff [2], RefCOCO [17], GRefCOCO [8]; (2) image editing dataset, including InstructPix2Pix [1], MagicBrush [18], Ultraedit [19], and ReasonEdit [5]; (3) visual question answering dataset, LLAVA-instruct-150k [9].

Parameter Settings. In our FireEdit, we set the rank of LORA for fine-tuning LLM to 8, alpha to 16, and Dropout to 0.05. In the HVCA module, we set the number of layers $L = 2$, the dimension of the input visual features to 1024, the intermediate features to 1024, and 768 for the output features. We initialize 16 learnable queries. In the TATI module, we also set the number of layers $N = 2$, with the dimension of the edit features set to 768, and we initialize 77 learnable queries to keep consistent with the length of text features extracted by the CLIP text encoder. In the cross-attention layers of Unet, we set the weight factor λ as 1. We set $L_r \leq 100$ equal to the number of detected ROIs. The fused tokens are indeed represented by $e \in \mathcal{R}^{32\times4096}$, which corresponds to the hidden embeddings of 32 special tokens $\{[\text{IMG}_r]\}_{r=1}^{32}$. During the inference phase, we set the timestep to 100, image guidance scale $s_I = 1.5$, and text guidance scale $s_T = 7.5$.

2. More Visualizations

In this section, we present additional visualization results. To comprehensively demonstrate the superiority of FireEdit, we validate it from both single-turn editing and multi-turn editing. We compare our approach with other state-of-the-art instruction-based editing methods, including IP2P [1], MagicBrush [18], HQ-Edit [6], UltraEdit [19], and SmartEdit [5]. In addition, we also provide more visualization samples of text understanding.

2.1. Single-Turn Editing Examples

In Figure 1, we qualitatively compare our method with other SOTA methods in three groups of editing actions. The first three rows perform addition operations on the input image under the guidance of the instructions. The middle three rows change the content of the image, including fine-grained local editing and changing the background content. The last three rows remove specified objects in the input image. We select input images with more complex backgrounds from the Eem Edit test set [15]. As shown in Figure 1, our method can more accurately implement local ed-

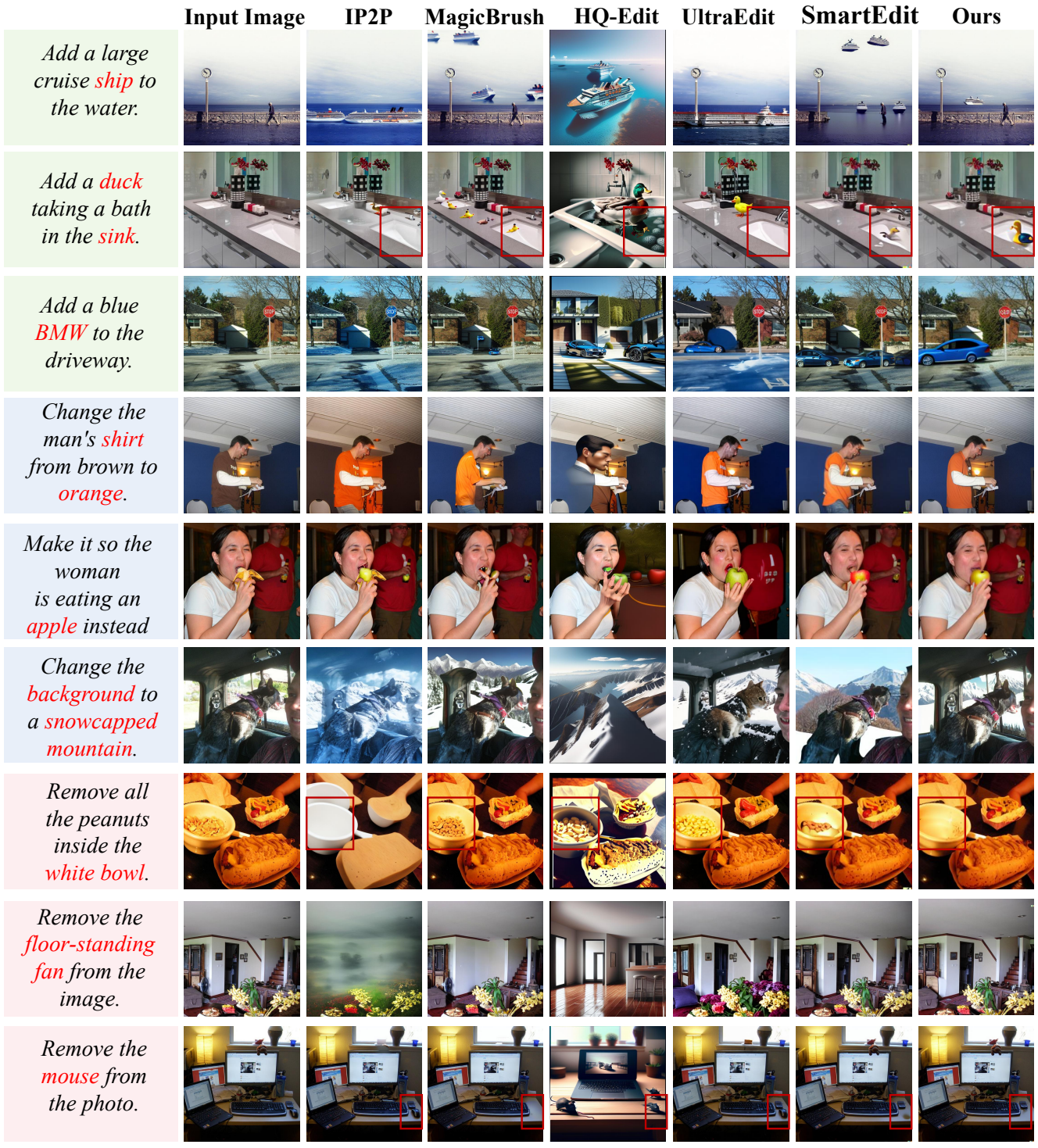

Figure 1. Qualitative comparison. We compare the editing performance of FireEdit with SOTA methods on the Emu Edit test set. The leftmost column contains the editing instructions. Compared with other SOTA methods, our approach is superior in accurately locating the edited objects or regions and preserving the detailed information of the input image.

its across the three edit types. In contrast, other baseline methods struggle to locate the edit targets or locations expressed in the instructions, and even alter the undesired areas. For example, in the first row, IP2P does not preserve

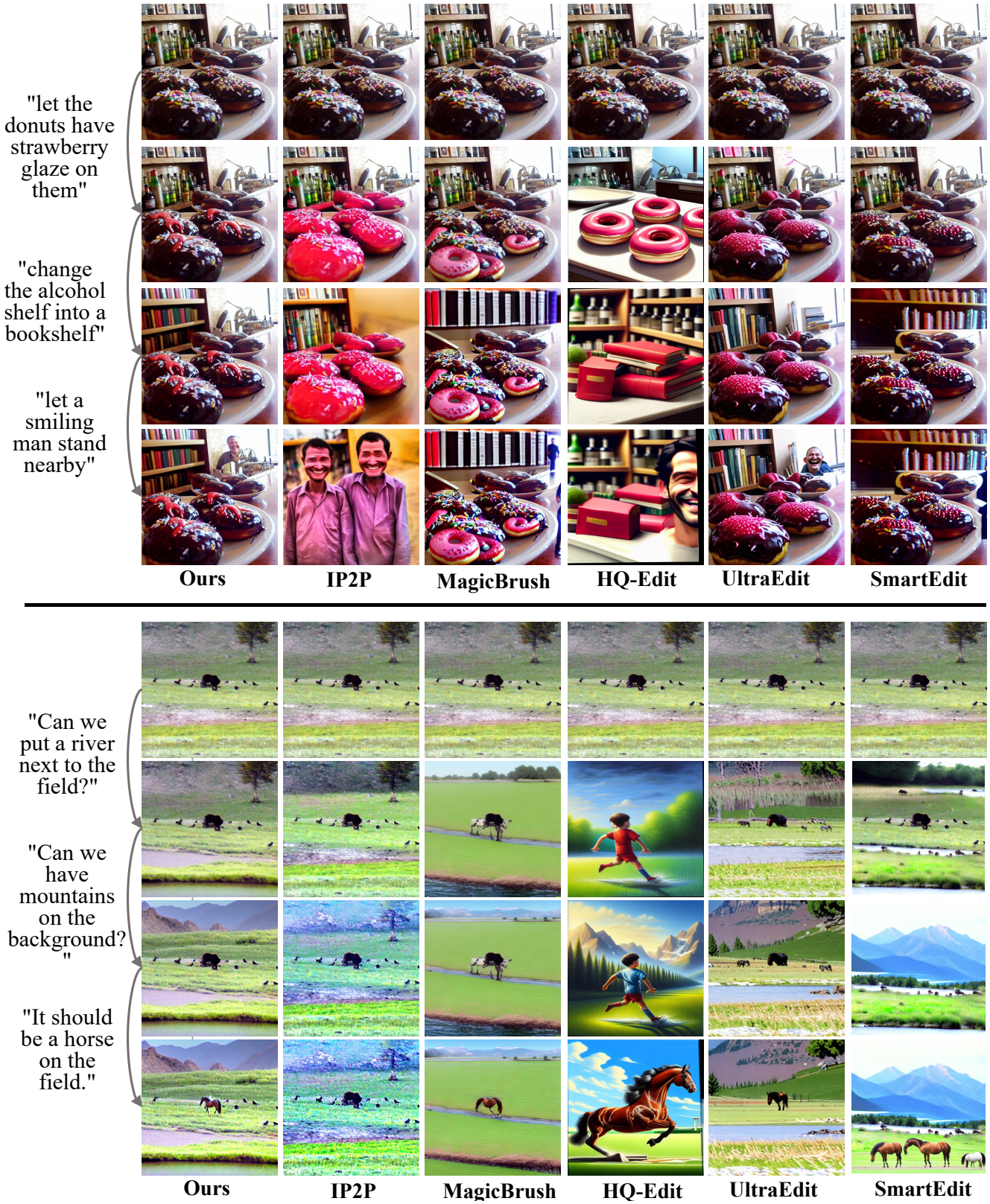

Figure 2. The qualitative comparison for our method and another state-of-the-art approach on the MagrichBrush benchmark under the multi-turn setting.

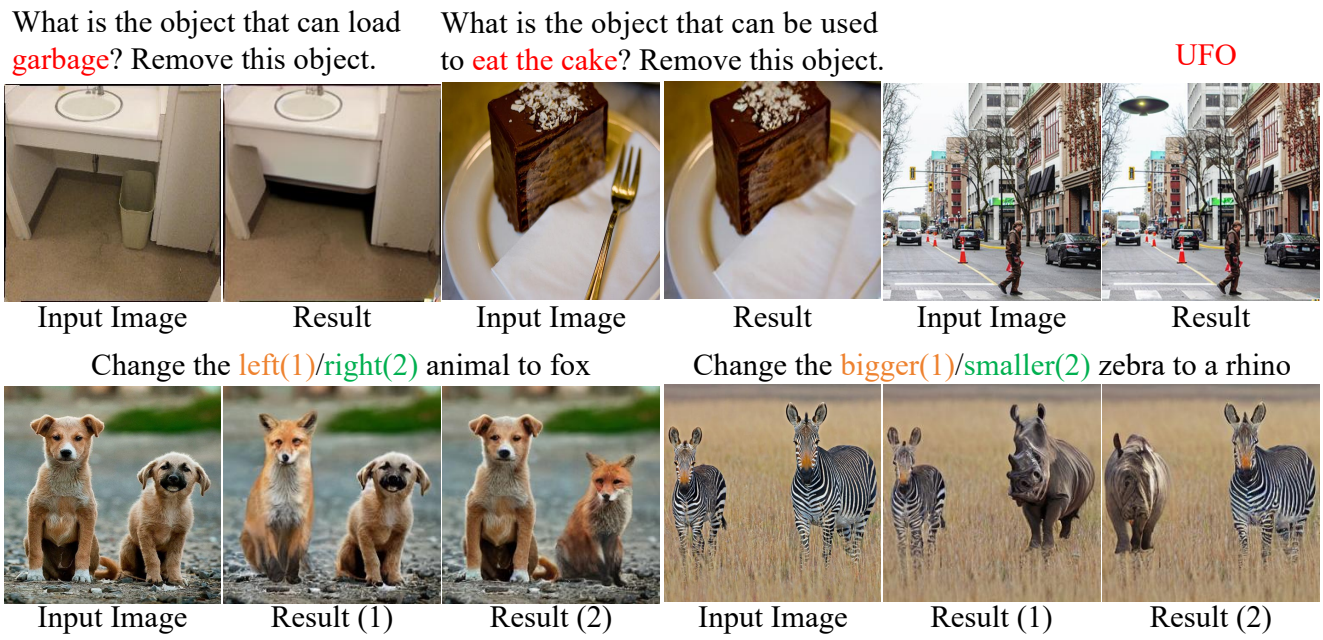

Figure 3. The visualization of text understanding.

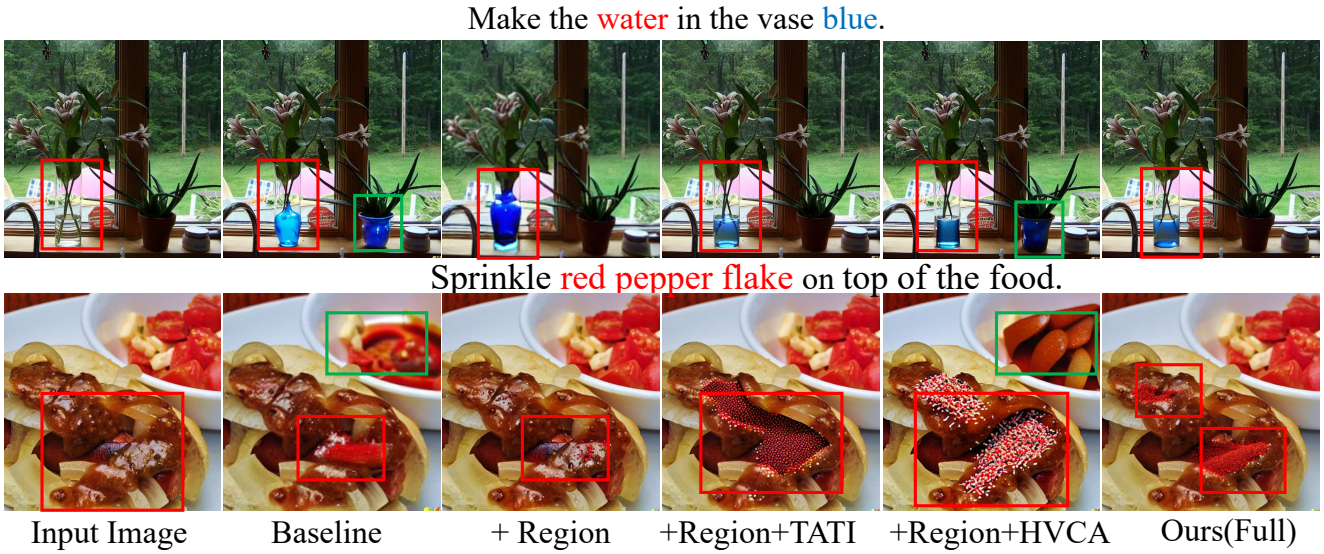

Figure 4. Visualization of ablation studies of different components.

the pedestrian and the coast, while SmartEdit adds a ship in the undesired area. In the seventh row, IP2P, MagicBrush, HQ-Edit, and UltraEdit do not accurately locate the "white bowl". Although SmartEdit locates the "white bowl", it does not completely remove the "peanuts". Our method not only accurately locates the "white bowl", but also removes the target elegantly. Therefore, our method has advantages over other methods in fine-grained local editing.

2.2. Multi-Turn Editing Examples

We perform three consecutive rounds of editing on two input images, with each round executing different editing instructions. We compare the editing results of our method with those of other methods. As shown in Figure 2, in the first example, each round of editing by our method successfully understands the instructions and accurately executes the edits without altering the semantic layout of the input image. In the second example, our method generates editing

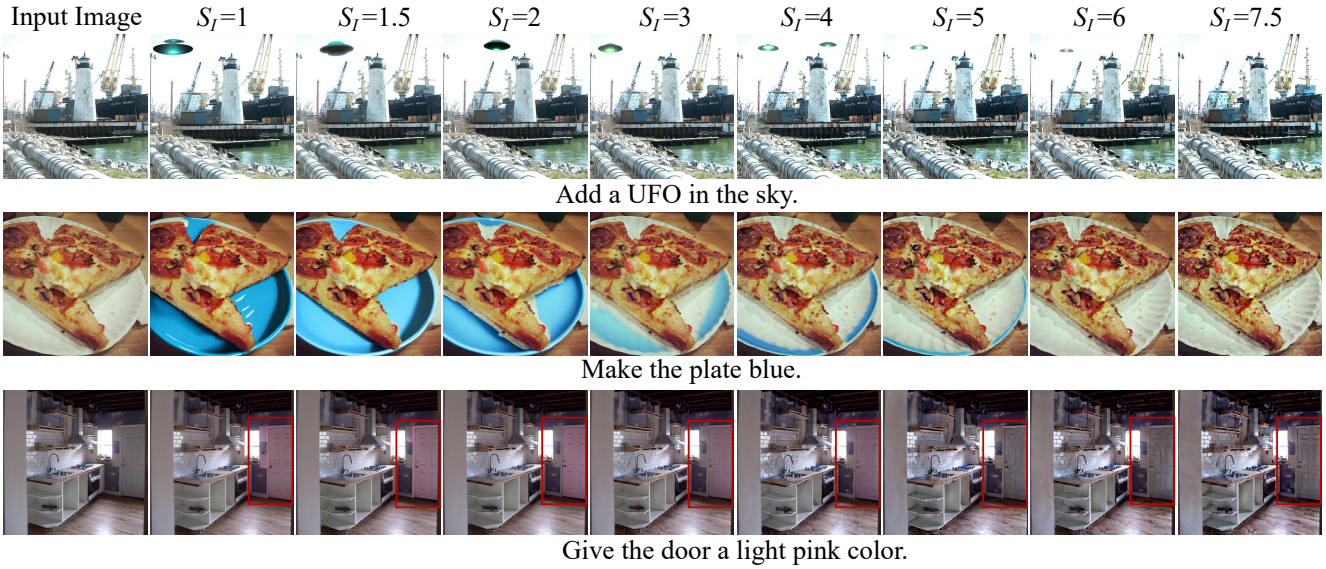

Figure 5. Effects of different image guidance scales.

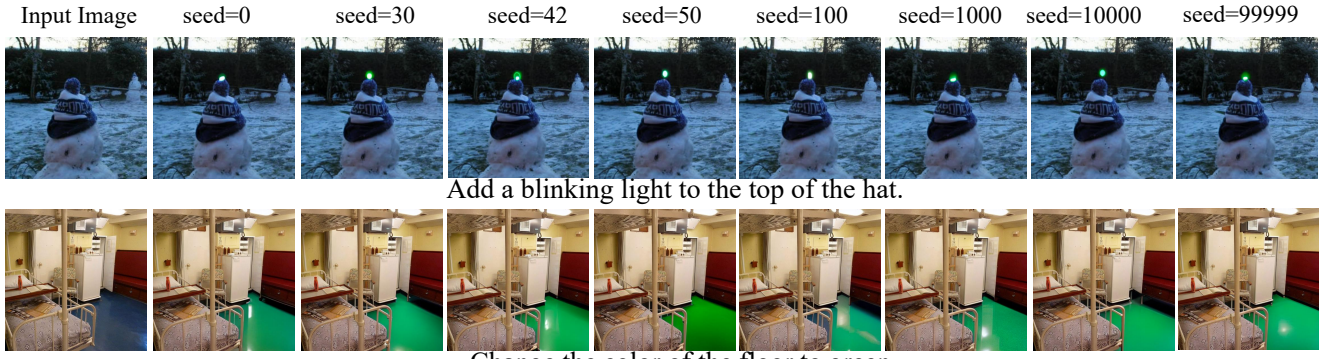

Figure 6. Effects of different random seeds.

results that are highly consistent with the instructions without causing any disharmony. However, other methods cannot generate satisfactory multi-round editing results, which are not preferred in practical applications.

2.3. Text Understanding

Our method is capable of understanding text and performing reasoning. As illustrated in Figure 3, it successfully leverages the reasoning ability of the LLM to infer and edit target objects precisely in scenarios that involve direction and relative size. Even for a simple instruction like "UFO", our method understands it well and adds the object in an appropriate place.

3. Additional Ablation Studies

In this section, we perform ablation studies on the components proposed in our method, the complexity and training data, the selection of detectors, the image guidance scale s_I, different random seeds, and sampling timesteps. We visualize the editing results under different factors.

Table 1. Comparison of complexity and training data.

Method	Data volume	Speed	Trainable Param#	Memory	CLIP-I↑	CLIP-T↑
UltraEdit	4M	3.60s	2028.4M	18.5G	0.8120	0.2773
SmartEdit	771k	7.30s	1163.8M	38.9G	0.8592	0.2740
SmartEdit	771k+4M	7.30s	1163.8M	38.9G	0.8721	0.2755
Ours	771k	7.50s	1093.8M	36G	0.8975	0.2762
Ours	771k+4M	7.50s	1093.8M	36G	**0.9140**	**0.2783**

3.1. Effect of Components

Figure 4 showcases two key module behaviors: HVCA preserves semantic details despite minor non-target effects, while TATI leverages temporal cues for superior text-guided denoising. Their synergistic integration enables precise localized editing. Specifically, TATI incorporates timestep information across the diffusion process, optimizing text representation utilization.

Figure 7. Effects of different sampling timesteps.

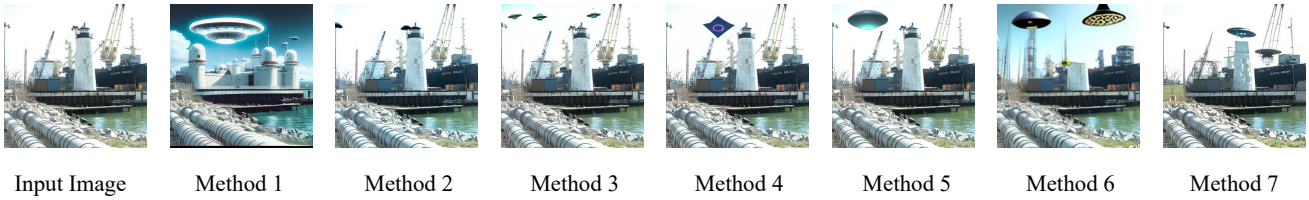

Figure 8. The interface of user preference study questionnaire.

3.2. Comparison of Complexity and Training Data

We estimate model efficiencies on a single A100 GPU with 50 denoising steps for all methods on the Emu Edit Test, as detailed in Table 1. The inference speed of our method (7.5s) is similar to those of cutting-edge methods (e.g., SmartEdit, 7.3s), but our method achieves significant performance improvements in fine-grained editing tasks. Moreover, with fewer trainable parameters, our approach

Table 2. Ablation studies of regional encoders.

Method	L1↓	CLIP-I↑	DINO↑	LPIPS↓	CLIP-T↑
Ours (w/224)	0.0723	0.9093	0.8762	0.1841	0.2798
Ours (w/448)	0.0772	0.9046	0.8626	0.1926	0.2786
Ours (+YOLOv10)	0.0586	0.9074	0.8775	0.1705	0.2785
Ours (+SAM)	0.0561	0.9159	0.8885	0.1632	0.2769
Ours	0.0574	0.9140	0.8829	0.1373	0.2783

fine-tunes the model in just 64 hours on 16 A100 GPUs, achieving a better balance between efficiency and performance. Table 1 summarizes the training data used in previous methods. For a fair comparison, we adopt SmartEdit as the baseline and utilize the same training data. When trained on the 771k dataset, our method consistently surpasses SmartEdit across all metrics.

3.3. Effects of Different Detectors

We aim at image editing for open-world scenarios, hence we adopt a modified DDETR which has a class-agnostic detection head and can detect more high-quality potential ROIs in open scenes. It can be replaced by other off-the-shelf detectors/segmentation models like YOLOv10 [16] and SAM [7]. As shown in Table 2, our method performs stably with these detectors. The latency of extracting region tokens is about 0.2s, which is ignorable. In the HVCA module, we employ two resolutions (224 and 448) to enhance visual details. By integrating features from the CLIP image encoder and DINOV2, HVCA effectively captures both global and local visual information.

3.4. Image Guidance Scale s_I

To effectively observe the impact of the image guidance scale on the editing results, we set the random seed to 42, the sampling timestep to 100, and the text guidance strength to 7.5. We change the value of s_I from 1 to 7.5. The visualization results are shown in Figure 5. As s_I increases, the intensity of image guidance gradually increases, while the influence of text guidance gradually decreases. When s_I increases to 6, the effect of text instructions disappears. We found that the editing result is best when s_I is 1.5, which is consistent with [1, 5].

3.5. Random Seed

We fixed the text guidance factor at 7.5, the image guidance factor at 1.5, and the default sampling timestep at 100. We selected a set of random seeds from a range of 0 to 100000. As shown in Figure 6, the value of the random seed has a minimal impact on the editing results, indicating that our method exhibits strong robustness.

3.6. Sampling Timestep

We keep other parameters unchanged and visualize the results when the sampling timestep is 20, 30, 50, 60, 80, and 100. In Figure 7, we observe that as the sampling timestep becomes larger, the editing quality first improves and then decreases. This indicates that with an increase in sampling timestep, the risk of over-editing becomes higher.

4. Human Evaluation

We sampled 30 real images from the test set and generated edited results on the 7 methods, resulting in 30 questions. We invited 40 participants to answer these 30 questions. To ensure that participants could choose the best edited result by asking them to give a comprehensive rating based on the following three aspects: 1) semantic consistency, i.e., the preservation of undesired areas in the edited image; 2) text-image alignment between the edited image and the output caption; and 3) the quality and fidelity of the edited image. Figure 8 shows the questionnaire format we used in the human preference study, where the order of displaying the 7 methods for each question was randomized.

5. Societal Impact

We propose a novel image editing method that leverages the powerful multimodal perception capabilities of visual language models (VLMs) to facilitate the understanding of editing instructions in complex scenarios. By incorporating regional details, we further enhance the fine-grained perception ability of the VLM, thereby implicitly locating the desired editing regions or targets. By associating time step information with the output of the VLM, the generated images can preserve high-frequency detail information. Consequently, our editing technique can be applied to social media and assistive art creation. Due to the fine-grained local control achieved by our method, it could potentially be misused by malicious groups to manipulate image content and mislead the public. However, we believe that these potential threats will be mitigated with the improvement of regulations and the maturity of intelligent detection technologies.

References

[1] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023. 1, 7

[2] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1209–1218, 2018. 1

[3] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3558–3568, 2021. 1

[4] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen.

Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 1

[5] Yuzhou Huang, Liangbin Xie, Xintao Wang, Ziyang Yuan, Xiaodong Cun, Yixiao Ge, Jiantao Zhou, Chao Dong, Rui Huang, Ruimao Zhang, et al. Smartedit: Exploring complex instruction-based image editing with multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8362–8371, 2024. 1, 7

[6] Mude Hui, Siwei Yang, Bingchen Zhao, Yichun Shi, Heng Wang, Peng Wang, Yuyin Zhou, and Cihang Xie. Hq-edit: A high-quality dataset for instruction-based image editing. *arXiv preprint arXiv:2404.09990*, 2024. 1

[7] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 7

[8] Chang Liu, Henghui Ding, and Xudong Jiang. Gres: Generalized referring expression segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 23592–23601, 2023. 1

[9] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 1

[10] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[11] Chuofan Ma, Yi Jiang, Jiannan Wu, Zehuan Yuan, and Xiaojuan Qi. Groma: Localized visual tokenization for grounding multimodal large language models. In *European Conference on Computer Vision*, pages 417–435. Springer, 2025. 1

[12] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 1

[13] Hanoona Rasheed, Muhammad Maaz, Sahal Shaji, Abdelrahman Shaker, Salman Khan, Hisham Cholakkal, Rao M Anwer, Eric Xing, Ming-Hsuan Yang, and Fahad S Khan. Glamm: Pixel grounding large multimodal model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13009–13018, 2024. 1

[14] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 1

[15] Shelly Sheynin, Adam Polyak, Uriel Singer, Yuval Kirstain, Amit Zohar, Oron Ashual, Devi Parikh, and Yaniv Taigman. Emu edit: Precise image editing via recognition and generation tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8871–8879, 2024. 1

[16] Ao Wang, Hui Chen, Lihao Liu, Kai Chen, Zijia Lin, Jungong Han, et al. Yolov10: Real-time end-to-end object detection. *Advances in Neural Information Processing Systems*, 37:107984–108011, 2024. 7

[17] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 69–85. Springer, 2016. 1

[18] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. *Advances in Neural Information Processing Systems*, 36, 2024. 1

[19] Haozhe Zhao, Xiaojian Ma, Liang Chen, Shuzheng Si, Rujie Wu, Kaikai An, Peiyu Yu, Minjia Zhang, Qing Li, and Baobao Chang. Ultraedit: Instruction-based fine-grained image editing at scale. *arXiv preprint arXiv:2407.05282*, 2024. 1

[20] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 1