Supplementary Material
ZONE: Zero-Shot Instruction-Guided Local Editing

Shanglin Li[1]*, Bohan Zeng[1]*, Yutang Feng[1]*, Sicheng Gao[1] Xiuhui Liu[1], Jiaming Liu[2],
Lin Li[2], Xu Tang[2], Yao Hu[2], Jianzhuang Liu[4], Baochang Zhang[1,3,5]†
[1]Beihang University [2]Xiaohongshu Inc [3]Nanchang Institute of Technology, China
[4]Shenzhen Institute of Advanced Technology, China [5]Zhongguancun Laboratory, China

In this supplementary, we first give more visualization results, then detail the datasets and the implementation, and finally discuss the limitations and state the social impact.

A. More Visualizations

In this section, we first present more visualizations of the samples from the test set under two comparison settings: (i) single-turn editing, and (ii) multi-turn editing. To make the comparison more representative, we compare our ZONE with three state-of-the-art (SOTA) text-to-image approaches, Text2LIVE (T2L) [1], InstructPix2Pix (IP2P) [2], and MagicBrush (MB) [9]. Then we conduct an ablation study to show the efficacy of our fused IP2P module, through cross-attention map visualization.

A.1. Single-Turn Editing Examples

We show more single-turn editing examples to further validate ZONE's remarkable ability of local image editing. In particular, we compare it with the other methods for local editing using 9 images and their corresponding instructions (or prompts equivalent to the instructions). As evident in Fig. A, the results generated by ZONE surpass those of the other methods, demonstrating its impressive prowess in local editing.

A.2. Multi-Turn Editing Examples

We use our ZONE to edit 2 images in a multi-turn style and compare the editing results with those obtained from the other methods. Specifically, each method is employed to edit each image three times, with different instructions. As illustrated in Fig. B, our ZONE can achieve high-quality local edits under multiple instructions and preserve the original image's non-edited regions. In contrast, the results generated by the other methods exhibit noticeable distortions from the original images after multiple rounds of editing, which is not preferred in practical applications.

A.3. Cross-Attention Map Visualization

As shown in Fig. C, the first row demonstrates the editing results, and the second row illustrates the averaged cross-attention maps. From the cross-attention maps, we can see that by fusing the denoised latents of the two methods as described in Equation (5) of the main paper, our approach achieves a better localization capability under the "Remove" editing intent compared to the two methods.

B. Implementation Details

We conduct all our experiments based on open-source projects and models. We adopt an NVIDIA V100-SXM2-32GB GPU for the action classifier training and for ZONE testing. The action classifier $\mathcal{A}_I$ leverages the instruction embeddings extracted by the text encoder of InstructPix2Pix (IP2P) [2] as its input, and outputs the probability logits for each action. To train the action classifier, we first use GPT-3.5 to generate samples for training and testing, and then we lock the weights of the text

*These authors contributed equally.
†Corresponding author: bczhang@buaa.edu.cn

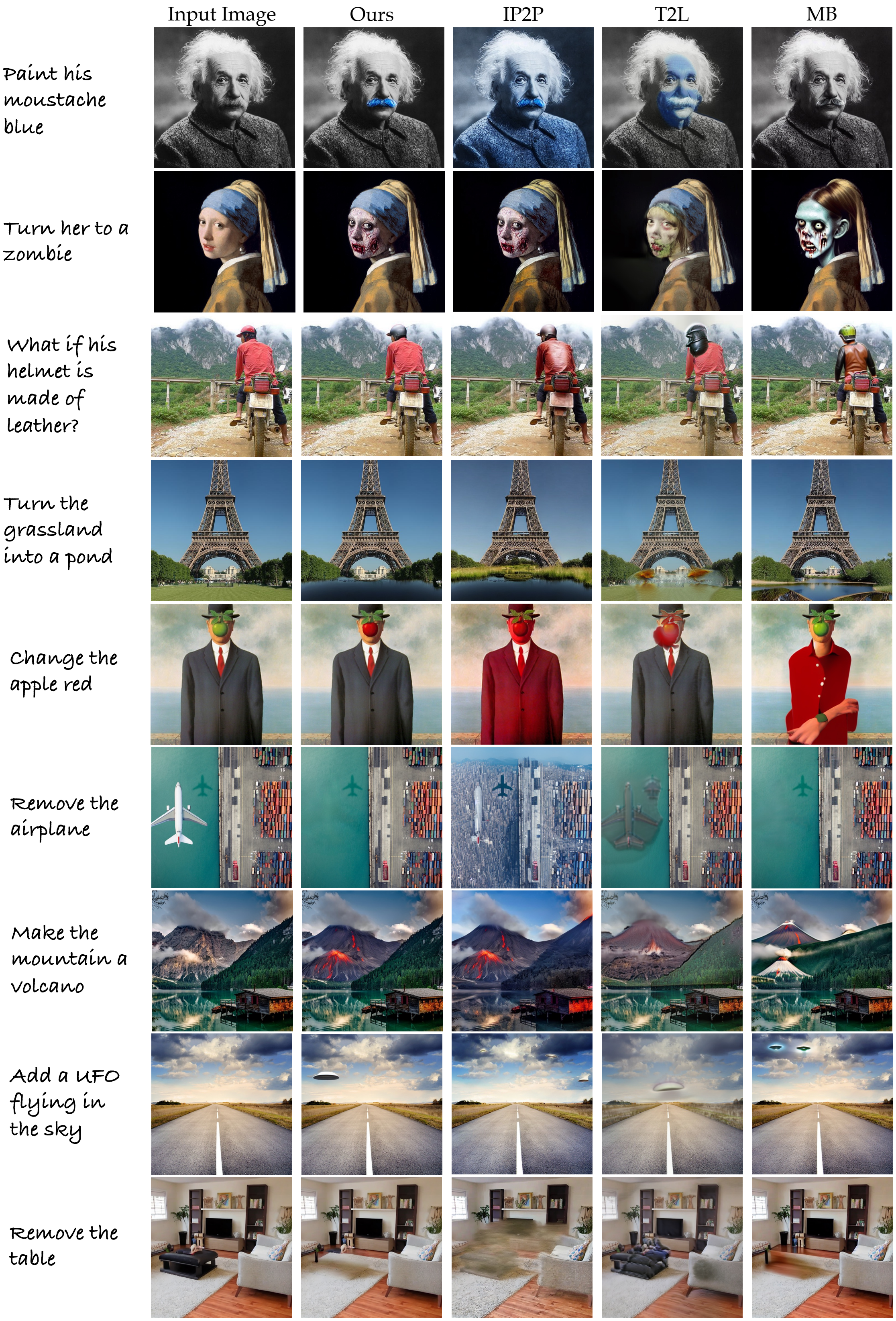

Figure A. **Single-turn editing examples.** IP2P: InstructPix2Pix [2]; T2L: Text2LIVE [1]; MB: MagicBrush [9].

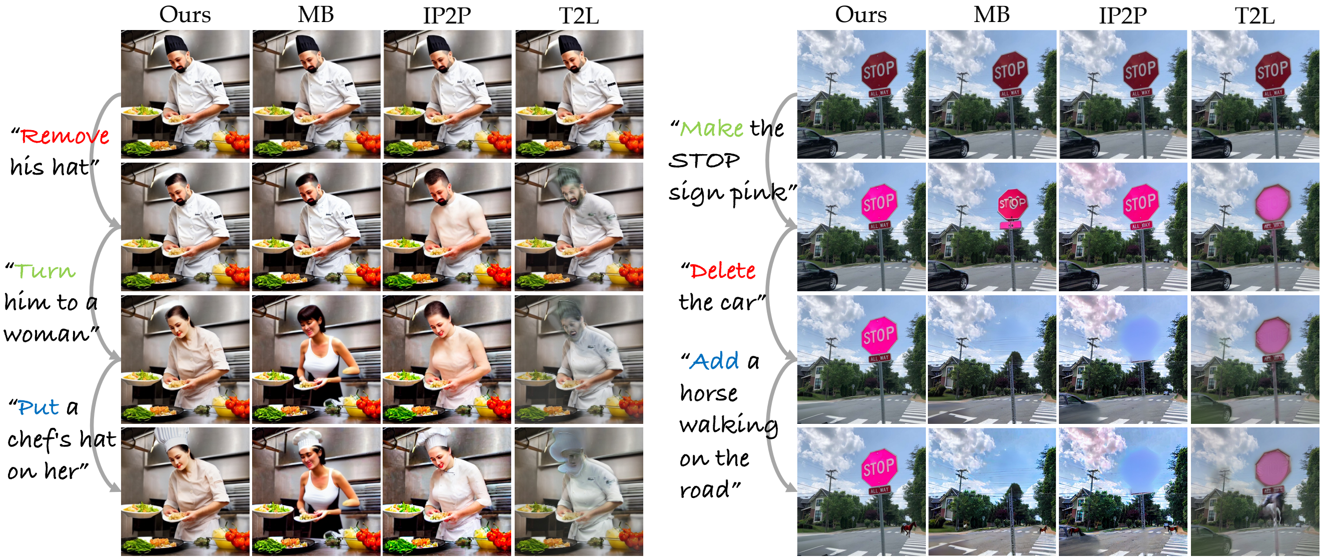
Ours
MB
IP2P
T2L
"Remove his hat"
"Turn him to a woman"
"Put a chef's hat on her"
Ours
MB
IP2P
T2L
"Make the STOP sign pink"
"Delete the car"
"Add a horse walking on the road"

Figure B. **Multi-turn editing examples.** IP2P: InstructPix2Pix [2]; T2L: Text2LIVE [1]; MB: MagicBrush [9]. Best viewed zoomed in.

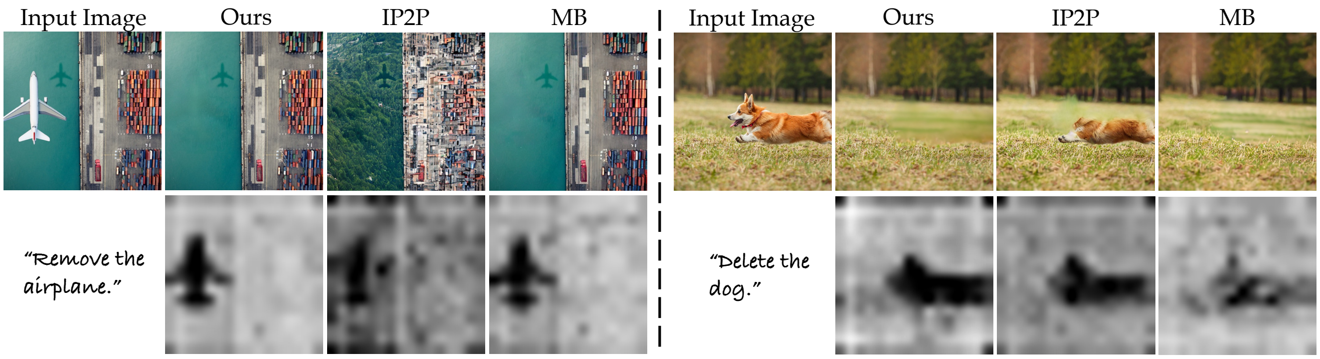

Figure C. **Cross-attention map comparisons.** The **darker parts** in each cross-attention map (the second row) denote the edit regions.

encoder of IP2P and optimize $\mathcal{A}_I$ using Adam [4] with a learning rate of 0.1 for 30 epochs. The action classifier achieves 100% top-1 classification accuracy on the test set.

We set 20 sampling steps for the fused IP2P and average the cross-attention layers of the first three UNet upsampling blocks and the second to the fourth downsampling blocks to get the fused cross-attention maps among all the denoising steps.

The action classifier $\mathcal{A}_I$ is a simple Multi-Layer Perceptron (MLP), comprising two linear layers with an intermediate ReLU activation function. The input dimension of the first linear layer and the length of the embedding outputted from the CLIP text encoder [7] are the same (equal to 768), and the output dimension at this layer is 128. The intermediate ReLU function introduces non-linearity to the output, and the second linear layer takes the 128-dimensional output from the ReLU function and produces a 3-dimensional output to classify the given instruction.

C. Experimental Details

C.1. Baselines

To ensure consistency and convenience in method comparison, we uniformly adopt the implementation from the diffusers project [1] for IP2P [2], MagicBrush [9], DiffEdit [3], and Pix2Pix-Zero [6], and use their default parameters to generate results and calculate the metrics. For Text2LIVE [1], we conduct experiments using its official code repository. To eliminate the potential discrepancies in generative capabilities arising from different versions of Stable Diffusion used across these meth-

[1] https://github.com/huggingface/diffusers

ods, we employ Stable Diffusion 1.5 [2] as the base model. Notably, since half of these methods do not support instructions as textual inputs, we design text prompts or additional assistance equivalent to instructions during our comparative experiments to achieve a relatively fair comparison.

C.2. Datasets

In this section, we provide the generation details of the dataset for action classification and the test set that we collect to evaluate the metrics for our ZONE and other editing methods.

Dataset for action classification. We employ GPT-3.5 [3] to generate the dataset used for training the action classifier. Our primary objective is to generate sentences that closely resemble user instructions, with the editing focus on common items found in real images. To achieve this, we choose categories from the COCO dataset to serve as the vocabulary for sentence generation. The following prompt is designed for generating training and testing data:

"Now you are a dataset bot, who will generate a training dataset for a three-fold (change, add, and remove) sentence classification task. Specifically, you should generate a sentence along with its label. In this task, we aim to generate a dataset for "change", "add", and "remove" (labeled 0, 1, 2): For example: "turn the cat into a dog, 0", "give the dog a hat, 1", "get rid of the person on the left, 2". You should generate 450 sentence-label pairs if I give the instruction "train", and 150 pairs when I give the instruction "test". I expect your response to be straight-forward, each sentence should be within 30 words, and you should freely select the words in the following list: ['person', 'bicycle', 'car', 'motorcycle', 'airplane', 'bus', 'train', 'truck', 'boat', 'traffic light', 'fire hydrant', 'stop sign', 'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'cow', 'elephant', 'bear', 'zebra', 'giraffe', 'backpack', 'umbrella', 'handbag', 'tie', 'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball', 'kite', 'baseball bat', 'baseball glove', 'skateboard', 'surfboard', 'tennis racket', 'bottle', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl', 'banana', 'apple', 'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza', 'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed', 'dining table', 'toilet', 'tv', 'laptop', 'mouse', 'remote', 'keyboard', 'cell phone', 'microwave', 'oven', 'toaster', 'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors', 'teddy bear', 'hair drier', 'toothbrush'] and make sure the sentence is short and clear, and the label is correct, and the dataset is balanced. Please just reply with the sentence-label pairs, and wait for my instructions. Note that the generated sentence-label pairs should not repeat."

The data generated by GPT-3.5 undergoes manual verification. The final training dataset includes 150 samples each for the "add", "remove", and "change" actions, while the test dataset comprises 50 samples for each action. We show some samples from the training dataset in Table A.

Turn the bicycle into a motorcycle, 0
Make the apple a banana, 0
Swap the baseball glove for a tennis racket, 0
Replace the chair with a couch, 0
Put a frisbee next to the cat, 1
Attach a remote to the TV, 1
Include a toothbrush on the dining table, 1
Give the horse a suitcase, 1
Remove the horse, 2
Take away the umbrella, 2
Delete the traffic light, 2
Erase the microwave, 2

Table A. Examples of the training dataset of the action classifier.

[2] https://huggingface.co/runwayml/stable-diffusion-v1-5

[3] https://chat.openai.com

Test set for evaluation. We present the test set utilized in our evaluation in Fig. D. Initially, we gather 60 images from the Internet and create 40 synthetic images using Stable Diffusion 1.5 [8]. Subsequently, each image is cropped to a resolution of 512 × 512. Then we use BLIP [5] to caption each image and manually annotate the instructions, output captions, source objects, and target objects. Three annotation examples are shown in Table B.

Keys	**Example 1**	**Example 2**	**Example 3**
action	Change	Remove	Add
input caption	A blue car in front of a forest	A man in black with a tie	A photo of Elon Musk
output caption	A red car in front of a forest	A man in black	A photo of Elon Musk with glasses
instruction	paint the car red	get off his tie	give him glasses
source object	blue car	a tie	N/A
target object	red car	N/A	glasses

Table B. **Three annotation examples.** "N/A" indicates the absence of words.

C.3. Evaluation Metrics

L1/L2 distance. The L1 and L2 distances serve as the metrics for evaluating structural and pixel-wise similarities between two images. The L1 distance measures absolute differences in pixel values, while the L2 distance calculates squared differences. Both metrics play a critical role in assessing dissimilarity, with smaller distances indicating greater image similarity in both pixel intensity and spatial structure.

LPIPS score. LPIPS (Learned Perceptual Image Patch Similarity) [10] is a metric designed for evaluating the perceptual similarity between two images. It takes into account both pixel-level differences and high-level visual features, providing a comprehensive measure of how similar images appear to humans.

CLIP-based metrics. CLIP, or Contrastive Language-Image Pre-training, is a transformative model that excels in understanding the intricate relationships between text descriptions and images [7]. Through a pre-training process that employs contrastive learning, CLIP learns a shared embedding space where images and text descriptions are represented as vectors. This shared space is designed to bring semantically related content in close proximity. The model tokenizes images into regions and text into tokens, leveraging a transformer architecture with cross-modal attention to establish connections between corresponding regions and tokens. Both the CLIP-I and CLIP-T metrics evaluate the input image/text in the shared embedding space:

- CLIP image similarity (CLIP-I) is designed to evaluate the image quality in both semantics and structure. This metric is computed by calculating the cosine similarity of the embedding vectors of the source image and the target image.
- CLIP text-image similarity (CLIP-T) is used to evaluate the alignment between the edited image and its corresponding caption. More specifically, CLIP-T calculates the cosine similarity between the embedding vectors of the edited image and its corresponding caption.

More evaluation details. We employ 512 × 512 images as inputs for each method during evaluations. However, DiffEdit [3] requires image inputs with a resolution of 768 × 768 to function properly. So we first resize the test images to 768 × 768 for DiffEdit to ensure its proper performance and resize the outputs back to 512 × 512 to calculate the metrics.

C.4. Human Evaluation

Success rate. We invite five volunteers to annotate the success rates of the six methods on the test set. To simplify the annotation process and avoid bias, we design a tool that can display the editing results of each method in a randomly shuffled order and anonymous style (see Fig. E). The volunteers are then asked to decide whether to accept or reject the edited image based on the editing quality (*i.e.,* preservation of the non-edited regions and the realism of the edited image) and text-image alignment between the output caption and the edited image. Ultimately, the success rate of each method is obtained by dividing the number of accepted results by the total number. To minimize annotation bias, we calculate the mean and standard deviation of the success rates from the five volunteers and demonstrate the results in Table 2 of the main paper.

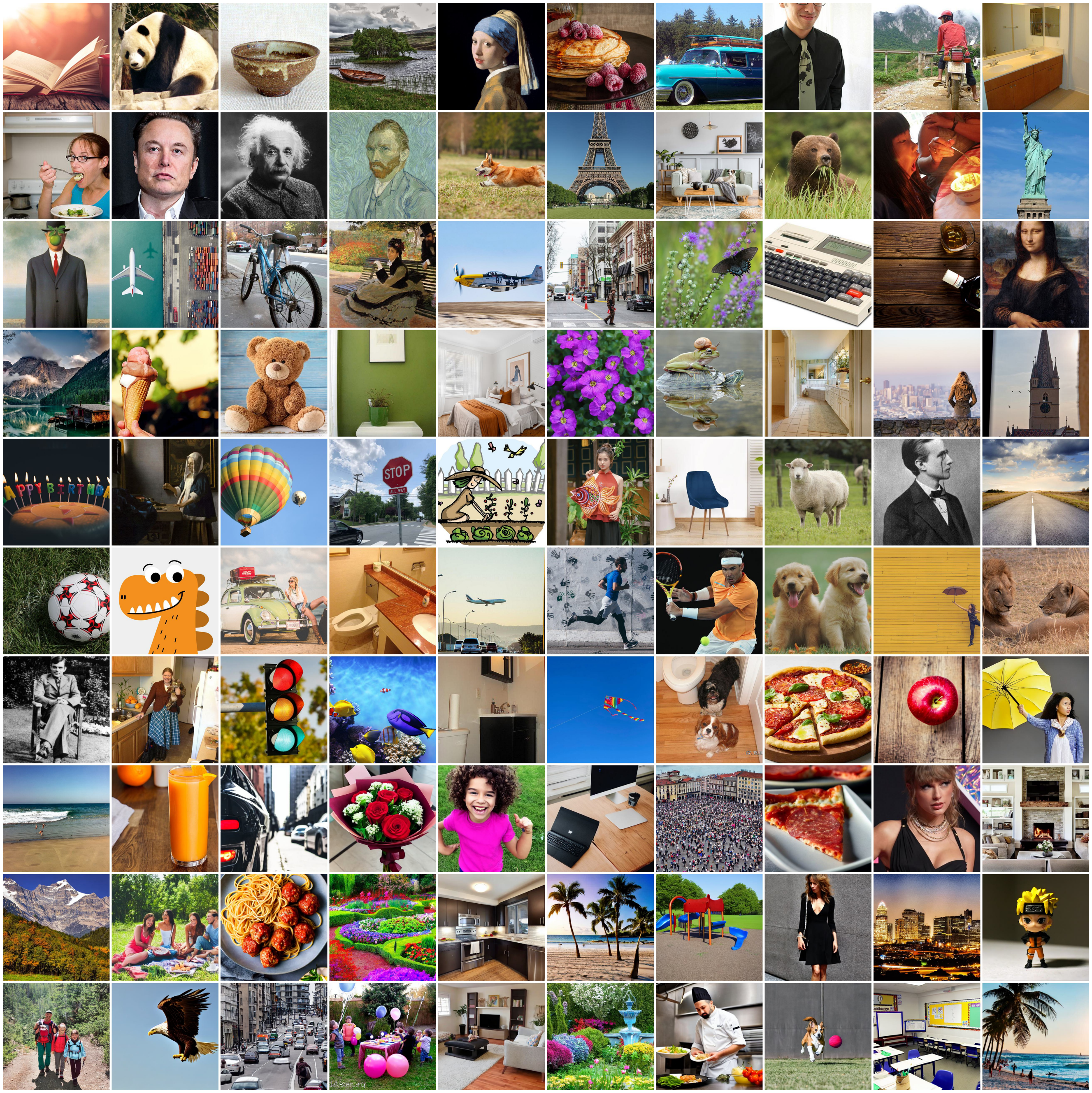

Figure D. **Images in the test set**. We calculate the evaluation metrics and provide visualizations in the main paper with images in this set.

User preference rate. We conduct a user study, which includes 16 sets of randomly selected editing results. Each set contains six results obtained by the six methods that we compare in the experiment, presented in a randomly shuffled order. The users are asked to give a preference score according to the degree of agreement between each editing result and the corresponding instruction, as well as the similarity to the original image, with the score from 1 to 10 and a higher score indicating a higher preference. A total of 30 users participate in this test. The final results are calculated by dividing the total

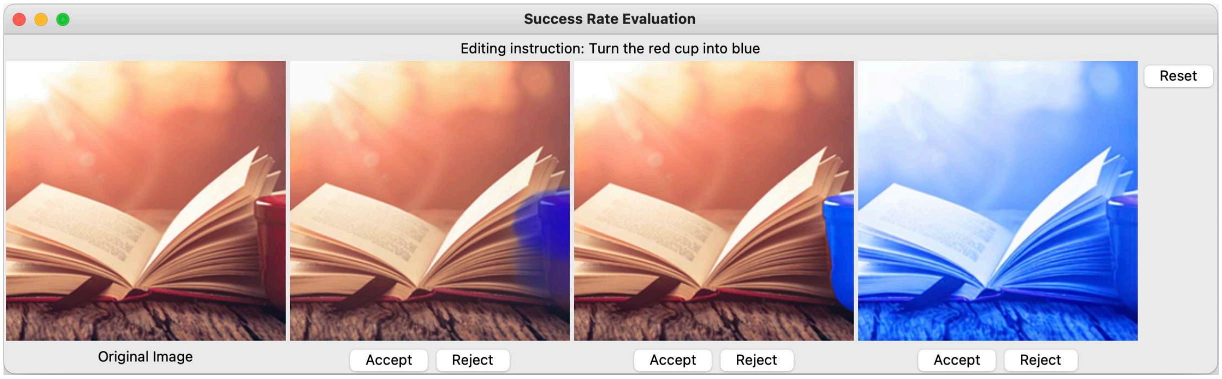

Figure E. **A screenshot of the annotation tool.** The original image, the instruction, and three results randomly selected from the six methods are displayed each time.

score obtained for each method S_i by the total score obtained for all methods:

$$UPR(i) = 100 \times S_i / \sum_{i=1}^{6} S_i. \tag{A.1}$$

C.5. Additional Ablations

We perform further ablations to demonstrate the effectiveness of each component within our proposed ZONE. First, we ablate on the β parameter of the fused IP2P module. β is a hyperparameter that controls the guidance strength of MagicBrush (MB) on the fused IP2P module. A higher β emphasizes MB's effects on the editing results (please check L302-314 in our paper for more details). We illustrate the results generated by the models under various β in Fig. F:

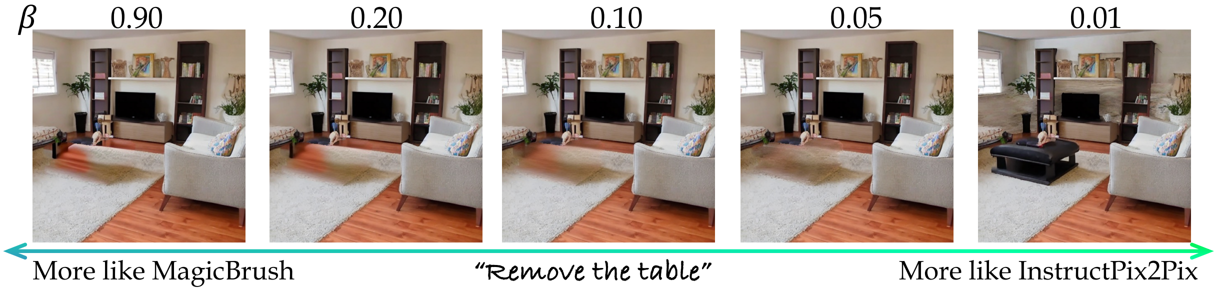

Figure F. **Ablation on β.** As β decreases from left to right, the generated result increasingly resembles that produced by InstructPix2Pix and less resembles that generated by MagicBrush. Given the limits of current instruction-guided diffusion models, we employ this character to handle different editing actions.

Then we conduct an ablation study on the components of ZONE, including SAM and the edge smoother, with human evaluation using the user preference rate (UPR) and the CLIP image similarity (CLIP-I) for metrics evaluation. As demonstrated in Tab. C, the implementation of SAM notably boosts the user preference rate, signifying a visually discernible enhancement compared to its absence, a fact also mirrored in the CLIP-I score. Concurrently, the user preference rate exhibits a significant improvement with the integration of the Edge Smoother, underscoring its efficacy.

D. Limitations.

While our method can produce impressive local manipulations of images and address the over-edit issue of InstructPix2Pix, it still has limitations. First, its editing capabilities are constrained by the instruction-guided diffusion models we employ, which may lead to occasional ineffectiveness in editing. This issue can be addressed in the future with more powerful instruction-guided diffusion models. Secondly, our method falls short of localization in complex scenes (*e.g.*, multiple similar objects or tiny objects), which is a challenging task that still needs to be explored. Lastly, the current set of editing actions is relatively limited, more actions like "move", "resize", or "copy" will be considered in future work.

Component	SAM		Edge Smoother	
	w/o	w/	w/o	w/
CLIP-I ↑	0.95	**0.97**	**0.97**	**0.97**
UPR (%) ↑	5.4	**94.6**	22.7	**77.3**

Table C. **Ablation on ZONE's components.** We evaluate both components with human evaluation and CLIP metrics.

E. Social Impact

Our work introduces a novel method for image local editing, which edits a specific region in the original image with an intuitive instruction. This method allows for precise local editing without affecting other areas of the image, resulting in a realistic final composite image. Malicious groups may exploit this advantage to spread false information or cause misunderstanding. However, we believe that the harm caused by such improper usage can be mitigated with AI-generated content watermarking algorithms or supervising regulations.

References

[1] Omer Bar-Tal, Dolev Ofri-Amar, Rafail Fridman, Yoni Kasten, and Tali Dekel. Text2live: Text-driven layered image and video editing. In *ECCV*, 2022. 1, 2, 3

[2] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023. 1, 2, 3

[3] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. *arXiv preprint arXiv:2210.11427*, 2022. 3, 5

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[5] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022. 5

[6] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. In *SIGGRAPH*, 2023. 3

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 3, 5

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 5

[9] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. *arXiv preprint arXiv:2306.10012*, 2023. 1, 2, 3

[10] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5