Editable Image Elements for Controllable Synthesis

Jiteng Mu[1], Michaël Gharbi[2], Richard Zhang[2], Eli Shechtman[2], Nuno Vasconcelos[1], Xiaolong Wang[1], and Taesung Park[2]

[1] University of California, San Diego, USA {jmu,nuno,xiw012}@ucsd.edu
[2] Adobe Research, USA {mgharbi,rizhang,elishe,tapark}@adobe.com
https://jitengmu.github.io/Editable_Image_Elements/

Abstract. Diffusion models have made significant advances in text-guided synthesis tasks. However, editing user-provided images remains challenging, as the high dimensional noise input space of diffusion models is not naturally suited for image inversion or spatial editing. In this work, we propose an image representation that promotes spatial editing of input images using a diffusion model. Concretely, we learn to encode an input into "image elements" that can faithfully reconstruct an input image. These elements can be intuitively edited by a user, and are decoded by a diffusion model into realistic images. We show the effectiveness of our representation on various image editing tasks, such as object resizing, rearrangement, dragging, de-occlusion, removal, variation, and image composition.

Keywords: Image Editing · Disentangled Representation · Diffusion Models

1 Introduction

High capacity diffusion models [36,37,39] trained for the text-conditioned image synthesis task are reaching photorealism. The strong image prior learned by these models is also effective for downstream image synthesis tasks, such as generating new scenes from spatial conditioning [29,52], or from a few example photos of a custom object [13,22,38].

However, while diffusion models are trained to generate images "from scratch", retrofitting them for image *editing* remains surprisingly challenging. One paradigm is to invert from image space into the noise space [29,34,42]. However, there is a natural tension between faithfully reconstructing the image and having an editable representation that follows the training distribution, leading to challenges in what types and how much noise and regularization to add. An alternative is to tune the diffusion model to condition on a representation of the image [52], such as a ControlNet conditioned on edgemaps. However, while the diffusion model will adhere to the guidance, it may not capture properties of the input image that are absent in the guidance signal. Finally, one option is to tune the network on a set of images of a concept [13, 22, 38]. Although such methods

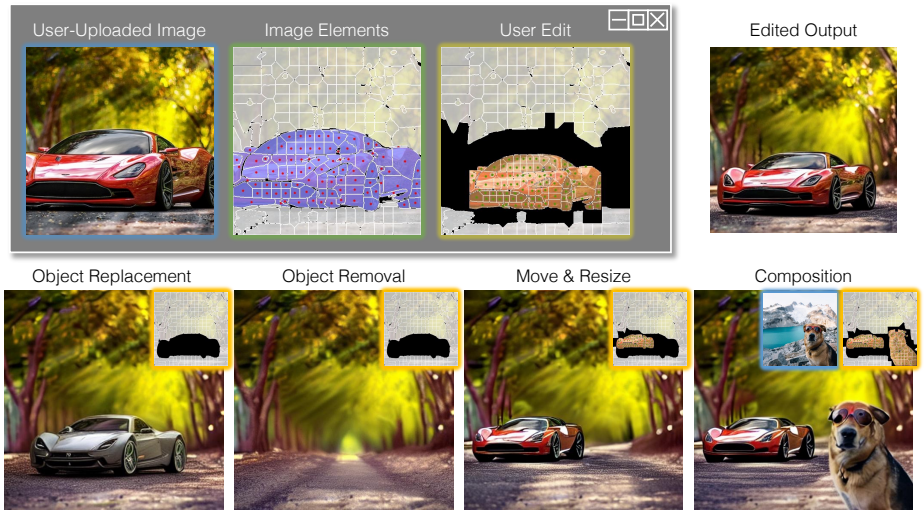

Fig. 1: We propose *editable image elements*, a flexible representation that faithfully reconstructs an input image, while enabling various spatial editing operations. *(top)* The user simply identifies interesting image elements (red dots) and edits their locations and sizes (green dots). Our model automatically performs object-shrinking and de-occlusion in the output image to respect the edited elements. For example, the missing corners of the car are inpainted. *(bottom)* More editing outputs are shown: object replacement, object removal, re-arrangement, and image composition.

generate new instances of the concept in novel situations, these are completely new images and not modifications of the original image. Furthermore, in these existing workflows, the representations (the input noise map, or edge maps) are not amenable to precise spatial controls. Our goal is to explore a complementary representation to enable spatial editing of an input image.

To this end, we present an image editing framework that not only achieves faithful reconstruction of the input image at fast speed without optimization loops, but also allows for spatial editing of the input image. Our editing process begins by dividing each content of the input image into patch regions (Figure 1) and encoding each patch separately. We represent the image as the collection of patch embeddings, sizes, and centroid locations, which are directly exposed to the user as controls for editing. The patch visualization provides intuitive control points for editing, as patches are movable, resizable, deletable, and most importantly delineated with semantically meaningful boundaries. The edited patch attributes are decoded into realistic images by a strong diffusion-based decoder. In particular, while our decoder is trained to preserve the input content as much as possible, it is still able to harmonize edited attributes into realistic images, even when some patch embeddings are missing or conflicting. Our method shares the same goal as image "tokenization" methods, such as VQGAN [11] or the KL-autoencoder of Latent Diffusion Models [37], as it aims to autoencode an input

image into a collection of spatial embeddings that can faithfully reconstruct the original image. However, rather than adhering to the convolutional grid, our tokenization is spatially flexible, instead aligning to the semantically meaningful segments of the input image.

In summary, we propose a new image representation that supports

- spatial editing of the input image content
- faithful reconstruction of the input image with low runtime
- various image editing operations like object removal, inpainting, resizing, and rearrangement

2 Related Work

Image editing with diffusion models. In image editing, as opposed to image synthesis, there exists extra challenges in preserving the content of the input images. While text-to-image diffusion models are not designed to handle input images, several directions were developed to leverage the input content as inputs to the diffusion model, such as inverting the input image into the input noise map [30, 47], into the text embedding space [20], starting the denoising process from intermediate noise levels [29,37] or conditioning on particular modalities like depthmap, pose estimation, or image embedding [13,36,46,52]. Also, there exist methods to handle specialized tasks in image editing, such as inpainting [8,27,37, 50], text-guided modification [4,7], point-based editing [28,41], or customization from a few example images [22,38]. However, most existing methods suffer from either inability to change the spatial layout [20,29,30,41,47] or loss of detailed contents [4, 13, 38]. Most notably, Self-Guidance [9] proposes a way to enable spatial editing by caching attention maps and using them as guidance, but we observe that the edited outputs often fall off realism on many input images.

Image Editing with Autoencoders. The autoencoder design is a promising choice for image editing, where the input content is captured by the encoder and is assimilated with editing operations by the decoder to produce realistic images. Since the encoding process is a feed-forward pass of the encoder, the speed is significantly faster than optimization-based methods [1, 20, 38], enabling interactive editing processes. In GAN settings, Swapping Autoencoder [33] showed structure-preserving texture editing by decomposing the latent space into structure and style code. E4E [45] designed an encoder for a pretrained StyleGAN [19] generator. In diffusion, Diffusion Autoencoder [35] trains an encoder jointly with the diffusion-based decoder to capture the holistic content of the input image. However, there exists a fundamental trade-off between reconstruction accuracy and spatial editing capability, since the latent codes typically requires large spatial dimensions for good reconstruction, which cannot be easily edited with techniques like interpolation. Our method also falls under the autoencoder formulation, with an encoder processing individual image elements and a diffusion-based decoder mapping the edited image elements back into the pixel space. Still, we overcome the reconstruction-editing trade-off by representing the latent codes with continuous positional embeddings that are easily editable.

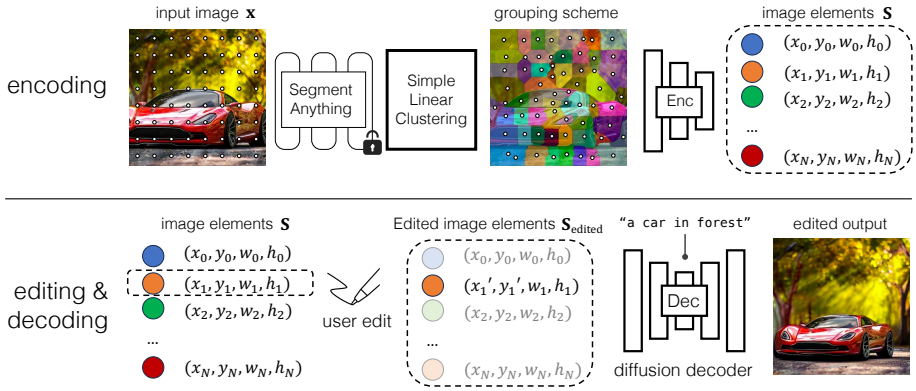

Fig. 2: Overview of our image editing pipeline. *(top)* To encode the image, we extract features from Segment Anything Model [21] with equally spaced query points and perform simple clustering to obtain grouping of object parts with comparable sizes, resembling superpixels [2]. Each element is individually encoded with our convolutional encoder and is associated with its centroid and size parameters to form image elements. *(bottom)* The user can directly modify the image elements, such as moving, resizing, or removing. We pass the modified image elements to our diffusion-based decoder along with a text description of the overall scene to synthesize a realistic image that respects the modified elements.

Layout Control in Generative Models Controlling the generation process with layouts has been an active topic in deep image generative models [3, 12, 18, 24–26, 31, 43, 44, 51, 53], including segmentation-to-image synthesis models [17, 32, 49], or hierarchically refining the output image [14]. In particular, BlobGAN [10, 48] exposes objects knobs that can be continuously moved and resized, similar to our method. However, most existing methods cannot be used for editing input images, since the layout conditioning is insufficient to represent all input contents. Our method aims to enable image editing by comprehensively encoding all input contents into image elements in the form of superpixels [2].

3 Method

Our goal is to learn a representation that allows for photo-realistic image editing, particularly with spatial manipulations. To achieve this, the high-level idea is to divide an image into a set of manipulatable entities, such that each entity captures a part of an object or a stuff class, which can be intuitively edited for diverse operations like deletion, composition, or spatial re-arrangement. In addition, we train a decoder that conditions on the embeddings of these entities to produce realistic output RGB images.

Our discussion will proceed in three sections. First, we present the definition of image elements that can be easily edited by the user. Second, we design an encoder to capture the content of each image element independently by training

an autoencoder with image elements as the bottleneck. Lastly, we replace the decoder of the autoencoder with a more powerful text-guided diffusion model to create realistic images from the conditioning provided by the image elements.

3.1 Image Elements

We aim to represent an image $\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$ with "image elements" that capture the content of the image while being amenable to editing at the same time. Specifically, each image element should correspond to an identifiable part of objects or "stuff" classes. Moreover, they should stay within the manifold of real image elements under editing operations such as deletion or rearrangement.

To this end, we perform grouping on the input image into disjoint and contiguous patches based on the semantic similarity and spatial proximity, represented as $\mathbf{A} = \{\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_N\}$, where $\mathbf{a}_n \in \mathbb{R}^{H_n \times W_n \times 3}$ is a cropped masked patch of the image. For reference, we operate on images of size $H \times W = 512 \times 512$, using $N = 256$ element, with an average element size of 1024 pixels.

Obtaining image elements. To divide the image into patches, we modify Simple Linear Iterative Clustering (SLIC) [2] to operate in the feature space of the state-of-the-art point-based Segmentation Anything Model (SAM) [21]. We start with N query points using 16×16 regularly spaced points on the image. SAM can predict segmentation masks for each query point, making it a plausible tool for predicting image elements. However, the final predicted segmentation masks are not suitable as editable image elements, since the segments tend to vary too much in shape and size, and extreme deviation from the regular grid is not amenable to downstream encoding and decoding. As such, we combine the predicted SAM affinity map $\mathbf{s}(m, n) \in [0, 1]$ with the Euclidean distance in spatial coordinates $\mathbf{d}(m, n)$, between pixel location index m and query point n. Each pixel m is grouped into a query element n

$$g(m) = \underset{n \in \{1,2,\cdots,N\}}{\arg\max} [\mathbf{s}(m, n) - \beta \cdot \mathbf{d}(m, n)], \tag{1}$$

where hyperparameter β is used to balance between feature similarity and spatial distance. The above formulation is equivalent to running one iteration of SLIC, thanks to the high-quality semantic affinity of the SAM feature space. According to $g(m)$, all pixels are then assigned to one of the N query elements, resulting in a set of disjoint super-pixels $\mathbf{A}$. We post-process each patch $\mathbf{a}_n$ by running connected components and selecting the largest region to ensure each patch is contiguous. In practice, this results in a small percentage of pixels ($\sim 0.1\%$), corresponding to the smaller connected components, being dropped.

Autoencoding image elements. Now that we defined the grouping scheme for the image elements, we learn to encode the appearance of each patch by training an auto-encoder to reconstruct the image with the image element as the bottleneck. That is, we design encoder $\mathcal{E}$ and decoder $\mathcal{D}$ to incorporate the information from all image elements and reconstruct the image.

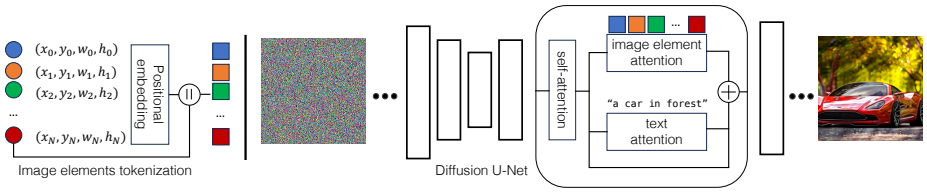

Fig. 3: Details of our diffusion-based decoder. First, we obtain positional embeddings of the location and size of the image elements, and concatenate them with the content embeddings to produce attention tokens to be passed to the diffusion model. Our diffusion model is a finetuned text-to-image Stable Diffusion UNet, with extra cross-attention layers on the image elements. The features from the text cross-attention layer and image element cross-attention layer are added equally to the self-attention features. Both conditionings are used to perform classifier-free guidance with equal weights.

3.2 Content Encoder

The goal of our encoding scheme is to disentangle the appearance and spatial information of the image elements, such that the elements can be later edited. This is achieved by obtaining an appearance embedding on each patch *separately*, to a convolutional encoder, such that the embedding is agnostic to its spatial location. To ensure the size parameters are decoupled from the feature, all patches $\mathbf{a}_n$ are resized to the same size before inputting to the encoder, and the patch embeddings are then obtained via the encoder. In addition, we collect patch properties $\mathbf{p}_n$ – centroid location (x_n, y_n) and bounding box size (w_n, h_n).

The encoder is trained to maximize the informational content in its embedding by jointly training a lightweight decoder $\mathcal{D}_{\text{light}}$ to reconstruct the input image with the Euclidean loss.

$$\mathcal{E}^* = \arg\min_{\mathcal{E}} \min_{\mathcal{D}_{\text{light}}} \ell_2\big(\mathbf{x}, \mathcal{D}_{\text{light}}(\mathbf{S})\big), \\ \text{where } \mathbf{S} = \{(\mathcal{E}(\mathbf{a}_n), \mathbf{p}_n)\}, \text{encoded image elements} \tag{2}$$

In Section 4.4, we show through an ablation study that training such a lightweight transformer decoder is beneficial for learning better features. The decoder is designed as a transformer, inspired by the Masked Auto-Encoder (MAE [15]) with 8 self-attention blocks. We insert 4 additional cross-attention layers in between to take the image elements as input. In addition, we input coordinate embeddings as the queries of the semantic decoder as a starting point.

3.3 Diffusion Decoder

While the auto-encoder aforementioned produces meaningful image elements, the synthesized image quality is limited, since it is only trained with the MSE loss, resulting in blurry reconstructions. Furthermore, when the image elements are edited by the user $\mathbf{S} \rightarrow \mathbf{S}_{\text{edited}}$, we also empirically observe that realism is further compromised. To produce photo-realistic images from the image elements, the

decoder should be powerful enough to fill in *missing* information, unspecified by the edited image elements. We base such a decoder on the pretrained Stable Diffusion [37] model, modifying it to condition on our image elements.

Stable Diffusion background. Diffusion models are probabilistic models that define a Markov chain of diffusion steps to learn a data distribution by gradually denoising a normally distributed variable. As one of the most successful instantiation of diffusion models in the text2image generation models family, Stable Diffusion learns the distribution in the latent space to reduce the computation while preserving the image synthesis quality. Stable Diffusion is composed of two key components, 1) a variational auto-encoder to embed a clean image into latent space $\mathbf{z}_0$, and 2) a base model $\mathcal{U}$ with parameters $\theta_{\mathcal{U}}$ performs denoising of a noisy latent $\mathbf{z}_t$, which is obtained by adding Gaussian noises over the clean latent $\mathbf{z}_0$, over timestep $t \in \{0, 1, \cdots, T\}$, with $T = 1000$,

Formally, the forward sampling process $q(\mathbf{z}_t|\mathbf{z}_0)$ at time step t is defined as,

$$q(\mathbf{z}_t|\mathbf{z}_0) = \sqrt{\bar{\alpha}_t}\mathbf{z}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{3}$$

where $\bar{\alpha}_t = \prod_k^t \alpha_k$ and $\alpha_1, \cdots, \alpha_t$ are pre-defined noise schedules. A sentence describing the picture is encoded to a text embedding $\mathbf{C}$ using an additional text encoder and then fused into the base model via cross-attention. The training target of the base model $\mathcal{U}$ amounts to the following equation,

$$\mathcal{L}_{SD} = \mathbb{E}_{\mathbf{z},\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I}),t}||\epsilon - \mathcal{U}(\mathbf{z}_t, t, \mathbf{C}; \theta_{\mathcal{U}})||_2^2. \tag{4}$$

During inference, a clean latent $\mathbf{z}_0$ can be generated by reversing the Gaussian process, going from pure Gaussian noise $\mathbf{z}_T \sim \mathcal{N}(0, 1)$ to less noisy samples $\mathbf{z}_{T-1}, \mathbf{z}_{T-2}, \cdots, \mathbf{z}_0$ step-by-step, conditioned on the input text embedding $\mathbf{C}$.

Incorporating image elements. The Stable Diffusion base model $\mathcal{U}$ is implemented with a UNet architecture, which contains a set of ResNet Blocks and Attention Blocks. The model takes a noisy latent $\mathbf{z}_t$ as input and predicts the corresponding noise $\hat{\epsilon}$. To incorporate the image element conditioning into the UNet model, we modify the attention blocks to jointly take the text embedding $\mathbf{C}$ and the image elements $\mathbf{S}$, as shown in Figure 3.

Specifically, for each layer with a cross attention to text embeddings in the original UNet architecture, we insert a new cross attention layer with parameters $\theta_{\mathcal{S}}$ to take the image elements. Both cross attention layers, one on text and the other on image elements, are processed using the output features of the previous self-attention layers. The output features of the cross attention are then added to the self-attention layer features with equal weights. With this new layer, the original training target in Equation 4 now becomes,

$$\mathcal{L}_{SD}^{\text{new}} = \mathbb{E}_{\mathbf{z},\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I}),t}||\epsilon - \mathcal{U}(\mathbf{z}_t, t, \mathbf{C}, \mathbf{S}; \theta_{\mathcal{U}}, \theta_{\mathcal{S}})||_2^2. \tag{5}$$

We initialize the set of parameters $\theta_{\mathcal{U}}$ using the pre-trained model and the new set of introduced parameters $\theta_{\mathcal{S}}$ are randomly initialized. Similar to the original Stable Diffusion where the text encoder is frozen, we freeze the parameters of both the content encoder and text encoder to ensure stability. We empirically

justify the design decisions with ablation studies in Section 4.4. We obtain an image $\hat{\mathbf{x}} = \mathcal{D}(\mathbf{z}_T, \mathbf{C}, \mathbf{S})$, where $\mathcal{D} \triangleq \bigcirc_{t=1}^{T} \mathcal{U}(\cdot, t, \cdot, \cdot)$ and $\mathbf{z}_T$ is randomly initialized noise. We apply the same guidance weight of 3 on both types of cross-attentions. More details are included in the supplementary materials.

Training with Image Element Dropout. While the above training scheme is effective for reconstructing the original input from unedited image elements, $\hat{\mathbf{x}} = \mathcal{D}(\mathbf{S})$, we observe that the realism quickly degrades when generated an edited image $\hat{\mathbf{x}}_{\text{edited}} = \mathcal{D}(\mathbf{S}_{\text{edited}})$, as the edited elements can introduce distributional discrepancies unseen in training, such as overlapping, missing elements, or gaps between them. Therefore, we reduce the discrepancy between training and test time editing by designing a dropout scheme of image elements during training.

In practice, we employ Semantic SAM [23], an improved version of SAM enabling segment images at any desired granularity, to obtain a database of object masks. Then a random object mask is overlaid on the input image, and image elements overlapping with the mask are dropped out. However, due to the unwanted correlation between object edges and image elements boundaries, the model tends to inpaint object boundaries aligned with that of the dropped image elements, which is less preferred in many cases. To address this, we propose ***random partition***, i.e., to divide image randomly to obtain the image elements. The insight behind is that the conditional probability of inpainting should ideally be independent of the image element partition. In practice, we simply obtain the random image elements partitioned from another sampled image.

Supporting editing operations. We show the following set of edits – ***delete***, ***move***, and ***resize***. As the encoded appearance features $\mathcal{E}(\mathbf{a}_n)$ are decoupled from their spatial properties $\mathbf{p}_n$, these operations can be obtained by appropriate manipulations on each. So we can easily delete a subset of image elements or edit the positions and sizes $\mathbf{p}_n$ of the image elements. For deletion, instead of removing the image elements, we zero out both appearance features and their spatial information (after positional embedding) to maintain uniform input length during training. When performing moving or resizing, the image elements that collide with the edited image elements are automatically deleted. Intuitively, a pixel in the image space is always covered by at most one image element.

4 Experiments

In this section, we show the rich examples on various image editing tasks, including object resizing, rearrangement, dragging, de-occlusion, object removal, and object variations.

4.1 Dataset and Training Details

Our dataset contains $3M$ images from the LAION Dataset [40]. For extracting image elements in Equation 1, we empirically find $\beta = 64$ yields a good balance between reconstruction quality and editability. We train the content encoder and transformer decoder for 30 epochs with MSE loss. Our diffusion decoder is built

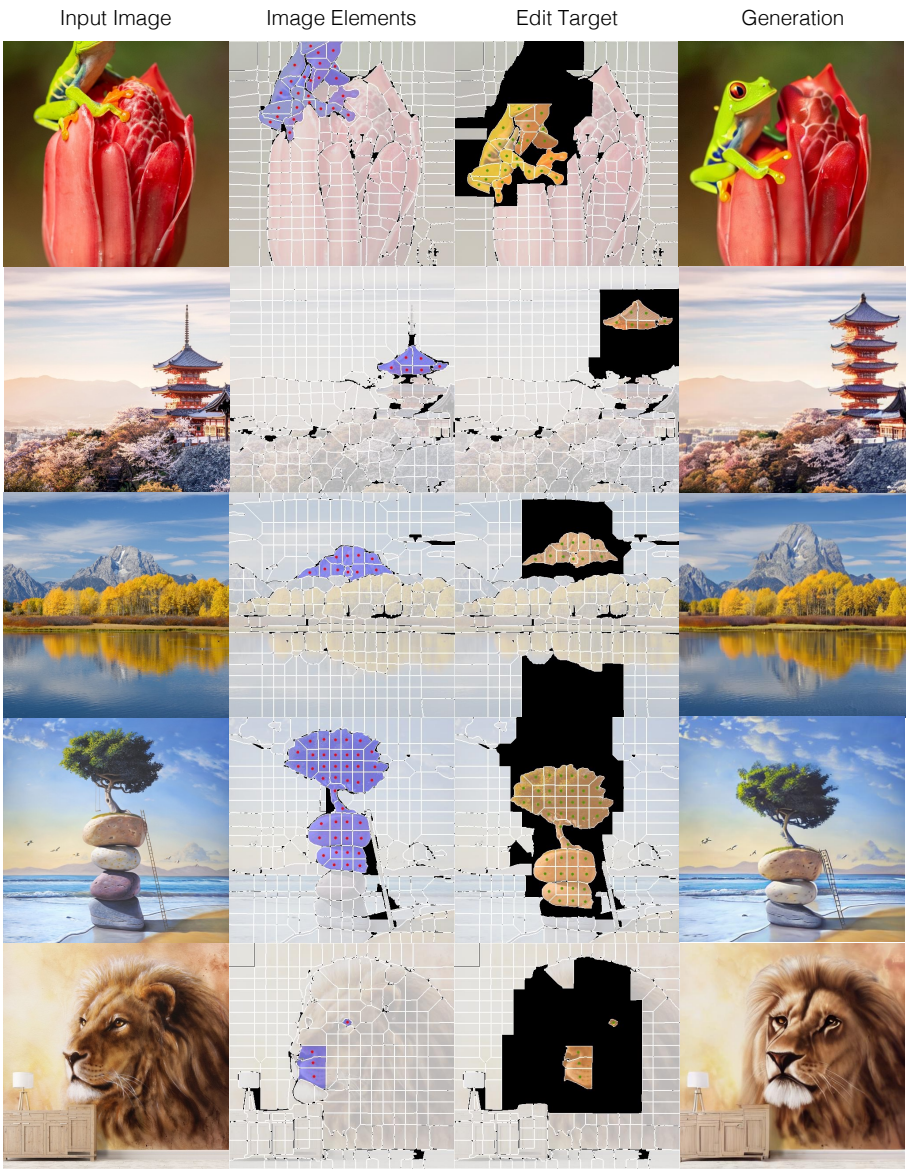

Fig. 4: The user can directly edit the image elements with simple selection, dragging, resizing, and deletion operations. The selected and edited elements are highlighted with red and green dots at the centroid of each element.

Fig. 5: The user can directly edit the image elements with simple selection, dragging, resizing, and deletion operations. The selected and edited elements are highlighted with red and green dots at the centroid of each element.

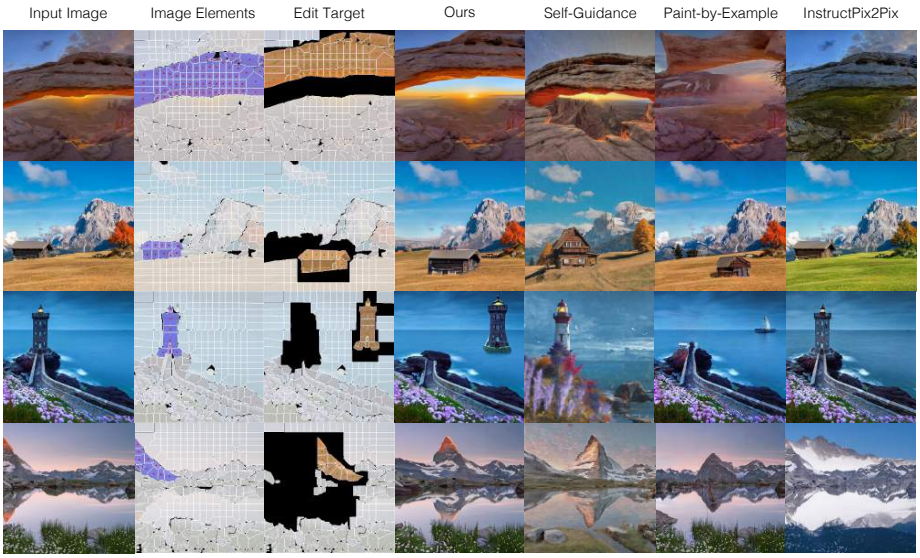

Fig. 6: Our method is compared to Self-guidance [9], Paint-by-Example [50], and InstructPix2Pix [5] on various edits. Our results attain superior results in preserving the details of the input and following the new edits. The baseline results show various failures, such as decline in image quality, floating textures, and unfaithful to the edits.

on Stable Diffusion v1.5, trained with the same losses as Stable Diffusion. We report the results after around $180k$ iterations. We use classifier-free guidance with equal weights on text and image element conditioning to generate our examples; $\epsilon(z, \mathbf{C}, \mathbf{S}) = \epsilon(z, \emptyset, \emptyset) + w * [\epsilon(z, \mathbf{C}, \mathbf{S}) - \epsilon(z, \emptyset, \emptyset)]$. The examples in the paper are generated with $w = 3.0$ using 50 sampling steps with the DDIM sampler. More details are presented in the supplementary materials.

4.2 Spatial Editing

To achieve spatial editing, the user directly modifies the image elements. Our diffusion-based decoder takes both image and text inputs to synthesize a realistic image that is faithful to the modified elements. As shown in Figure 4 and Figure 5, we observe a number of interesting applications that are mostly unattempted by existing diffusion models.

We compared our method with various representative approaches, namely, gradient-based method Self-Guidance [9], exemplar-based inpainting method Paint-by-Example [50], language instructed approach InstructPix2Pix [5] in Figure 6. Self-Guidance extends prompt-to-prompt [16] with additional gradient term to achieve spatial control for image editing. Though the method showcased interesting results on the stronger yet publicly unavailable Imagen [39] model, we could only compare our result on the released SDXL [37] in Figure 6. We observe that with small guidance strength, the editing operation is not respected,

Fig. 7: Object variation. Our method supports object variation by deleting some image elements (shown in black in Target Edit), and performing inpainting guided by text prompt and the remaining image elements, such as the "beak" element in the top row.

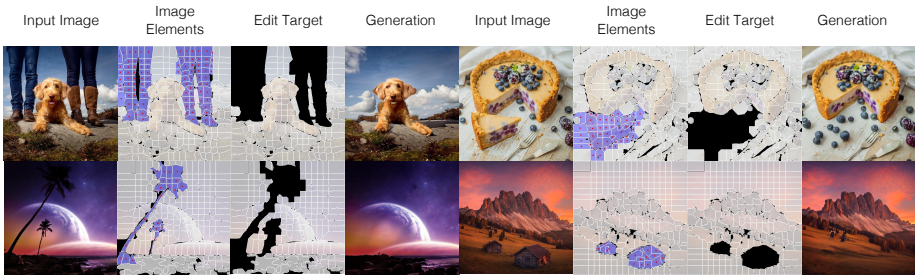

Fig. 8: Object removal. User selects elements to delete (shown in blue), and provide a text prompt pertaining to the background. Our diffusion decoder can generate content in the missing region (in black).

and with higher guidance, the realism is affected. We hypothesize that the self-guidance requires very strong image prior similar to Imagen and is sensitive to the hyper parameters. In contrast, our editing results are more faithful to the input image and the editing operations. We also compare with Paint-by-Example, an exemplar-guided image editing approach, by annotating images following the paper. To achieve spatial editing, we first crop a background region as the exemplar to inpaint the interesting area and then use the region of interest as the reference image to modify the target location. However, we empirically find the results usually lead to declined image quality for both stages. InstructPix2Pix is a language-based image editing method, which generates paired image editing examples using GPT3 [6] and then trains a diffusion model for image editing. Though the method produces realistic images, it tends to either not responds to

the edit or only modifies the global textures. In addition, it is also worth noting that using language only is naturally limited to achieve precise spatial editing.

Evaluating the quality of editing operation is a difficult task, as it is a combination of content preservation, adherence to task, and realism of the output images. Therefore, we run a Two-Alternative Forced Choice (2AFC) user study to quantify the quality of the editing results. We provide a pair of editing results with associated editing instructions, and ask the a user which image is better in terms of both image quality and faithfulness to the specified editing operation. As shown in Figure 9, our editing results are collected over 900 user judgments over three baselines, indicating that our method outperforms all baselines significantly.

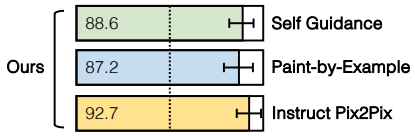

Fig. 9: Perceptual study where users are asked to choose which image better reflects both the editing and image quality.

4.3 Object Variations, Removal, and Composition

Our method naturally supports object variation and removal by inpainting the specified region, shown in Figure 7 and Figure 8. Since the model is conditioned on both the observed image elements and text, user can instruct model via text prompt to remove the object (inpaint with the background), or insert a new object in harmony with the context. Lastly, we can perform image composition with automatic harmonization by inserting elements from a different image, and removing the original elements in the overlapping region (Figure 1).

4.4 Ablation Studies

In this section, we analyze various design choices as presented in Table 1 and Figure 10. We compare both reconstruction and editing quality. For reconstruction, we provide each model with all image elements and run DDIM with 50 steps. For editing, similar to the user study conducted in Section 4.2, we present the user a pair of image editing results and ask the user which one is better regarding both faithfulness to the edit and image quality.

Our proposed model is composed of two stages, with the first stage training the content encoder (Section 3.2) with a transformer decoder, and the second stage training the diffusion decoder while freezing the content encoder (Section 3.3). The first question we study is: *why is staged training necessary*? To show this, instead of training the content encoder with a transformer decoder first, we jointly train the content encoder and diffusion decoder together. We observe this variant is worse in both image reconstruction and editing quality. To further justify our design, we still do staged-training but do not freeze the content encoder when training the diffusion decoder. Though it shows better reconstruction compared to the joint-training, the overall quality is still inferior

Staged Training	Freeze Content Encoder	Random Partition	MSE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	Preference VS ours ↑
✔	✔	✔	0.0069	22.98	0.6354	0.3376	10.82	-
✘	✘	✔	0.0138	19.74	0.5208	0.3697	11.91	34.2%
✔	✘	✔	0.0097	21.35	0.5962	0.3238	10.48	37.1%
✔	✔	✘	0.0066	23.15	0.6389	0.3262	9.75	27.3%

Table 1: Design differences. We study various design choices with both reconstruction and editing. "Preference VS ours" denotes the percentage of edited images that are preferred by MTurkers compared to our default setting. Our default setting achieves the best overall performance, both in image quality and in faithfulness to the editing.

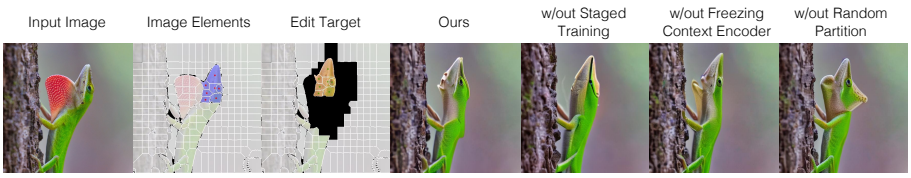

Fig. 10: Visualization of different design choices.

to our default setting. In terms of editing, the results are also less preferred compared to the default setting (37.1% compared to 62.9%).

Another trick for training the diffusion decoder is to overlay a partition to extract image elements. The reasoning for using random partitions, as opposed to the the actual partition of the image, is that the model are not supposed to learn the correlation of image elements partition and image edges. Visually, we find the model trained without random partition tends to in-paint object boundaries aligned with that of the dropped image elements, which is less preferred than continuing the content outside the mask. Table 1 shows only 27.3% of the editing results of the model without applying random partitions are more preferred.

5 Discussion and Limitations

We present the concept of editable image elements, which can be used to perform spatial editing on images using diffusion models. We show the inputs can be faithfully reconstructed, and our approach enables various editing operations like moving, resizing, de-occlusion, removal, or variations. Since the reconstruction quality is not perfect, editing user-provided high resolution images remains challenging. Moreover, while our framework supports spatial editing, the appearance embeddings of the image elements are still not easily editable.

Acknowledgements This work was started while Jiteng Mu was an intern at Adobe Research. This project was supported, in part, by NSF IIS-2303153, NSF CAREER Award IIS-2240014, Amazon Research Award, Cisco Faculty Award, Adobe Data Science Research Award, gifts from Qualcomm.

References

1. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan: How to embed images into the stylegan latent space? In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 4432–4441 (2019) 3
2. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(11), 2274–2282 (2012) 4, 5
3. Avrahami, O., Hayes, T., Gafni, O., Gupta, S., Taigman, Y., Parikh, D., Lischinski, D., Fried, O., Yin, X.: Spatext: Spatio-textual representation for controllable image generation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18370–18380 (2023) 4
4. Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: Learning to follow image editing instructions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18392–18402 (2023) 3
5. Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: Learning to follow image editing instructions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18392–18402 (2023) 11
6. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. In: Conference on neural information processing systems (NeurIPS) (2020) 12
7. Cao, M., Wang, X., Qi, Z., Shan, Y., Qie, X., Zheng, Y.: Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 22560–22570 (2023) 3
8. Chen, X., Huang, L., Liu, Y., Shen, Y., Zhao, D., Zhao, H.: Anydoor: Zero-shot object-level image customization. arXiv preprint arXiv:2307.09481 (2023) 3
9. Epstein, D., Jabri, A., Poole, B., Efros, A.A., Holynski, A.: Diffusion self-guidance for controllable image generation. In: Conference on Neural Information Processing Systems (NeurIPS) (2023) 3, 11
10. Epstein, D., Park, T., Zhang, R., Shechtman, E., Efros, A.A.: Blobgan: Spatially disentangled scene representations. In: European Conference on Computer Vision (ECCV). pp. 616–635 (2022) 4
11. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12873–12883 (2021) 2
12. Feng, W., He, X., Fu, T.J., Jampani, V., Akula, A.R., Narayana, P., Basu, S., Wang, X.E., Wang, W.Y.: Training-free structured diffusion guidance for compositional text-to-image synthesis. In: International Conference on Learning Representations (ICLR) (2022) 4
13. Gal, R., Arar, M., Atzmon, Y., Bermano, A.H., Chechik, G., Cohen-Or, D.: Encoder-based domain tuning for fast personalization of text-to-image models. ACM Transactions on Graphics (TOG) **42**(4), 1–13 (2023) 1, 3
14. Ge, S., Park, T., Zhu, J.Y., Huang, J.B.: Expressive text-to-image generation with rich text. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 7545–7556 (2023) 4
15. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16000–16009 (2022) 6

16. Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., Cohen-Or, D.: Prompt-to-prompt image editing with cross attention control. arXiv preprint arXiv:2208.01626 (2022) 11
17. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1125–1134 (2017) 4
18. Jahn, M., Rombach, R., Ommer, B.: High-resolution complex scene synthesis with transformers. arXiv preprint arXiv:2105.06458 (2021) 4
19. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8110–8119 (2020) 3
20. Kawar, B., Zada, S., Lang, O., Tov, O., Chang, H., Dekel, T., Mosseri, I., Irani, M.: Imagic: Text-based real image editing with diffusion models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6007–6017 (2023) 3
21. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. arXiv preprint arXiv:2304.02643 (2023) 4, 5
22. Kumari, N., Zhang, B., Zhang, R., Shechtman, E., Zhu, J.Y.: Multi-concept customization of text-to-image diffusion. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1931–1941 (2023) 1, 3
23. Li, F., Zhang, H., Sun, P., Zou, X., Liu, S., Yang, J., Li, C., Zhang, L., Gao, J.: Semantic-sam: Segment and recognize anything at any granularity. arXiv preprint arXiv:2307.04767 (2023) 8
24. Li, Y., Liu, H., Wu, Q., Mu, F., Yang, J., Gao, J., Li, C., Lee, Y.J.: Gligen: Open-set grounded text-to-image generation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 22511–22521 (2023) 4
25. Li, Z., Wu, J., Koh, I., Tang, Y., Sun, L.: Image synthesis from layout with locality-aware mask adaption. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 13819–13828 (2021) 4
26. Liu, N., Li, S., Du, Y., Torralba, A., Tenenbaum, J.B.: Compositional visual generation with composable diffusion models. In: European Conference on Computer Vision (ECCV). pp. 423–439 (2022) 4
27. Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., Van Gool, L.: Repaint: Inpainting using denoising diffusion probabilistic models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11461–11471 (2022) 3
28. Luo, G., Darrell, T., Wang, O., Goldman, D.B., Holynski, A.: Readout guidance: Learning control from diffusion features. arXiv preprint arXiv:2312.02150 (2023) 3
29. Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.Y., Ermon, S.: Sdedit: Guided image synthesis and editing with stochastic differential equations. In: International Conference on Learning Representations (ICLR) (2021) 1, 3
30. Mokady, R., Hertz, A., Aberman, K., Pritch, Y., Cohen-Or, D.: Null-text inversion for editing real images using guided diffusion models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6038–6047 (2023) 3
31. Mu, J., De Mello, S., Yu, Z., Vasconcelos, N., Wang, X., Kautz, J., Liu, S.: Coordgan: Self-supervised dense correspondences emerge from gans. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2022) 4

32. Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2337–2346 (2019) 4
33. Park, T., Zhu, J.Y., Wang, O., Lu, J., Shechtman, E., Efros, A.A., Zhang, R.: Swapping autoencoder for deep image manipulation. In: Conference on Neural Information Processing Systems (NeurIPS) (2020) 3
34. Parmar, G., Kumar Singh, K., Zhang, R., Li, Y., Lu, J., Zhu, J.Y.: Zero-shot image-to-image translation. In: ACM SIGGRAPH 2023 Conference Proceedings. pp. 1–11 (2023) 1
35. Preechakul, K., Chatthee, N., Wizadwongsa, S., Suwajanakorn, S.: Diffusion autoencoders: Toward a meaningful and decodable representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10619–10629 (2022) 3
36. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125 (2022) 1, 3
37. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10674–10685 (2022) 1, 2, 3, 7, 11
38. Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., Aberman, K.: Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 22500–22510 (2023) 1, 3
39. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. In: Conference on Neural Information Processing Systems (NeurIPS) (2022) 1, 11
40. Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al.: Laion-5b: An open large-scale dataset for training next generation image-text models. In: Conference on Neural Information Processing Systems (NeurIPS) (2022) 8
41. Shi, Y., Xue, C., Pan, J., Zhang, W., Tan, V.Y., Bai, S.: Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. arXiv preprint arXiv:2306.14435 (2023) 3
42. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502 (2020) 1
43. Sun, W., Wu, T.: Learning layout and style reconfigurable gans for controllable image synthesis. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(9), 5070–5087 (2021) 4
44. Sylvain, T., Zhang, P., Bengio, Y., Hjelm, R.D., Sharma, S.: Object-centric image generation from layouts. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). vol. 35, pp. 2647–2655 (2021) 4
45. Tov, O., Alaluf, Y., Nitzan, Y., Patashnik, O., Cohen-Or, D.: Designing an encoder for stylegan image manipulation. ACM Transactions on Graphics (TOG) **40**(4), 1–14 (2021) 3
46. Voynov, A., Abernan, K., Cohen-Or, D.: Sketch-guided text-to-image diffusion models. arXiv preprint arXiv:2211.13752 (2022) 3
47. Wallace, B., Gokul, A., Naik, N.: Edict: Exact diffusion inversion via coupled transformations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 22532–22541 (2023) 3

48. Wang, Q., Wang, Y., Birsak, M., Wonka, P.: Blobgan-3d: A spatially-disentangled 3d-aware generative model for indoor scenes. arXiv preprint arXiv:2303.14706 (2023) 4
49. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8798–8807 (2018) 4
50. Yang, B., Gu, S., Zhang, B., Zhang, T., Chen, X., Sun, X., Chen, D., Wen, F.: Paint by example: Exemplar-based image editing with diffusion models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18381–18391 (2023) 3, 11
51. Yang, Z., Liu, D., Wang, C., Yang, J., Tao, D.: Modeling image composition for complex scene generation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7764–7773 (2022) 4
52. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 3836–3847 (2023) 1, 3
53. Zhao, B., Meng, L., Yin, W., Sigal, L.: Image generation from layout. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8584–8593 (2019) 4