





# Toward Interactive Image Inpainting via Robust Sketch Refinement

Chang Liu , Shunxin Xu , Jialun Peng , Kaidong Zhang , and Dong Liu , *Senior Member, IEEE*

**Abstract**—One tough problem of image inpainting is to restore complex structures in the corrupted regions. It motivates interactive image inpainting which leverages additional hints, e.g., sketches, to assist the inpainting process. A sketch is simple and intuitive for end users to provide, but meanwhile has free forms with much randomness. Such randomness may confuse the inpainting models, and incur severe artifacts in completed images. To better facilitate image inpainting with sketch guidance, we propose a two-stage image inpainting system, termed SketchRefiner. The first stage of our approach serves as a data provider that simulates real sketches and derives the capability of sketch calibration from the simulated data. In the second stage, our approach aligns the sketch guidance with the inpainting process so as to elevate image inpainting with sketches. We also propose a real-world test protocol to address the evaluation of inpainting methods upon practical applications with user sketches. Experimental results on three prevailing benchmark datasets, i.e., CelebA-HQ, Places2, and ImageNet, and the proposed test protocol demonstrate the state-of-the-art performance of our approach, and its great potentials upon real-world applications. Further analyses illustrate that our approach effectively utilizes sketch information as guidance and eliminates the artifacts due to the free-form sketches.

**Index Terms**—Image inpainting, sketch-based image inpainting, sketch refinement, image editing, generative adversarial network (GAN).

## I. INTRODUCTION

IMAGE inpainting refers to the task of completing a given “corrupted” image in the hope of making the completed image consistent and photo-realistic [2]. In some scenarios, an image was indeed corrupted, e.g., due to transmission error; in other scenarios, users may intentionally “corrupt” a complete image, e.g. to remove an unwanted foreground object. Thus, image inpainting has revealed great practical value in real-world applications, e.g. old photo restoration, image editing, watermark removal, object removal, and so on. Among these applications,

one of the most challenging scenarios in real practice is to recover complex structures in the missing areas, which motivates studies on the interactive image inpainting task that allows users to provide hints for the inpainting process to alleviate the difficulty. Among numerous user-interacted physical medias, sketch has stood out as a simple and intuitive way to express creative ideas from users. By simply depicting several lines, sketch is able to highlight the outlines of important elements for the corrupted image and convey demands from users straightforwardly, which has emerged as a prominent attractive research direction in the field of image inpainting.

Existing methods for interactive image inpainting can be divided into two main categories, i.e., traditional methods and learning-based methods, respectively. Among traditional methods, some studies attempted to introduce user interactions as structural guidance [3], [4] to assist the image inpainting process. Nevertheless, these methods are heuristic and fail to recover plausible textures in the corrupted areas. Recently, learning-based methods gradually became dominant and revealed more plausible performance than traditional ones, where these methods derive outstanding capability of recovering corrupted regions with large-scale training data. Despite this, it is difficult for existing learning-based methods to effectively handle real sketches for interactive image inpainting, since there are no available image-sketch pairs in sufficient scale, and collecting them is also resource-consuming. To address such difficulties, a series of studies [1], [5], [6], [7], [8], [9], [10], [11], [12] are thus motivated to simulate hand-drawn sketches with automatic solutions, e.g., edge detection models [13], [14]. However, such methods neglect the modal characteristics of user-drawn sketches, where it is tough for untrained users to depict pixel-wise precise outlines as the ones in the edge map. On treating the input sketch as the edge map in inference, the inpainting model will be confused by incorrect guidance and incur severe artifacts similar to the examples presented in Fig. 1, thereby revealing less practical value in real-world applications. To facilitate interactive image inpainting upon real-world applications, the gap between sketch-like inputs and existing image inpainting models is expected to be bridged.

In this paper, we present a paradigm to address the aforementioned limitations and bridge the existing gap of the sketch-based interactive image inpainting task, namely SketchRefiner. SketchRefiner is a two-stage system that calibrates the input sketch and facilitates the inpainting process with elaborated guidance based two components, namely robust sketch refinement (RSR) and sketch-modulated image inpainting (SII),

Manuscript received 7 October 2023; revised 4 February 2024; accepted 14 May 2024. Date of publication 17 May 2024; date of current version 16 October 2024. This work was supported in part by the Natural Science Foundation of China under Grant 62036005 and in part by the Fundamental Research Funds for the Central Universities under Grant WK3490000006. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Metin Sezgin. (Corresponding author: Dong Liu.)

The authors are with the MOE Key Laboratory of Brain-Inspired Intelligent Perception and Cognition, University of Science and Technology of China, Hefei 230027, China (e-mail: lc980413@mail.ustc.edu.cn; sxu@mail.ustc.edu.cn; pjl@mail.ustc.edu.cn; richu@mail.ustc.edu.cn; dongeliu@ustc.edu.cn).

This article has supplementary downloadable material available at <https://doi.org/10.1109/TMM.2024.3402620>, provided by the authors.

Digital Object Identifier 10.1109/TMM.2024.3402620

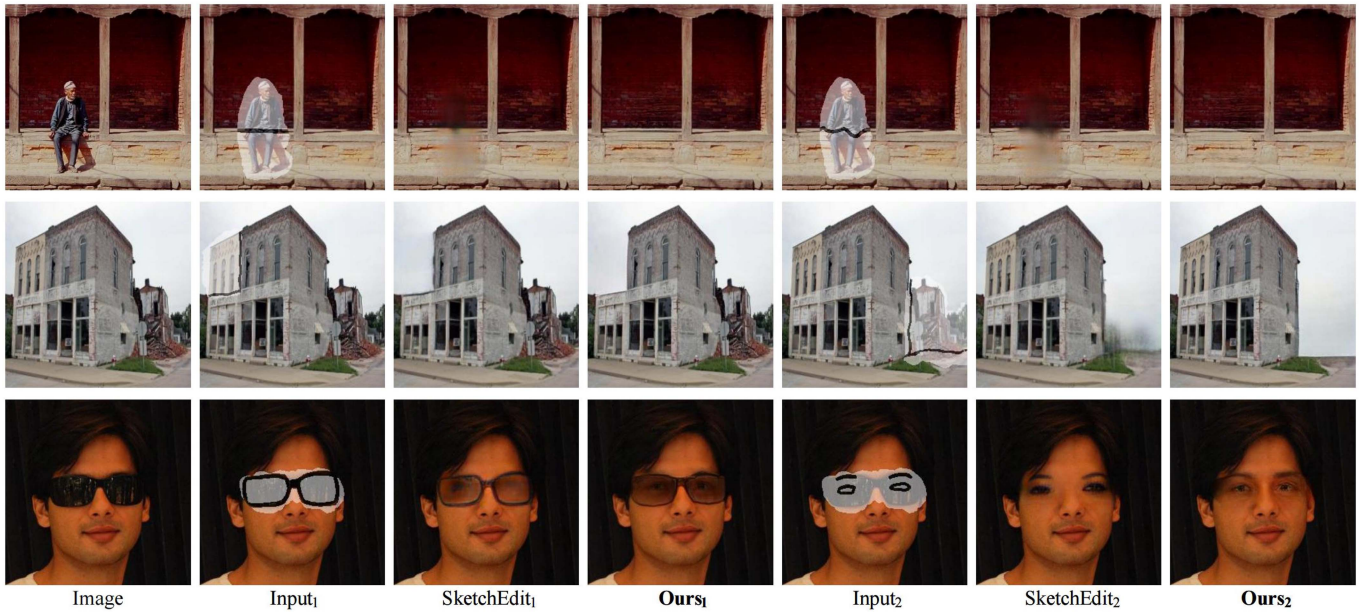


Fig. 1. Interactive image inpainting results of scene editing (top two rows) and face manipulation (the third row), produced by SketchEdit [1] and our approach. Given user-provided sketches and masks (the second and fifth columns), SketchEdit [1] tries to use the sketches as if they were edges in the missing area, thereby producing noticeable artifacts. In reverse, our method uses the sketches as coarse guiding information, thus reflects user intentions in the produced results and meanwhile tolerates certain randomness of user input.

respectively. Specifically, RSR performs the sketch refinement process, equipped with deformable sketch simulation (DSS) and sketch refinement network (SRN), which provides various deformed sketches and learns the sketch calibration process along with the simulated data, respectively. SII aligns the refined sketch with the inpainting process and recovers the missing textures in the corrupted regions with the assistance of structural guidance. Therefore, SketchRefiner is able to facilitate the image inpainting process with sketch guidance and eliminates artifacts in the inpainted results brought by the free forms of hand-drawn sketches. Experimental results on three public benchmark datasets, i.e., CelebA-HQ [15], Places2 [16], and ImageNet [17] demonstrate the guaranteed performance of our approach compared to existing state-of-the-art methods. For further evaluation of our approach on real applications, we propose a real-world test protocol with manually annotated masks and sketches for interactive image inpainting, which indicates the great potentials of our approach in a series of practical applications. Generally speaking, our contributions are three-fold:

- We re-investigate the challenges of sketch-based interactive image inpainting, by proposing a two-stage system called SketchRefiner to bridge the existing gap between sketch-like inputs and image inpainting models.
- SketchRefiner utilizes sketch guidance to facilitate the inpainting process with RSR and SII, which learns the structural calibration process with simulated data and aligns the sketch guidance with the inpainting process, respectively.
- We establish a real-world test protocol with manually annotated masks and sketches for sketch-based interactive image inpainting. Experimental results on three benchmark datasets, i.e., CelebA-HQ, Places2, ImageNet, and

the proposed test protocol illustrate that our approach obtains state-of-the-art performance and reveals great potentials on real-world applications.

Our code and the proposed test protocol are available at <https://github.com/AlonzoLeeeooo/SketchRefiner>.

## II. RELATED WORK

### A. Image Inpainting

Existing methods for image inpainting can be divided into two categories, i.e., traditional methods and learning-based methods, whose details are illustrated in the following text.

Among traditional methods, the most typical solution for image inpainting is to solve partial differential equations so as to predict the missing pixels in the corrupted regions according to the valid ones. Other studies, namely patch-based methods [18], [19], [20], attempted to fill the missing areas with retrieved similar patches from the target image [21] or a pre-defined database [22]. Nevertheless, methods that solve partial differential equations struggle to recover corrupted images with large missing areas, while patch-based methods suffer from the limitations that patches with similar textures are difficult to find, especially when the corrupted areas contain stochastic textures or complicated structures.

With the bloom of deep learning approaches, learning-based methods have become prominent and significantly promote the image inpainting task. Specifically, generative adversarial network (GAN), which optimizes a generator and a discriminator in an adversarial manner, have achieved great success in the field of image inpainting. For example, Context Encoder [23] is the first study that applies GAN on image inpainting, which

established the foundation paradigm of GAN-based methods using pixel-wise loss and adversarial loss to optimize the model. Iizuka et al. [24] equip the GAN with dilated convolution layers and discriminators with different receptive fields, while Yu et al. [25] adopt a contextual attention layer to perform image inpainting in a coarse-to-fine manner. Zheng et al. [26] propose to generate diverse inpainting results with a conditional VAE model, and Liu et al. [27] facilitate image inpainting with a feature equalizing encoder-decoder. Peng et al. [28] boost the inpainting process with various structural information generated by a hierarchical VQ-VAE, and Suvorov et al. [29] propose a fast Fourier convolution with large image receptive field to recover highly corrupted images. Zhang et al. [30] propose a concatenated U-Net inpainting network to enhance the texture and structure reconstruction for image inpainting. Other studies put emphasis on tackling image inpainting with novel network architectures, e.g., transformer- and diffusion-based models. Transformer-based methods demonstrate their superiority in modeling long-term correlation, where studies for pluralistic image completion [31], [32], bridging the interactions of different image regions [33], and large hole image inpainting [34] are promoted and achieve great success. Meanwhile, diffusion-based methods provided an alternative for image inpainting through iteratively de-noising to recover the corrupted image, where RePaint [35] is the first work to adopt pre-trained unconditional diffusion models for image inpainting. However, both transformer- and diffusion-based methods require large computational resources to implement and suffer from slow inference speed, which fails to meet the requirement of real-time interactions for image inpainting. In this paper, we address the image inpainting task following GAN-based methods.

### B. Interactive Image Inpainting

The topic of conducting the inpainting process interactively has been investigated across different data modalities, e.g., video [36], [37], image, etc. Particularly, interactive image inpainting attempts to facilitate the image inpainting process with additional useful interactions, e.g., text [38], [39], local binary pattern (LBP) [40], and so on. Among numerous interactive media, sketch has attracted much research interest for its simplicity and intuitive way of presenting user interactions. Early traditional studies [3], [4] utilized structural information to help the inpainting process by allowing users to provide it interactively, which reveals less practical usage due to the ill-presenting performance of patch-based methods. As learning-based methods become dominant, it is natural to explore their possibilities along with user-provided sketches. However, existing datasets lack available data pairs between real sketches and images, while collecting one is resource-consuming, which motivates existing studies to simulate hand-drawn sketches through automatically generated alternatives, e.g., edge maps extracted by edge detectors [13], [14]. Specifically, Yu et al. [5] propose to conduct generative image inpainting via free-form sketches. Liu et al. [41] present a structure-guided framework for image inpainting to maintain the neighborhood consistency and structure coherence of the inpainted region. Yang et al. [9]

perform sketch-guided face generation both locally and globally by calibrating the input face sketch according to their style. Liu et al. [11] edit images with interactions of sketch and color guidance. Zeng et al. [1] predict masks according to the input sketch and perform mask-free image inpainting. Nevertheless, these methods are presented based on the strong assumption that the interacted sketch aligns with the detected edge map by default, and neglect the scenarios that users may fail to provide accurate sketches for the missing areas, thereby resulting in unnatural artifacts as the examples shown in Fig. 1.

### C. Sketch Refinement

Early research for sketch refinement [42], [43] have proven that sketches are closely related to edges, both of which are visually closed outlines of objects. The major difference between sketches and edges is that edges are pixel-wise corresponding to sharp intensity gradients, while sketches are more diversified and abstract. Although several efforts [6], [8], [12], [44], [45], [46] have been motivated to recover sketches from edges, these methods can only complete disconnected structures that are corrupted by simulated masks, and fail to handle real sketches where misalignment occurs inside the masked areas, thus becoming less practical for real-world applications. Among the aforementioned discussed studies, DeepFill-v2 [5], DeepPS [9], and SketchEdit [1] are the most related ones to our study. Nevertheless, DeepPS is only limited to face images, while DeepFill-v2 and SketchEdit require pixel-wise preciseness in the user-provided sketches, which still struggle to satisfy the demands of real-world applications. On the contrary, our approach is capable of eliminating the artifacts brought by free-form sketches and provides precise structural guidance for the image inpainting process.

## III. APPROACH

Fig. 2 illustrates the overall pipeline of our approach, which consists of two stages, namely, robust sketch refinement (RSR) and sketch-modulated image inpainting (SII). RSR aims to refine the input sketch and provide more accurate sketch guidance for the inpainting process, and SII learns to leverage the sketch guidance to recover the corrupted image. Specifically, RSR consists of two components, namely, deformable sketch simulation (DSS) and sketch refinement networks (SRN), where DSS simulates the randomness of free-form sketches during user interactions and provides various sketches for SRN, and SRN calibrates the coarse sketches into more accurate structural guidance. SII involves latent sketch aligner (LSA) and sketch conditional inpainter (SCI), where LSA aligns the refined sketch guidance with the inpainting process, and SCI utilizes the aligned guidance to assist the process of generating the final result. In detail, given the input image  $\mathcal{I}$ , we first obtain the masked image  $\mathcal{I}_m$  by composing  $\mathcal{I}$  with the binary mask  $\mathcal{M}$ . In training, the DSS process  $f_{dss}(\cdot)$  produces the input sketch  $\mathcal{S}$  for SRN through  $\mathcal{S} = f_{dss}(\mathcal{I}, \mathcal{M}, \mathcal{D})$ , so as to have SRN learned the sketch refinement process. Herein,  $\mathcal{D}$  denotes a number that controls the magnitude of sketch deformation. Then, the SRN  $f_{srn}(\cdot)$  calibrates  $\mathcal{S}$  into the refined sketch  $\hat{\mathcal{S}}$  according to  $\mathcal{M}$  and  $\mathcal{I}_m$ . The



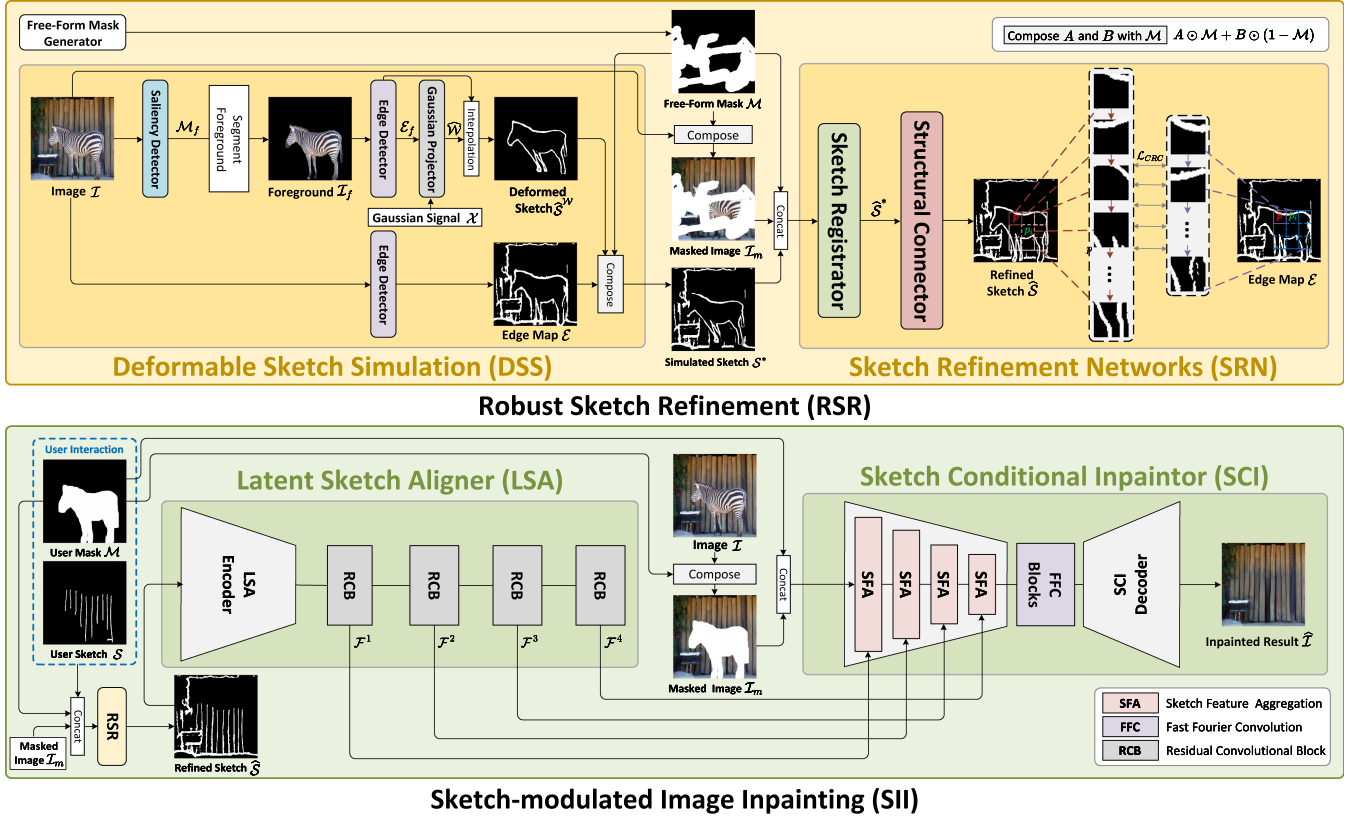


Fig. 2. The overall pipeline of our proposed approach with two main processes, namely, robust sketch refinement (RSR) and sketch-modulated image inpainting (SII). RSR contains deformable sketch simulation (DSS) and sketch refinement networks (SRN), while SII involves latent sketch aligner (LSA) and sketch conditional inpainter (SCI). Herein, we present an example image for better demonstration, where the goal of this example case is to remove the zebra from the original image and the user sketch aims to provide hints for the inpainting networks to recover the fence in the background.

overall process of RSR is formulated by

$$\hat{S} = f_{srn}(\mathcal{I}_m, \mathcal{M}, \mathcal{S}) \quad (1)$$

The SII process aligns  $\hat{S}$  through the LSA  $f_{lsa}(\cdot)$  with the inpainting process and incorporates the aligned guidance into the SCI  $f_{sci}(\cdot)$ , written as

$$\hat{\mathcal{I}} = f_{sci}(\mathcal{I}_m, \mathcal{M}, f_{lsa}(\hat{S})) \quad (2)$$

In RSR, SRN is optimized with our proposed cross-region correlation (CRC) loss  $\mathcal{L}_{CRC}^N$  with  $N$  referring to the region size. In SII, we utilize pixel-wise loss  $\mathcal{L}_{\ell_1}$ , adversarial loss  $\mathcal{L}_{adv}$  [47], feature matching loss  $\mathcal{L}_{fm}$ , and high receptive field perceptual loss  $\mathcal{L}_{hrf}$  [29]. The aforementioned loss functions are formulated as

$$\mathcal{L}_{RSR} = \sum_{N=1}^{N_r} \lambda_1^N \mathcal{L}_{CRC}^N \quad (3)$$

and

$$\mathcal{L}_{SII} = \lambda_2 \mathcal{L}_{\ell_1} + \lambda_3 \mathcal{L}_{adv} + \lambda_4 \mathcal{L}_{fm} + \lambda_5 \mathcal{L}_{hrf} \quad (4)$$

where  $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ , and  $\lambda_5$  are hyper-parameters to balance the contributions of the losses. Therefore, the final loss function  $\mathcal{L}$  of our approach is formulated as

$$\mathcal{L} = \mathcal{L}_{RSR} + \mathcal{L}_{SII} \quad (5)$$

In the following texts, we introduce each component according to the aforementioned processing sequence in detail.

#### A. Robust Sketch Refinement

The goal of our approach is to calibrate the input sketch so as to facilitate the inpainting process with more accurate sketch guidance. However, one of the biggest challenges is the absence of large-scale image-sketch pairs for the training of the sketch calibration process, where existing methods still struggle to handle real sketches using detected edge maps as alternatives. To address such challenges, we propose a novel method named robust sketch refinement (RSR) to simulate real sketches automatically and learn the sketch refinement process based on two sequentially connected components, namely deformable sketch simulation (DSS) and sketch refinement networks (SRN), respectively. In the following texts, we illustrate the aforementioned components in detail.

1) *Deformable Sketch Simulation*: DSS serves as the data provider that offers a series of simulated sketches for the sketch refinement process. In doing so, we utilize edge maps with different degrees of deformation to simulate the randomness of free forms in user-drawn sketches. Specifically, given a ground truth image  $\mathcal{I}$ , we use a saliency detector to detect the foreground mask  $\mathcal{M}_f$  from  $\mathcal{I}$  and use  $\mathcal{M}_f$  to segment the foreground  $\mathcal{I}_f$



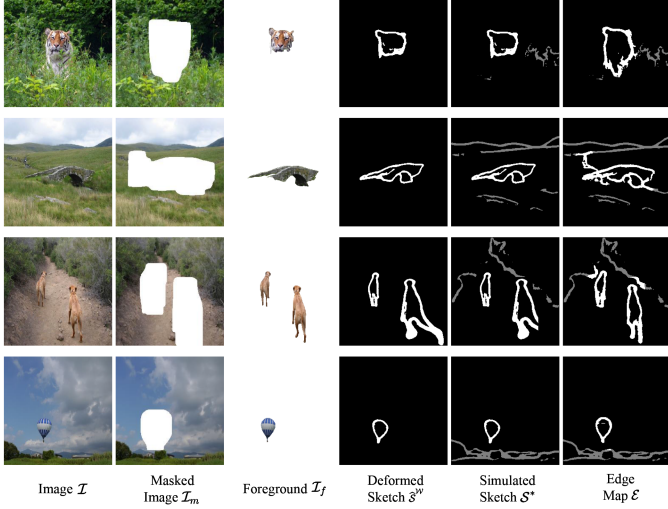


Fig. 3. Visualization of the intermediate results in DSS. Lines in the mask regions of the simulated sketch  $\mathcal{S}^*$  and the edge map  $\mathcal{E}$  are highlighted in white, whereas the others are in grey. We also show all the mathematical symbols of the intermediate results to offer a better demonstration of the DSS process.

from  $\mathcal{I}$ . Then we use an off-the-shelf edge detector to produce the foreground edge map  $\mathcal{E}_f$  from  $\mathcal{I}_f$ .<sup>1</sup> Afterward, we conduct a Gaussian projector<sup>2</sup>  $f_{gp}(\cdot)$  to obtain a warping map  $\widehat{\mathcal{W}}$  from a random Gaussian signal  $\mathcal{X}$ ,<sup>3</sup> where  $\widehat{\mathcal{W}}$  is then used to project each pixel in  $\mathcal{E}_f$  onto another position using bilinear interpolation  $f_{bi}(\cdot)$ , resulting in the deformed sketch  $\widehat{\mathcal{S}}^w$ . The overall process is formulated by

$$\begin{aligned}\widehat{\mathcal{W}} &= f_{gp}(\mathcal{X}) \odot \mathcal{D} \\ \widehat{\mathcal{S}}^w &= f_{bi}(\mathcal{E}_f, \widehat{\mathcal{W}})\end{aligned}\quad (6)$$

and

$$\mathcal{S}^* = \widehat{\mathcal{S}}^w \odot \mathcal{M} + \mathcal{E} \odot (1 - \mathcal{M}) \quad (7)$$

where  $\odot$  refers to the Hadamard product between matrices.  $\mathcal{D}$  is a number that controls the magnitude of sketch deformation, which is sampled from a uniform distribution  $U[\mathcal{D}_{\min}, \mathcal{D}_{\max}]$ .  $\mathcal{M}$  refers to randomly generated masks using free-form masking algorithm [5]. In training, we randomly vary  $\mathcal{D}$  and  $\mathcal{M}$  to ensure the diversity in simulated sketch  $\mathcal{S}^*$ . In Fig. 3, we show the visualization of the intermediate results in DSS.

2) *Sketch Refinement Networks*: Once the simulated sketch is obtained, the next step is to train SRN in learning the process of refining the input sketch into more useful structural guidance. In doing so, SRN aligns the input sketch with the missing edge lines of the corrupted image, and then connects the structural vacancies in the intermediate sketch based on two components,

<sup>1</sup>Herein, we use U<sup>2</sup>-Net [48] and BDCN [14] as the saliency detector and edge detector, respectively.

<sup>2</sup>The Gaussian projector is conducted as a convolutional layer with random Gaussian parameters as its model weights.

<sup>3</sup> $\mathcal{X}$  represents a randomly generated white Gaussian noise that has the same size as  $\mathcal{I} \in \mathbb{R}^{H \times W \times C}$ , where  $H$ ,  $W$ , and  $C$  indicate the height, width, and number of channels of  $\mathcal{I}$ , respectively. We use  $\mathcal{X}$  to obtain the randomness of free forms during user interactions.

namely sketch registrator (SR) and structural connector (SC).<sup>4</sup> To build SRN, we utilize gated convolutional blocks [5] to conduct the architecture of SR so as to selectively utilize information in the valid region of the corrupted image, and use vanilla convolutional blocks in SC to enhance the structural coherence of the intermediate sketch. Particularly, we send the concatenation of the masked image  $\mathcal{I}_m$ , the binary mask  $\mathcal{M}$ , and the input sketch  $\mathcal{S}$  into SR to generate the intermediate sketch  $\widehat{\mathcal{S}}^*$ , formulated by

$$\widehat{\mathcal{S}}^* = f_{sr}(\mathcal{I}_m, \mathcal{M}, \mathcal{S}) \quad (8)$$

where  $f_{sr}(\cdot)$  denotes SR. Note that  $\mathcal{S}$  refers to  $\mathcal{S}^*$  in training and denotes the user-provided sketch in inference.

Although SR provides a strong registration process for sketch refinement, there are still structural vacancies in the intermediate sketch, where these vacancies will mislead the inpainting process and cause structural incoherence in the inpainted result. To address the aforementioned limitation, the SC  $f_{sc}(\cdot)$  further eliminates the vacancies in the intermediate sketch by learning to connect them, which ensures the preciseness in the refined sketch. Practically, we send the intermediate sketch  $\widehat{\mathcal{S}}^*$  into SC to generate the final refined sketch  $\widehat{\mathcal{S}}$ , where the process is written as

$$\widehat{\mathcal{S}} = f_{sc}(\widehat{\mathcal{S}}^*) \quad (9)$$

To optimize the sketch refinement process, existing studies use the pixel-wise loss (normally L1/L2 distance) to optimize the sketch refinement process, expecting the networks to reconstruct the input sketch into the corresponding edge map. However, such methods are insufficient for the sketch refinement task, since calibrating the sketch usually requires region-level information where structures in different regions are highly correlated to each other, e.g., sketch registration. Therefore, we propose a cross region correlation (CRC) loss to utilize the interaction between regions and facilitate the sketch refinement process with their mutual information. Given the refined sketch  $\widehat{\mathcal{S}}$ , we attempt to compute the local means of  $\widehat{\mathcal{S}}$  within a sliding grid  $p_i$  in size of  $N^2$ , formulated as

$$\bar{\mathcal{S}}_{p_i}^r = \frac{1}{N^2} \sum_{j=1}^N \sum_{i=1}^N v_{i,j} \quad (10)$$

where  $\bar{\mathcal{S}}_{p_i}^r$  represents the local means of refined sketch within  $p_i$ ,  $v_{i,j}$  denotes the  $i$ -th column and  $j$ -th row pixel value within  $p_i$ . Similarly, we calculate the local means of input sketch  $\mathcal{S}$  and edge map  $\mathcal{E}$  within  $p_i$ , written as  $\bar{\mathcal{S}}_{p_i}$  and  $\bar{\mathcal{E}}_{p_i}$ , respectively. Next, CRC loss compares the local means within the sliding region  $p_i$  and a fixed region  $p$  in a formulation of the cross-correlation (CC) function. Specifically, for  $N = 1$ , which represents the pixel-wise correlation, we set  $\mathcal{L}_{CRC}^N$  as the Euclidean distance between different pixels, which is

$$\mathcal{L}_{CRC}^1(\widehat{\mathcal{S}}, \mathcal{E}) = \|\widehat{\mathcal{S}} - \mathcal{E}\| \quad (11)$$

<sup>4</sup>In Section I of our appendix, Tables I and II illustrate the detailed architectural designs of SR and SC, respectively.

For  $N > 1$ , we write the CRC loss in a formulation of cross correlation, formulated by<sup>5</sup>

$$\begin{aligned} \mathcal{L}_{CRC}^N(\hat{\mathcal{S}}, \mathcal{E}) &= -CC(\hat{\mathcal{S}}, \mathcal{E}) \\ &= -\sum_{p \in \Omega} \frac{\sum_{p_i \in p} \|\bar{\mathcal{S}}_{p_i}^r - \bar{\mathcal{E}}_p\| \|\bar{\mathcal{E}}_{p_i} - \bar{\mathcal{E}}_p\|}{\left(\sum_{p_i \in p} \|\bar{\mathcal{S}}_{p_i}^r - \bar{\mathcal{E}}_p\|\right) \left(\sum_{p_i \in p} \|\bar{\mathcal{E}}_{p_i} - \bar{\mathcal{E}}_p\|\right)} \end{aligned} \quad (12)$$

where  $\Omega$  represents the overall regions of  $\hat{\mathcal{S}}$  and  $\|\cdot\|$  represents the Euclidean distance.  $p_i$  iterates over all pixels within  $p$  and  $p$  iterates over  $\Omega$ . In training, we optimize SRN in the formulation of (5), where SRN learns the sketch refinement process from different region levels.

### B. Sketch-Modulated Image Inpainting

Once RSR has calibrated the input sketch, SII aims to inpaint the corrupted image with the assistance of the refined sketch. To incorporate the sketch guidance into the inpainting process, most existing studies send the sketch guidance into an additional channel of the inpainting networks, so that the sketch guidance is integrated by learning the inter-information between channels. However, such studies suffer from the misalignment between input sketch in the pixel space and image information in the feature space, where the sketch guidance diminishes as networks go deeper and contributes little to deep features in the inpainting process after downsampling and pooling. To address such misalignment, we conduct the SII process along with two components, namely latent sketch aligner (LSA) and sketch conditional inpainter (SCI).<sup>6</sup> Details of the aforementioned components are illustrated as follows.

1) *Latent Sketch Aligner*: To inpaint the corrupted image using the sketch guidance, the first step is to align the inpainting process with the refined sketch. In doing so, LSA projects the refined sketch onto a feature space and aligns the projected features with the corresponding image features in SCI. Specifically, given the refined sketch  $\hat{\mathcal{S}}$ , LSA first encodes  $\hat{\mathcal{S}}$  into a series of latent features, formulated by

$$\{\mathcal{F}_1^s \dots \mathcal{F}_{N_s}^s\} = f_{lsa}(\hat{\mathcal{S}}) \quad (13)$$

where  $f_{lsa}$  represents LSA and  $\mathcal{F}_i^s$  ( $i \in \{1, 2, \dots, N_s\}$ ) refers to the extracted feature from the  $i$ -th layer with  $N_s$  features in total. For further explanation, LSA encodes  $\hat{\mathcal{S}}$  into  $\{\mathcal{F}_1^s \dots \mathcal{F}_{N_s}^s\}$  and aligns each individual  $\mathcal{F}_i^s$  with the corresponding feature in  $i$ -th layer of SCI.

2) *Sketch Conditional Inpaintor*: The final step is to recover the missing textures in the corrupted images with the assistance of aligned sketch guidance. To achieve so, we propose SCI based on LaMa [29] to complete the corrupted image according to the sketch guidance and SFA to facilitate the feature aggregation process of SCI. Particularly, SCI consists of a generator and

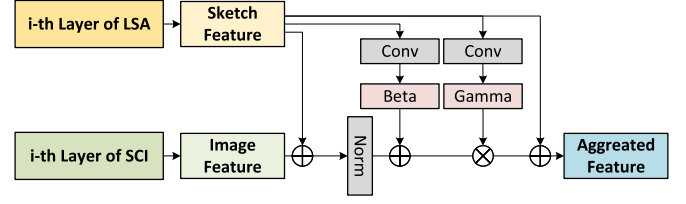


Fig. 4. Detailed architectural design of SFA block.

discriminator, where we conduct the generator in an encoder-decoder architecture that incorporates the sketch guidance with SFA blocks and completes the corrupted regions, and we use the default discriminator in LaMa. Given the corrupted image  $\mathcal{I}_m$ , binary mask  $\mathcal{M}$ , and aligned sketch features  $\{\mathcal{F}_1^s \dots \mathcal{F}_{N_s}^s\}$ , we send the concatenation of  $\mathcal{I}_m$  and  $\mathcal{M}$  into the encoder and inject  $\{\mathcal{F}_1^s \dots \mathcal{F}_{N_s}^s\}$  into corresponding block, formulated by

$$\mathcal{F}^i = f_{sci}^e(\mathcal{I}_m, \mathcal{M}; \mathcal{F}_1^s \dots \mathcal{F}_{N_s}^s) \quad (14)$$

where  $\mathcal{F}^i$  denotes the encoded input feature and  $f_{sci}^e$  represents the encoder of SCI. Herein, Fig. 4 illustrates the detailed architectural design of SFA block, where the  $i$ -th SFA block in  $f_{sci}^e$  aggregates the sketch feature  $\mathcal{F}_i^s$  by

$$\hat{\mathcal{F}}_i^v = \gamma_i(\mathcal{F}_i^s) \odot \frac{(\mathcal{F}_i^v + \mathcal{F}_i^s) - \mu_i}{\sigma_i} + \beta_i(\mathcal{F}_i^s) + (\mathcal{F}_i^v + \mathcal{F}_i^s) \quad (15)$$

where  $\hat{\mathcal{F}}_i^v$  denotes the processed image feature.  $\gamma_i(\mathcal{F}_i^s)$  and  $\beta_i(\mathcal{F}_i^s)$  are trainable tensors that adaptively learn the scale and shift of the aggregation process.  $\mu_i$  and  $\sigma_i$  are mean and standard deviation of  $(\mathcal{F}_i^v + \mathcal{F}_i^s)$ , which are used to normalize the added features. The intermediate layers are conducted with fast Fourier convolution blocks, and the decoder consists of upsampling layers, which process  $\mathcal{F}^v$  and predict the missing pixels in the corrupted regions, formulated by

$$\hat{\mathcal{I}} = f_{sci}^d(\mathcal{F}^v) \quad (16)$$

where  $\hat{\mathcal{I}}$  and  $f_{sci}^d(\cdot)$  denote the inpainted result and the decoder of SCI, respectively. Therefore, SCI restores the corrupted regions according to the sketch guidance. In training, we optimize LSA and SCI with  $\mathcal{L}_{SII}$  in the formulation of (4) following the default setup of LaMa [29].

## IV. EXPERIMENTS

### A. Datasets

We conduct our experiments on public benchmark datasets for evaluation with simulated data, and collect a real-world test protocol with manually annotated masks and sketches for evaluation on sketch-based real-world applications. Details of the public benchmarks and the real-world test protocol are illustrated in the following texts.

1) *Public Benchmarks*: We conduct our approach on three conventional benchmark datasets of image inpainting, which are CelebA-HQ [15], Places2 [16], and ImageNet [17]. CelebA-HQ dataset is widely used to perform image inpainting upon face images, where we split 28,000 images as the training set, 1,000 images as the validation set, and 1,000 images as the test set,

<sup>5</sup>A greater cross-correlation value (i.e.,  $CC(\hat{\mathcal{S}}, \mathcal{E})$ ) indicates a better alignment between the refined sketch and edge map, where we set the cross-correlation value negative to formulate the loss function.

<sup>6</sup>In Section I of our appendix, Tables III and IV illustrate the detailed architectural designs of LSA and SCI, respectively.

respectively. We use the Places2-Standard split of Places2 to address the challenging scene-level image inpainting, which consists of about 180,000 images in its training set. We randomly sample 10,000 images from the default validation and test split of Places2 to formulate our validation set and test set. We also conduct our experiments on ImageNet with object images to perform applications such as object completion or editing, where ImageNet contains about 120,000 images for training. Similarly, we randomly sample 10,000 images to formulate the validation and test sets on ImageNet. In inference, we utilize simulated sketches by DSS for evaluation on the public benchmark datasets.

2) *Real-World Test Protocol*: To evaluate the performance of inpainting methods upon real-world applications, we build a sketch-based test protocol with hand-drawn sketches and masks. Specifically, we collect 40 unseen face images from CelebA-HQ to demonstrate applications of portrait images, and 80 images (40 scene images and 40 object images) from ECSSD [49] to perform applications on natural images. For face images, we manually annotate the mask for each image by covering common editing regions, e.g., eyes, mouth, hair, etc. For scene images, we annotate the masks based on the saliency masks in ECSSD, where the final masks cover partial or whole objects to perform scene editing or object removal. For sketches, we ask different users to complete the edges in the corrupted region according to the pre-defined masks. Both sketches and masks are produced using the application SketchBook on iPad with an Apple pencil. By doing so, the test protocol covers a series of applications, including scene editing, face editing, object removal, etc. In inference, we use the user masks and sketches for evaluation on the test protocol.<sup>7</sup> We also propose an interactive demo that allows users to experience the interactive image inpainting process.<sup>8</sup>

## B. Baselines and Evaluation Metrics

We compare our method with both state-of-the-art image inpainting and image editing methods. For comparison with the baseline, we compare our approach with LaMa [29]. For comparison with previous sketch-based image inpainting or editing methods, we choose DeepFill-v2 [5], DeepPS [9], and SketchEdit [1]. Also, we compare our approach with ZITS [12] which serves as a strong competitor and achieves the state-of-the-art performance on edge-guided image inpainting.

For evaluation on public benchmark datasets, we use conventional metrics including peak signal-to-noise ratio (PSNR), structural similarity (SSIM), and fr chet inception distance (FID) [50] so as to measure the reconstruction quality of the inpainted results. Since there is no ground truth on the real-world test protocol, we evaluate the results using qualitative metrics, including inception score (IS) [51], and style loss [52] following SketchEdit [1]. We also conduct a user study on the real-world test protocol to perform a human evaluation on all compared methods. Each user is presented with inpainted results of different methods according to the same input, and requested to rank the results according to their quality. Eventually, we calculate

<sup>7</sup>In Section II of our appendix, Fig. 2 demonstrates some examples of the proposed test protocol.

<sup>8</sup>In Section III of our appendix, Fig. 3 presents our proposed interactive demo.

the top-1 percentage of human preferences to measure which method is mostly preferred.

## C. Implementation Details

In the following texts, we illustrate the implementation details of our approach following our pipeline sequence and other compared methods.

In RSR, we set  $k = 11$  for the Gaussian projector in DSS and uniformly sample  $\mathcal{D}$  from  $U(25, 200)$ . SRN is conducted with sequentially connected SR and SC. We use Adam [54] for 500k steps of optimization with a learning rate of  $2e - 4$  and a batch size of 10. We try different hyper-parameter settings on ImageNet and use the one with highest performance on the validation set, where we use  $N = 1$  and 9 with  $\lambda_1^1 = 1, \lambda_1^9 = 0.4$  for SR, and the same  $N$  with  $\lambda_1^1 = 1, \lambda_1^9 = 0.9$  for SC in (5). It is worth noting that the trained SRN is able to directly generalize to unseen datasets (i.e., CelebA-HQ [15] and Places2 [16]) without retraining. In SII,  $N_s$  is set to 4 to extract features in different dimensions from sketches. For optimization of LSA and SCI, we use Adam [54] optimizer with a batch size of 25 and learning rates of the generator and the discriminator as  $3e - 4$  and  $1e - 4$ , respectively, where the learning rates are reduced to half in the middle of the training process. We train LSA and SCI for 400k steps on CelebA-HQ, and 800k steps on Places2 and ImageNet. As for the loss function in (4), we follow the default setup of LaMa [29] with  $\lambda_2 = 10, \lambda_3 = 10, \lambda_4 = 100$ , and  $\lambda_5 = 30$ . In training and inference, all images, masks, sketches, and edge maps are resized to  $256 \times 256$  resolution, where the pixel values of sketches and edge maps are binarized into 0 or 1.

For implementation of other compared methods, we follow their default setup reported in their paper to achieve their best performance. We retrain DeepFill-v2 [5] for 500k steps, LaMa [29] for 800k steps, ZITS [12] for 800k steps on Places2 [16] and ImageNet [17] to achieve sufficient optimization, and reimplement these methods with steps in half on CelebA-HQ [15].<sup>9</sup> For DeepPS and SketchEdit, we use their released model weights for evaluation.<sup>10</sup> For fair comparison in training and inference, we use the same edge detector as ours to extract edge maps from corresponding ground truth images for DeepFill-v2, ZITS, and SketchEdit. Also, for fair comparison between SketchEdit and other compared methods, we use the same masks as others for evaluation instead of the original estimated masks.

## D. Comparison With Previous Studies

1) *Quantitative Comparison*: Table I reports the results of quantitative comparison with simulated data on CelebA-HQ [15], Places2 [16], and ImageNet [17] test set, and Table II reports the comparison on the real-world test protocol using qualitative metrics. There are several observations. First,

<sup>9</sup>Since our evaluation do not focus on the inpainting setting with large corrupted regions that requires 1 M steps of optimization, training LaMa with 800k steps is sufficient to reproduce its best performance on small mask setting.

<sup>10</sup>The model weights and inference code of DeepPS [9] and SketchEdit [1] are available at <https://github.com/VITA-Group/DeepPS> and <https://github.com/zengxianyu/sketchedit>.



TABLE I  
QUANTITATIVE COMPARISON USING PSNR, SSIM, AND FID [50] ON PUBLIC BENCHMARK DATASETS, INCLUDING CELEBA-HQ [15], PLACES2 [16] AND IMAGENET [17]

| Method          | CelebA-HQ         |                   |                  | Places2           |                   |                  | ImageNet          |                   |                  |
|-----------------|-------------------|-------------------|------------------|-------------------|-------------------|------------------|-------------------|-------------------|------------------|
|                 | PSNR <sup>↑</sup> | SSIM <sup>↑</sup> | FID <sup>↓</sup> | PSNR <sup>↑</sup> | SSIM <sup>↑</sup> | FID <sup>↓</sup> | PSNR <sup>↑</sup> | SSIM <sup>↑</sup> | FID <sup>↓</sup> |
| DeepPS [9]      | 20.59             | 0.739             | 72.74            | -                 | -                 | -                | -                 | -                 | -                |
| DeepFill-v2 [5] | 22.06             | 0.788             | 26.78            | 24.42             | <u>0.872</u>      | 6.71             | 20.46             | 0.745             | 22.34            |
| LaMa [29]       | <u>23.48</u>      | <u>0.825</u>      | 19.13            | 24.17             | 0.871             | 6.57             | <u>21.25</u>      | <u>0.752</u>      | <u>14.31</u>     |
| SketchEdit [1]  | 20.21             | 0.759             | 41.28            | 22.08             | 0.806             | 11.12            | -                 | -                 | -                |
| ZITS [12]       | 22.79             | 0.812             | <u>18.79</u>     | <u>24.50</u>      | 0.871             | <u>6.20</u>      | 20.98             | 0.743             | 17.71            |
| Ours            | <b>23.90</b>      | <b>0.831</b>      | <b>16.32</b>     | <b>24.99</b>      | <b>0.877</b>      | <b>5.53</b>      | <b>22.16</b>      | <b>0.769</b>      | <b>9.53</b>      |

Herein, the best and second best results are highlighted in boldface and underlines. “-” stands for unavailable results because corresponding methods did not perform experiments on that dataset.

TABLE II  
QUANTITATIVE COMPARISON USING QUALITATIVE METRICS (I.E., IS [51] AND STYLE LOSS [52]) ON THE PROPOSED REAL-WORLD TEST PROTOCOL

| Method          | Face Editing    |                              |                              | Object Removal  |                              |                              |
|-----------------|-----------------|------------------------------|------------------------------|-----------------|------------------------------|------------------------------|
|                 | IS <sup>↑</sup> | SL <sub>1</sub> <sup>↓</sup> | SL <sub>2</sub> <sup>↓</sup> | IS <sup>↑</sup> | SL <sub>1</sub> <sup>↓</sup> | SL <sub>2</sub> <sup>↓</sup> |
| DeepPS [9]      | 2.973           | 0.0039                       | 0.0154                       | -               | -                            | -                            |
| DeepFill-v2 [5] | 3.149           | 0.0018                       | 0.0078                       | 4.516           | 0.0081                       | 0.0393                       |
| LaMa [29]       | <u>3.287</u>    | 0.0023                       | 0.0107                       | 4.692           | 0.0078                       | 0.0381                       |
| SketchEdit [1]  | 3.058           | 0.0033                       | 0.0123                       | 4.669           | 0.0089                       | 0.0479                       |
| ZITS [12]       | 3.268           | 0.0019                       | 0.0098                       | 4.711           | <u>0.0074</u>                | <u>0.0369</u>                |
| Ours            | <b>3.339</b>    | <b>0.0013</b>                | <b>0.0076</b>                | <b>4.840</b>    | <b>0.0067</b>                | <b>0.0354</b>                |

Herein, the best and second best results are highlighted in boldface and underlines. “SL<sub>1</sub>” and “SL<sub>2</sub>” denote the style loss computed between features extracted from the first and second ReLU layers of VGG [53].

TABLE III  
TOP-1 PERCENTAGE OF HUMAN PREFERENCES ON DIFFERENT MODELS ON THE REAL-WORLD TEST PROTOCOL

| Methods         | Test Protocol |               |               |
|-----------------|---------------|---------------|---------------|
|                 | Face          | Scene         | Total         |
| DeepPS [9]      | 10.61%        | -             | -             |
| DeepFill-v2 [5] | 10.61%        | 11.36%        | 11.56%        |
| LaMa [29]       | 12.12%        | 15.91%        | 14.97%        |
| SketchEdit [1]  | 16.67%        | 7.955%        | 12.24%        |
| ZITS [12]       | <u>18.18%</u> | <u>22.73%</u> | <u>21.77%</u> |
| Ours            | <b>31.81%</b> | <b>42.05%</b> | <b>39.46%</b> |

Herein, the best and second best results are highlighted in boldface and underlines.

our approach achieves state-of-the-art on all public benchmark datasets, suggesting the promising performance of SketchRefiner for applications of face, scene and object images. Second, our approach outperforms all other methods on the collected real-world test protocol, where our approach produces results with the closest style similarity to natural images, indicating the superiority of SketchRefiner while handling real-world applications. Third, by comparing our approach to the baseline model (i.e., LaMa), our approach reveals significant improvements with the assistance of sketch guidance, which confirms the effectiveness of facilitating image inpainting with user sketches. Fourth, by comparing our approach with state-of-the-art edge-based methods (i.e., ZITS) and sketch-based methods (i.e., DeepPS, DeepFill-v2, and SketchEdit), these methods struggle to deal with both simulated and real sketches that are misaligned in the pixel level, where our approach better utilizes sketches for the inpainting process.

2) *Qualitative Comparison*: We perform the qualitative comparison with selected samples from public benchmarks and the test protocol, where we demonstrate the results of face images and natural images in Fig. 5 and Fig. 6, respectively.<sup>11</sup> Also, we present the visualization of refined sketches and the corresponding inpainting results in Fig. 7 to demonstrate the effect of the sketch refinement process in our approach. It is observed that LaMa fails to complete the corrupted image accordingly (e.g., the fourth row in Fig. 5) and struggles to restore

complex backgrounds without user guidance (e.g., the fourth row in Fig. 6), where our approach effectively guides the inpainting process with sketches and produces plausible results. For edge-based methods, ZITS treats the input sketch as edge map and generates misguided results due to the inaccurate edges in user sketches (e.g., the second row in Fig. 5). For sketch-based methods, DeepPS fails to calibrate the user sketches correctly which leads to severe artifacts in the inpainted results (e.g., the first and third rows in Fig. 5). DeepFill-v2 and SketchEdit request for pixel-wise precise sketch guidance and struggle to produce plausible result if the input sketch is not elaborately drawn (e.g., the first and second rows in Fig. 6). On the contrary, our approach is able to effectively refine the input sketch so as to facilitate the inpainting process with more accurate structural guidance, thereby resulting in more photo-realistic results.

3) *Human Evaluation*: We compare the results of human evaluation upon all compared methods in Table III. It is observed that users overwhelmingly preferred the generated results by our approach than other methods, which reveals the same trend as the aforementioned quantitative and qualitative comparison. This observation confirm the effectiveness and superiority of our approach for interactive image inpainting, especially upon real applications that interacts via user sketch.

#### E. Ablation Studies

In this section, we analyze the effect of the RSR and SII processes in our approach. We first explore the effect of different components in RSR, which includes DSS, SR, SC, and CRC

<sup>11</sup>In Section IV in our appendix, Fig. 3 and 4 demonstrate more results on applications of face and natural images, respectively.

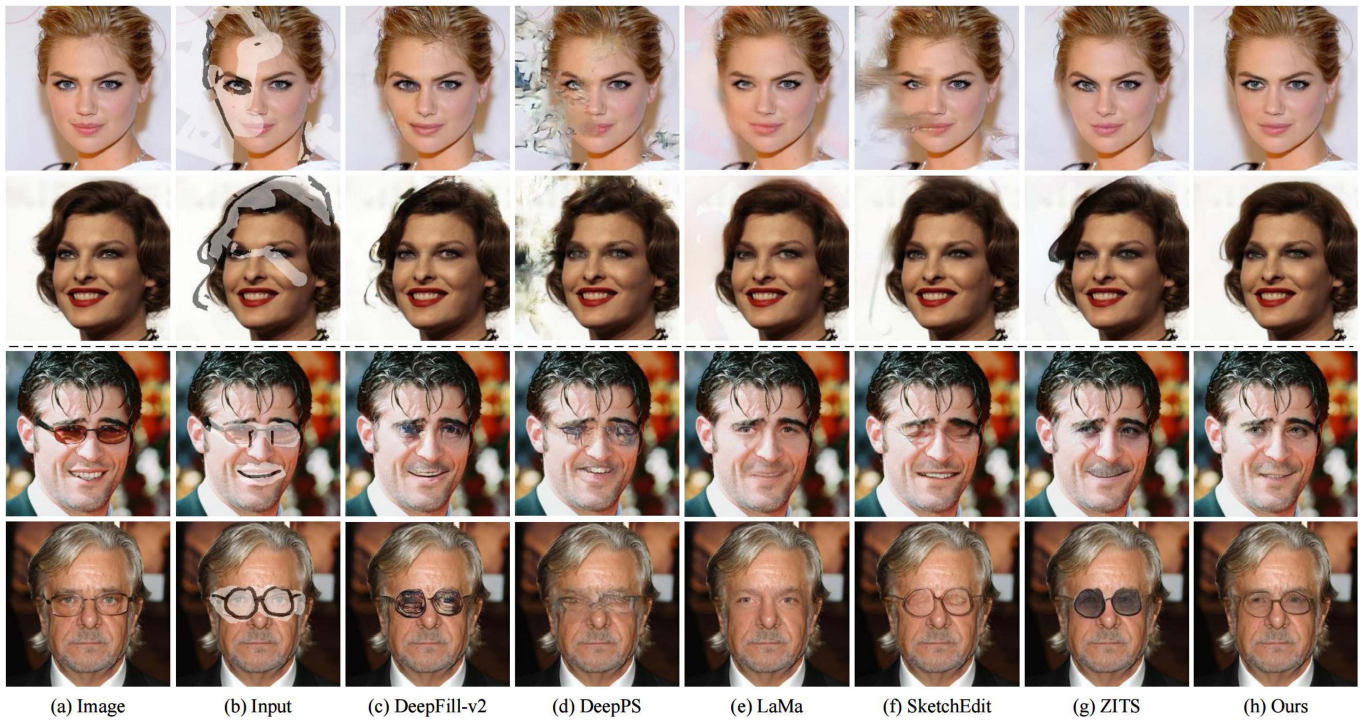


Fig. 5. Qualitative results of different models with simulated sketches (top two rows) and user-drawn sketches (bottom two rows) on applications of face images. Note that (b) represents the visualization of masked image and sketch.

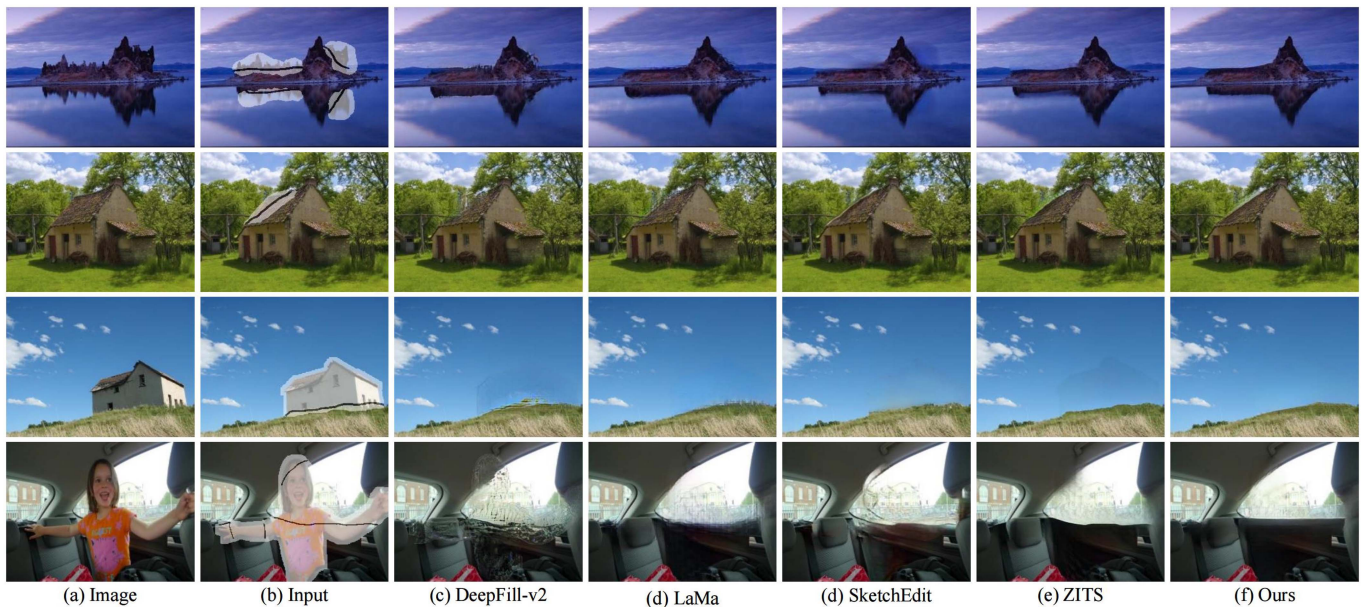


Fig. 6. Qualitative results of different models with simulated sketches (top two rows) and user-drawn sketches (bottom two rows) on applications of scene and object images. Note that (b) represents the visualization of masked image and sketch.

loss. Then, we investigate the effect of the overall RSR process on both our approach as well as other methods. Finally, we analyze the effect of different components in SII, including LSA, the generator, and SFA.

1) *Effect of Components in RSR*: We explore the effect of different components in RSR following the pipeline sequence, where details are illustrated in the following text.

a) *DSS*: To investigate the effect of DSS, we try an alternative setting that directly utilizes detected edge map for simulation. Table IV compares the results on ImageNet [17] test set, where “Ours (edge)” denotes the aforementioned setting. It is observed that “Ours (DSS)” significantly outperforms “Ours (edge)”, which confirms the effectiveness of our sketch simulation strategy compared to previous solutions. This might be



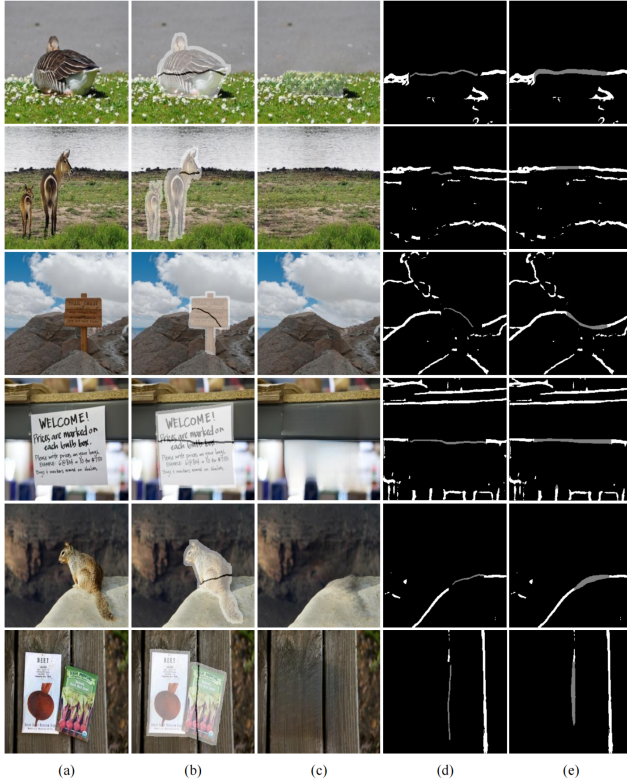


Fig. 7. Visualization of refined sketches and corresponding inpainting results. (a) Image, (b) input, (c) inpainted result, (d) sketch, and (e) refined sketch. Note that (b) represents the visualization of masked image and sketch. In (d) and (e), grey strokes represent sketch in the mask regions.

TABLE IV  
QUANTITATIVE COMPARISON FOR ABLATION STUDIES OF DSS ON  
IMAGENET [17] TEST SET

| Methods     | PSNR $\uparrow$ | SSIM $\uparrow$ | FID $\downarrow$ |
|-------------|-----------------|-----------------|------------------|
| Ours (edge) | 20.72           | 0.723           | 23.58            |
| Ours (DSS)  | <b>22.16</b>    | <b>0.769</b>    | <b>9.53</b>      |

Herein, the best results are highlighted in boldface. “Ours (edge)” represents directly using edge maps detected by BDCN [14] to simulate real sketches and “Ours (DSS)” denotes our approach with DSS.

explained that directly using detected edge maps fails to effectively simulate real sketches, where the model generates misguided results when given misaligned sketches in inference. The observation also addresses the superiority of our approach (i.e., using DSS for simulation) compared to previous solutions (i.e., using detected edge maps for simulation), where our approach provides a more elaborated automatic simulation solution for sketch-based interactive image inpainting.

*b) SC and CRC Loss:* We explore the effect of SC and CRC loss by comparing the performance of whether using them or not, where the comparisons are presented in Table V and Fig. 8. By comparing the first two rows in Table V, (c) and (d) in Fig. 8, one can see that the use of SC brings performance gain in all metrics and generates results with more explicit edges, where SC helps the sketch refinement process to handle the structural vacancies in the intermediate sketch and further facilitates the

TABLE V  
QUANTITATIVE RESULTS FOR ABLATION STUDIES OF DIFFERENT COMPONENTS  
IN RSR (I.E.,  $\mathcal{L}_{CRC}$ , SC, AND SR) ON CELEBA-HQ [15] TEST SET,  
CORRESPONDING TO FIG. 8

| Components of SRN   |    |    | CelebA-HQ       |                 |                  |
|---------------------|----|----|-----------------|-----------------|------------------|
| $\mathcal{L}_{CRC}$ | SC | SR | PSNR $\uparrow$ | SSIM $\uparrow$ | FID $\downarrow$ |
|                     |    | ✓  | 23.60           | 0.828           | 20.81            |
|                     | ✓  | ✓  | 23.93           | 0.831           | 16.66            |
| ✓                   | ✓  | ✓  | <b>24.99</b>    | <b>0.877</b>    | <b>5.54</b>      |

Herein, the best results are highlighted in boldface.

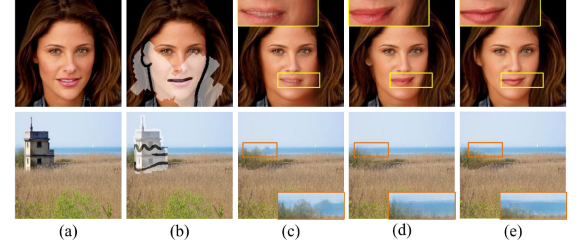


Fig. 8. Qualitative results for ablation studies of different components in RSR, corresponding to Table V. (a) Image, (b) input, (c) using SR, (d) using SR and SC, and (e) using SR, SC, and  $\mathcal{L}_{CRC}$ . Note that (b) represents the visualization of masked image and sketch.

TABLE VI  
QUANTITATIVE COMPARISON FOR ABLATION STUDIES OF THE OVERALL RSR  
PROCESS ON IMAGENET [17] TEST SET, CORRESPONDING TO FIGURE 9

| Methods              | PSNR $\uparrow$ | SSIM $\uparrow$ | FID $\downarrow$ |
|----------------------|-----------------|-----------------|------------------|
| $f_{sii}$ w/o sketch | 20.46           | 0.745           | 22.34            |
| $f_{sii}$            | 21.03           | 0.749           | 16.03            |
| $f_{sii} + f_{rsr}$  | <b>22.16</b>    | <b>0.769</b>    | <b>9.53</b>      |

Herein, the best results are highlighted in boldface.  $f_{sii}$  and  $f_{rsr}$  represent the SII and RSR processes, respectively. “ $f_{sii}$  w/o sketch” denotes the model that completes the corrupted images only based on the masked regions.

inpainting process with more coherent sketch guidance. By using both SC and  $\mathcal{L}_{CRC}$  in RSR, it is observed that our full model reveals the best performance and produces the most plausible inpainted results, which confirms the vitalness of  $\mathcal{L}_{CRC}$  for the optimization of RSR process. This observation illustrates the effectiveness of the two-stage architecture in SRN that handles the sketch refinement task in a coarse-to-fine fashion, and indicates the superiority of  $\mathcal{L}_{CRC}$  to facilitate the task with mutual information across different regions from the calibrated sketch and the supervision label.

*2) Effect of RSR on Different Methods:* We conduct experiments to investigate the effect of the overall RSR process upon our approach as well as other inpainting methods, where details are illustrated in the following paragraphs.

*a) Our approach:* To explore the effect of the overall RSR process for our approach, we try two alternative settings of completing the corrupted images, which completes the corrupted images only based on the masked regions, or directly uses the SII process to handle the input sketch. Fig. 9 presents the qualitative comparison and Table VI reports corresponding quantitative results on ImageNet test set, where “ $f_{sii}$  w/o sketch” and “ $f_{sii}$ ” in



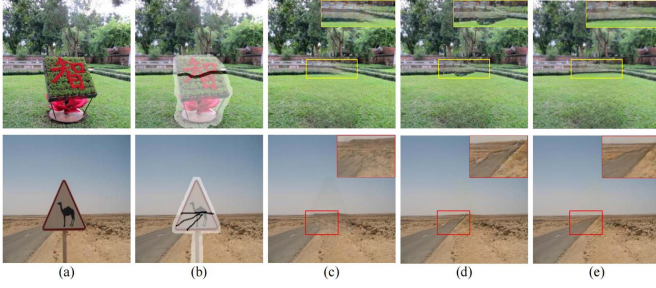


Fig. 9. Qualitative results for ablation studies of the overall RSR process, corresponding to Table VI. (a) Image, (b) input, (c) result without sketch, (d) result using unrefined sketch, and (e) result using sketch refined by RSR. Note that (b) represents the visualization of masked image with sketch.

TABLE VII  
QUANTITATIVE RESULTS FOR ABLATION STUDIES OF THE OVERALL RSR PROCESS UPON EDGE- AND SKETCH-BASED METHODS (I.E., DEEPFILL-V2 [5], SKETCHEDIT [1], AND ZITS [12]) ON CELEBA-HQ [15] TEST SET

| Methods            | PSNR <sup>↑</sup> | SSIM <sup>↑</sup> | FID <sup>↓</sup> |
|--------------------|-------------------|-------------------|------------------|
| DeepFill-v2        | 22.06             | 0.788             | 26.78            |
| DeepFill-v2 w/ RSR | <b>23.61</b>      | <b>0.817</b>      | <b>24.72</b>     |
| SketchEdit         | 20.21             | 0.759             | 41.28            |
| SketchEdit w/ RSR  | <b>22.62</b>      | <b>0.813</b>      | <b>20.15</b>     |
| ZITS               | 22.79             | 0.812             | 18.79            |
| ZITS w/ RSR        | <b>23.60</b>      | <b>0.830</b>      | <b>17.63</b>     |

Experiments using the same method represent a comparison group. Herein, the best results in each group are highlighted in boldface.

Table VI denote the aforementioned settings, respectively. There are several observations. First, “ $f_{sii}$  w/o sketch” struggles to restore the complex backgrounds without sketch guidance (e.g., (c) in Fig. 9) and leads to poor quantitative performance, which confirms the effectiveness of introducing sketch guidance into the inpainting process, especially while addressing some challenging scenarios with complex structures to recover. However, when directly utilizing the sketch guidance, the model suffers from the inaccurate guidance brought by free forms of hand-drawn sketches (e.g., (d) in Fig. 9) without the enhancement of RSR, thereby producing results with unnatural artifacts. Second, our full approach (i.e., “ $f_{sii} + f_{rsr}$ ”) present the best performance both quantitatively and qualitatively, where the model is capable of restoring the corrupted regions with coherent structures. This observation verifies the effectiveness of sketches for image inpainting, and demonstrates the superiority of RSR as a task-specific technique that better facilitates the inpainting process with sketch guidance.

*b) Other Inpainting Methods:* We analyze the impact of the overall RSR process on other image inpainting methods by facilitating their inpainting processes with the refined sketch from RSR. Table VII presents the quantitative results on CelebA-HQ test set, where we perform the experiments on a series of state-of-the-art image inpainting methods, including DeepFill-v2 [5], SketchEdit [1], and ZITS [12]. DeepFill-v2, which originally utilizes detected edge maps by HED [13] for sketch simulation, reveals significant improvements with the enhancement of RSR, suggesting that RSR helps DeepFill-v2 to address sketch-like inputs. Although SketchEdit proposes to augment

TABLE VIII  
QUANTITATIVE RESULTS FOR ABLATION STUDIES OF LSA UNDER DIFFERENT SETTINGS OF SKETCH-IMAGE ALIGNMENT, EVALUATED ON CELEBA-HQ [15] TEST SET

| Methods           |                   |                   |                   |         | CelebA-HQ         |                   |                  |
|-------------------|-------------------|-------------------|-------------------|---------|-------------------|-------------------|------------------|
| $\mathcal{F}_1^s$ | $\mathcal{F}_2^s$ | $\mathcal{F}_3^s$ | $\mathcal{F}_4^s$ | Concat. | PSNR <sup>↑</sup> | SSIM <sup>↑</sup> | FID <sup>↓</sup> |
|                   |                   |                   |                   | ✓       | 19.58             | 0.681             | 25.97            |
|                   |                   |                   | ✓                 |         | 20.76             | 0.709             | 17.11            |
|                   |                   | ✓                 | ✓                 |         | 21.23             | 0.731             | 13.83            |
|                   | ✓                 | ✓                 | ✓                 |         | 21.83             | 0.757             | 11.48            |
| ✓                 | ✓                 | ✓                 | ✓                 | ✓       | <b>22.16</b>      | <b>0.769</b>      | <b>9.53</b>      |

Here, the best results are high-lighted in boldface. *Concat.* represents concatenating sketches as an additional channel of the SCI instead of using LSA.  $\mathcal{F}_i^s$  ( $i \in \{1 \dots N_s\}$  and  $N_s = 4$ ) represents the used features in  $i$ -th layer of LSA with corresponding feature in SCI.

the corrupted images by randomly warping, such solution is still insufficient to simulate real sketches and obtains poorer performance than the one using RSR, where the original SketchEdit suffers from the gap between simulation in training and user interactions in inference. ZITS puts emphasis on facilitating image inpainting with Canny edge maps and Hough lines, which is further strengthened by RSR in the hand-drawn sketch setting, resulting in more convincing performance in all metrics. The aforementioned analyses prove the effectiveness of RSR on both our approach and other inpainting methods, where RSR is able to serve as a plug-and-play plugin for other methods and promote further related studies for the sketch-based interactive image inpainting task.

*3) Effect of Components in SII:* In this section, we analyze the effect of different components in SII following the pipeline sequence, which includes LSA and SFA, respectively. Details are illustrated in the following text sequentially.

*a) LSA:* LSA introduces the sketch guidance into the inpainting process by aligning features of both modalities. To explore the effect of LSA, we conduct experiments under different settings of sketch-image alignment by concatenating the sketch as input or varying the number of aligned features  $N_s$ . Fig. 10 and Table VIII present corresponding qualitative and quantitative results of the aforementioned settings, respectively. By concatenating sketch as an additional channel of SCI, the model struggles to align the sketch guidance with the inpainting process, thereby completing corrupted images with blur edges (e.g., (c) in Fig. 10) and obtaining poor quantitative performance (e.g., the first row in Table VIII). By using LSA, it is observed that the sketch guidance is gradually aligned with the inpainted results as  $N_s$  increases (e.g., (d) to (g) in Fig. 10), which results in gradually improved quantitative performance. This observation might be explained from the following perspectives. First, the misalignment between sketch guidance and the inpainted result proves the feature diminishing problem caused by concatenation, where the sketch guidance gradually vanishes as the network goes deeper and fails to activate the guidance for deep features in the inpainting networks. Second, when using LSA with relatively small  $N_s$  (i.e.,  $N_s = 1, 2, 3$ ), one can see that the inpainted results still reveal artifacts in the guided regions, where only aligning features in shallow layers is insufficient to achieve an effective sketch-image mapping. Third, when  $N_s$  is

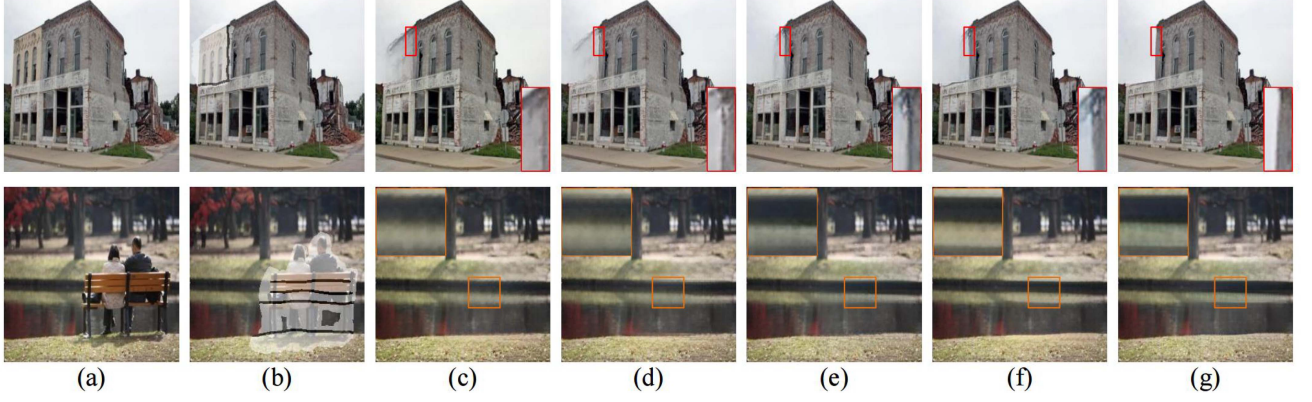


Fig. 10. Qualitative results for ablation studies of LSA under different settings of sketch-image alignment, corresponding to Table VIII. (a) Image, (b) input, (c) *Concat.*, (d) using  $\mathcal{F}_1^s$ , (e) using  $\mathcal{F}_1^s$  and  $\mathcal{F}_2^s$ , (f) using  $\mathcal{F}_1^s$ ,  $\mathcal{F}_2^s$ , and  $\mathcal{F}_3^s$ , and (g) using  $\mathcal{F}_1^s$ ,  $\mathcal{F}_2^s$ ,  $\mathcal{F}_3^s$ , and  $\mathcal{F}_4^s$  (i.e., all extracted features in LSA). *Concat.* represents concatenating sketches as an additional channel of the SCI instead of using LSA.

TABLE IX  
QUANTITATIVE RESULTS FOR THE ABLATION STUDY UNDER DIFFERENT SETTINGS OF THE GENERATOR ON IMAGENET [17]

| Methods | PSNR $\uparrow$ | SSIM $\uparrow$ | FID $\downarrow$ |
|---------|-----------------|-----------------|------------------|
| LaMa    | 21.49           | 0.76            | 15.83            |
| Ours    | 22.16           | 0.77            | 9.53             |

Herein, “LaMa” refers to the model using the original generator in LaMa [29] and “Ours” represents our approach.

sufficiently large, the model generates plausible results with explicit edges and consistent textures, where the sketch guidance is finally aligned with the inpainting process and further facilitates the restoration of missing textures. The aforementioned analyses confirm the effectiveness of LSA that establishes a robust sketch-image alignment to enhance the inpainting process with user sketches.

*b) Generator:* To investigate the effect of the generator design on our approach, we compare our approach with the one using the original generator in LaMa [29]. Table IX reports the quantitative results on ImageNet. It is observed that our approach obtains significant improvements over “LaMa”, indicating the effectiveness of the proposed generator. The reason can be explained that the vanilla LaMa generator is not designed to integrate sketch guidance, thereby resulting in inferior performance due to the misalignment between guidance from LSA and the inpainting process by SCI.

*c) SFA:* To explore the effect of SFA, we try different settings of feature aggregation, including aggregating by adding, using SPADE [55], and using SFA. Fig. 11 shows the qualitative results of the aforementioned settings and Table X reports corresponding quantitative results on CelebA-HQ test set. There are several analyses for the results. First, “ $f_{sci}$  (add)” performs the worst performance of all settings, where adding sketch and image features struggles to inject the aligned sketch features into the inpainting process. The possible reason behind might be that the extracted sketch features are different from the image features in magnitude according to their modality properties so that directly adding them would be less reasonable. Second, “ $f_{sci}$  (SPADE)” shows similar performance as “ $f_{sci}$

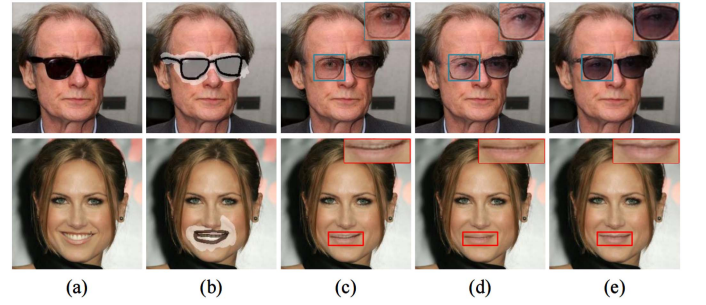


Fig. 11. Qualitative results for ablation studies of SFA under different settings of feature aggregation. (a) Image, (b) input, (c) aggregating by *adding*, (d) SPADE [55], and (e) SFA.

TABLE X  
QUANTITATIVE RESULTS FOR ABLATION STUDIES UNDER DIFFERENT SETTINGS OF SKETCH FEATURE AGGREGATION, EVALUATED ON CELEBA-HQ [15] TEST SET

| Methods           | PSNR $\uparrow$ | SSIM $\uparrow$ | FID $\downarrow$ |
|-------------------|-----------------|-----------------|------------------|
| $f_{sci}$ (add)   | 20.46           | 0.745           | 22.34            |
| $f_{sci}$ (SPADE) | 20.59           | 0.761           | 20.97            |
| $f_{sci}$ (SFA)   | <b>22.16</b>    | <b>0.769</b>    | <b>9.53</b>      |

“ $f_{sci}$  (add)” indicates the model that injects sketch guidance into SCI by directly adding the sketch feature to corresponding image feature, “ $f_{sci}$  (SPADE)” represents the model using SPADE [55] instead of SFA blocks, and “ $f_{sci}$  (SFA)” denotes our approach. Herein, the best results are highlighted in boldface.

(add)”, which also fails to merge the features across modalities. It can be explained that SPADE is designed to merge segmentation masks into the generation process and fails to handle sketch-like condition, where sketch is sparse and binary-valued in terms of its modal characteristics. Third, our approach (i.e., using SFA) leads to significant improvements and produces results with more photo-realistic edges and textures (e.g., (e) in Fig. 11), which confirm the superiority of SFA blocks compared to “ $f_{sci}$  (add)” and “ $f_{sci}$  (SPADE)”. This observation can be explained that SFA adaptively activates SCI with sketch features, which serves as a vital component of our approach to effectively facilitate the inpainting process with aligned sketches.

TABLE XI  
EFFICIENCY COMPARISON BETWEEN OUR APPROACH AND OTHER IMAGE INPAINTING METHODS WITH RESPECT TO THEIR MODEL ARCHITECTURES (“ARCHITECTURE”), MODEL PARAMETERS (“PARAMS”), AND AVERAGE INFERENCE TIME (“ $\Delta T$ ”)

| Methods         | Architecture | Params | $\Delta T$ (s) |
|-----------------|--------------|--------|----------------|
| DeepPS [9]      | GAN          | 39M    | 0.0467         |
| DeepFill-v2 [5] | GAN          | 4M     | 0.0109         |
| LaMa [29]       | GAN          | 27M    | 0.0167         |
| SketchEdit [1]  | GAN          | 8M     | 0.0122         |
| ZITS [12]       | GAN          | 67M    | 0.0389         |
| MAT [34]        | Trans.       | 62M    | 0.0760         |
| ICT [32]        | Trans.       | 97M    | 144.73         |
| RePaint [35]    | Diff.        | 608M   | 292.04         |
| Ours            | GAN          | 38M    | 0.0193         |

We conduct all experiments on the CelebA-HQ test set under the same software and hardware environment. Herein, “Trans.” and “Diff.” represent Transformer- and diffusion-based methods, respectively.

### F. Efficiency Comparison

Our approach introduces additional components on top of the standard image inpainting pipeline. We then conduct an efficiency comparison between our approach and others, so as to evaluate the extra computational complexity. In doing so, we measure the computational complexity of our approach compared to current state-of-the-art image inpainting methods from different perspectives, including model parameters and average inference time. In addition to GAN-based methods, we also make comparisons with Transformer-based [32], [34] and diffusion-based methods to comprehensively consider the impact of different model architectures. Table XI reports the results of the aforementioned experiments with several observations as follows. From the perspective of model parameters, our approach obtains a moderate scale of parameters (i.e., 38 M) among GAN-based methods, and has significantly fewer parameters than Transformer-based (e.g., ICT [32] with 97 M) or diffusion-based methods (e.g., RePaint [35] with 608 M), indicating that our approach is efficient to deploy. As for the average inference time, our approach brings little extra inference time (around 0.003 seconds) compared to LaMa [29]. Moreover, our approach is significantly faster than Transformer-based approaches, especially when compared to ICT [32] that also conducts two stages to finish the image inpainting process. This can be explained that the autoregressive paradigm of the Transformer architecture requires to reconstruct pixels sequentially and causes additional inference time compared to the generative paradigm of GAN, especially when the mask region is large with complex structures to recover. Notably, RePaint [35] requires the most inference time on average (nearly 5 minutes), since the de-noising process of diffusion models needs adequate sampling time steps (1,000 for RePaint [35] following its default settings) to ensure the quality of the generated image. Generally speaking, our approach demonstrates superior efficiency compared to all other methods and reveals the potential of being deployed on real-time applications with portable devices such as smartphones.

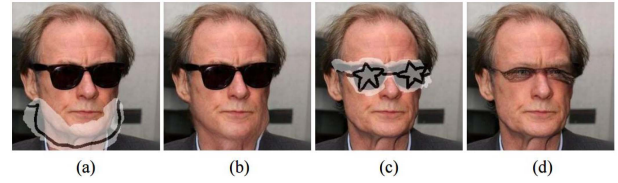


Fig. 12. Failure cases of our approach. (a) Input, (b) result, (c) input, and (d) result. Note that (a) and (c) represent the visualization of masked image and sketch.

### G. Failure Cases and Limitations

Although the aforementioned sections comprehensively illustrate the effectiveness of our proposed approach, we observe that the proposed approach is still limited while handling some real-world scenarios. Therefore in this section, we discuss the limitations and working boundaries of our approach with some particular failure cases presented. Fig. 12 shows the aforementioned failure cases, which are selected from our test protocol. There are some observations from different aspects. First, as is shown in (b) of Fig. 12, our approach fails to calibrate misaligned sketches with large corrupted regions, where the inpainted result is blurred and contains artifacts. The possible explanation for this is that the valid information for the sketch refinement and inpainting processes gradually becomes limited as the corrupted region enlarges, and the model reaches its working boundaries and struggles to refine the input sketch with large regions of misalignment. Second, it is observed that our approach tends to over-refine sketches with meticulous structures, e.g., the star-shaped sunglasses in (b) of Fig. 12. Our approach tends to calibrate the sketch from a low-level vision perspective, which cannot recognize the semantic meaning of the refined structures from a high-level perspective. This observation also indicates the diversities and challenges of the sketch refinement task during user interactions, where user intentions may sometimes be misaligned with the model capability. Third, our approach only accepts monochrome sketches (i.e., binary-valued sketches) as the inputs currently, which fails to recognize color information provided in the input sketch. This indicates that our approach is still limited to some image editing applications that work with colored strokes.

## V. CONCLUSION

In this article, we re-investigate the challenges of sketch-based interactive image inpainting. We attempt to bridge the gap between sketch-like inputs and current image inpainting models by proposing a two-stage system, namely SketchRefiner. SketchRefiner is able to restore complex structures in the corrupted regions by calibrating the input sketch to facilitate the image inpainting process with more elaborated guidance. Moreover, we collect a real-world test protocol with manually annotated masks and sketches to promote further studies upon sketch-based real-world applications. Experiments on three public benchmark datasets and the real-world test protocol demonstrate the superiority of our approach both qualitatively and quantitatively. Ablation studies analyze the effectiveness of



different components in SketchRefiner and indicate the possible extension of our approach upon further related studies. SketchRefiner provides a paradigm for interactive image inpainting, which reveals the potentials of extending to other interactive media, e.g., colored stroke, colored sketch, etc.

## REFERENCES

- [1] Y. Zeng, Z. Lin, and V. M. Patel, "SketchEdit: Mask-free local image manipulation with partial sketches," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5951–5961.
- [2] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. SIGGRAPH*, 2000, pp. 417–424.
- [3] J. Sun, L. Yuan, J. Jia, and H.-Y. Shum, "Image completion with structure propagation," in *Proc. SIGGRAPH*, 2005, pp. 861–868.
- [4] D. Sun, L. Yuan, Y. Zhang, J. Zhang, and G. Pan, "Structure-aware image completion with texture propagation," in *Proc. IEEE 6th Int. Conf. Image Graph.*, 2011, pp. 199–204.
- [5] J. Yu et al., "Free-form image inpainting with gated convolution," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 4471–4480.
- [6] K. Nazeri, E. Ng, T. Joseph, F. Qureshi, and M. Ebrahimi, "EdgeConnect: Structure guided image inpainting using edge prediction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops*, 2019, pp. 3265–3274.
- [7] Y. Jo and J. Park, "SC-FEGAN: Face editing generative adversarial network with user's sketch and color," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1745–1753.
- [8] W. Xiong et al., "Foreground-aware image inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5840–5848.
- [9] S. Yang, Z. Wang, J. Liu, and Z. Guo, "Deep plastic surgery: Robust and controllable image editing with human-drawn sketches," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 601–617.
- [10] X. Guo, H. Yang, and D. Huang, "Image inpainting via conditional texture and structure dual generation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 14134–14143.
- [11] H. Liu et al., "DeFLOcNet: Deep image editing via flexible low-level controls," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 10765–10774.
- [12] Q. Dong, C. Cao, and Y. Fu, "Incremental transformer structure enhanced image inpainting with masking positional encoding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11358–11368.
- [13] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 1395–1403.
- [14] J. He, S. Zhang, M. Yang, Y. Shan, and T. Huang, "BDCN: Bi-directional cascade network for perceptual edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 100–113, Jan. 2022.
- [15] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–26.
- [16] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1452–1464, Jun. 2018.
- [17] J. Deng et al., "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.
- [18] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, Sep. 2004.
- [19] A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, vol. 2, 1999, pp. 1033–1038.
- [20] I. Drori, D. Cohen-Or, and H. Yeshurun, "Fragment-based image completion," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 303–312, Jul. 2003.
- [21] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patch-Match: A randomized correspondence algorithm for structural image editing," in *ACM Trans. Graph.*, vol. 28, no. 3, pp. 1–11, 2009.
- [22] J. Hays and A. A. Efros, "Scene completion using millions of photographs," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 4–es, Jul. 2007.
- [23] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2536–2544.
- [24] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–14, 2017.
- [25] J. Yu et al., "Generative image inpainting with contextual attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5505–5514.
- [26] C. Zheng, T.-J. Cham, and J. Cai, "Pluralistic image completion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1438–1447.
- [27] H. Liu, B. Jiang, Y. Song, W. Huang, and C. Yang, "Rethinking image inpainting via a mutual encoder-decoder with feature equalizations," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 725–741.
- [28] J. Peng, D. Liu, S. Xu, and H. Li, "Generating diverse structure for image inpainting with hierarchical VQ-VAE," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 10775–10784.
- [29] R. Suvorov et al., "Resolution-robust large mask inpainting with fourier convolutions," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2022, pp. 2149–2159.
- [30] R. Zhang, W. Quan, Y. Zhang, J. Wang, and D.-M. Yan, "W-Net: Structure and texture interaction for image inpainting," *IEEE Trans. Multimedia*, vol. 25, pp. 7299–7310, 2023.
- [31] Y. Yu et al., "Diverse image inpainting with bidirectional and autoregressive transformers," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 69–78.
- [32] Z. Wan, J. Zhang, D. Chen, and J. Liao, "High-fidelity pluralistic image completion with transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 4672–4681.
- [33] C. Zheng, T.-J. Cham, J. Cai, and D. Phung, "Bridging global context interactions for high-fidelity image completion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11512–11522.
- [34] W. Li et al., "MAT: Mask-aware transformer for large hole image inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10758–10768.
- [35] A. Lugmayr et al., "RePaint: Inpainting using denoising diffusion probabilistic models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11461–11471.
- [36] K. Zhang, J. Fu, and D. Liu, "Flow-guided transformer for video inpainting," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 74–90.
- [37] K. Zhang, J. Fu, and D. Liu, "Inertia-guided flow completion and style fusion for video inpainting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5972–5981.
- [38] L. Zhang, Q. Chen, B. Hu, and S. Jiang, "Text-guided neural image inpainting," in *Proc. 28th ACM Int. Conf. Multimedia*, 2020, pp. 1302–1310.
- [39] O. Avrahami, D. Lischinski, and O. Fried, "Blended diffusion for text-driven editing of natural images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 18208–18218.
- [40] H. Wu, J. Zhou, and Y. Li, "Deep generative model for image inpainting with local binary pattern learning and spatial attention," *IEEE Trans. Multimedia*, vol. 24, pp. 4016–4027, 2022.
- [41] J. Liu, S. Yang, Y. Fang, and Z. Guo, "Structure-guided image inpainting using homography transformation," *IEEE Trans. Multimedia*, vol. 20, no. 12, pp. 3252–3265, Dec. 2018.
- [42] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 44:1–44:10, 2012.
- [43] Y. Vinker et al., "CLIPasso: Semantically-aware object sketching," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 1–11, Jul. 2022.
- [44] X. Ren, C. C. Fowlkes, and J. Malik, "Scale-invariant contour completion using conditional random fields," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, vol. 2, 2005, pp. 1214–1221.
- [45] Y. Ming, H. Li, and X. He, "Connected contours: A new contour completion model that respects the closure effect," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 829–836.
- [46] S. Xu, D. Liu, and Z. Xiong, "E2I: Generative inpainting from edge to image," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 4, pp. 1308–1322, Apr. 2021.
- [47] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5769–5779.
- [48] X. Qin et al., "U2-Net: Going deeper with nested U-structure for salient object detection," *Pattern Recognit.*, vol. 106, 2020, Art. no. 107404.
- [49] J. Shi, Q. Yan, L. Xu, and J. Jia, "Hierarchical image saliency detection on extended CSSD," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 4, pp. 717–729, Apr. 2016.
- [50] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6629–6640.
- [51] T. Salimans et al., "Improved techniques for training GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 2234–2242.
- [52] L. Gatys, A. Ecker, and M. Bethge, "A neural algorithm of artistic style," *J. Vis.*, vol. 16, no. 12, pp. 326–326, 2016.

- [53] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–14.
- [54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
- [55] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2337–2346.



**Chang Liu** received the B.S. degree from Xiangtan University, Xiangtan, China, in 2021. He is currently working toward the Ph.D degree with the University of Science and Technology of China, Hefei, China. His research interests include image inpainting, conditional image synthesis, video editing, and computer vision.



**Shunxin Xu** received the Ph.D. degree from the University of Science and Technology of China, Hefei, China, in 2022. His research interests include image processing and computer vision.



**Jialun Peng** received the B.S. degree in applied physics in 2018 from the University of Science and Technology of China, Hefei, China, where he is currently working toward the Ph.D degree in electrical engineering. His research interests include image restoration, generation, and editing.



**Kaidong Zhang** received the B.S. degree in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 2019. He is currently working toward the Ph.D. degree in electrical engineering with the University of Science and Technology of China, Hefei, China. His research interests include deep learning, video restoration, video editing, computer graphics, and medical image processing.



**Dong Liu** (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electrical engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2004 and 2009, respectively. From 2009 to 2012, he was a Member of Research Staff with Nokia Research Center, Beijing, China. In 2012, he joined USTC as a faculty member and became a Professor in 2020. He has authored or coauthored more than 200 papers in international journals and conferences and has more than 30 granted patents. His research interests include image and video processing, coding, analysis, and data mining. He has several technique proposals adopted by standardization groups. Dr. Liu is a Senior Member of CCF and CSIG, and an elected Member of MSA-TC of IEEE CAS Society. He is/was the Chair of IEEE 1857.11 Standard Working Subgroup (also known as Future Video Coding Study Group), an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING, an Guest Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and an Organizing Committee Member for VCIP 2022, ChinaMM 2022, and ICME 2021. He was the recipient of the 2009 IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY Best Paper Award, VCIP 2016 Best 10% Paper Award, and ISCAS 2022 Grand Challenge Top Creativity Paper Award. He and his students were winners of several technical challenges held in ISCAS 2023, ICCV 2019, ACM MM 2019, ACM MM 2018, ECCV 2018, CVPR 2018, and ICME 2016.