Generative Models for Semantic Facial Image Editing: Multimodal Approaches

by

© Wanglong Lu

A thesis submitted to the

School of Graduate Studies

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Supervisor: Dr. Xianta Jiang, Dr. Hanli Zhao, and Dr. Yuanzhu Chen

Department of Computer Science

Memorial University of Newfoundland

May 2025

St. John's Newfoundland

Abstract

With the rapid development of digital imaging and machine learning, generative models for facial image manipulation have emerged as powerful tools, significantly impacting various domains, from entertainment to law enforcement. Despite significant advancements in generating natural-looking images, facial editing poses unique challenges, such as generating high-quality and detailed facial features, preserving identity, expression, and the integrity of facial structures.

This thesis investigates the application of generative models to facial image manipulation, targeting four key tasks: unconditional global facial editing (face restoration), unconditional local facial editing (face inpainting), conditional facial editing (exemplar-guided facial inpainting), and multimodal face editing.

For the first task, traditional face restoration techniques typically miss finer facial details. We explored the use of latent representations as style prompts by using GANs and diffusion models to guide the restoration, improving image quality and detail. In the second task, existing image inpainting methods often depend on extensive training data, limiting their effectiveness in few-shot scenarios. We developed a GAN-based method that achieves high-quality results with small-scale data. For the third task, current methods usually require substantial professional skills to edit facial attributes like identity, expression, and gender. We propose an exemplar-guided GAN framework that ensures a seamless blend between edited and unedited areas of the face. For the fourth task, current multimodal editing techniques can alter unedited background areas and rely heavily on manually annotated paired data. We introduce a novel multimodal editing method using GANs that allows for incremental editing of facial images and reduces reliance on manual annotations.

We introduce novel frameworks that significantly enhance the realism and applicability of facial image manipulation by solving problems in fidelity in restoration, data efficiency, exemplar-guided inpainting, and multimodal editing. Our contributions are mainly four manifold:

- A novel framework for blind face restoration is presented, leveraging latent representations as style prompts to guide the restoration process, thereby enhancing the fidelity and detail of restored facial images from degraded sources.

- We introduce a data-efficient generative model for facial image inpainting that achieves high-quality results on limited datasets, addressing the challenge of data scarcity and overfitting in image inpainting.

- We proposed an interactive, example-guided facial inpainting framework that enables users to manipulate facial features with high realism, facilitating user-driven customization in facial image editing.

- A multimodal facial image editing framework is proposed, integrating various types of inputs to achieve comprehensive and personalized facial edits, catering to the diverse needs of digital content editing.

Acknowledgements

On a particularly difficult day, I faced the rejection of two papers simultaneously. As I sat in a restaurant, trying to figure out how to improve my third manuscript, a heavy weight settled in my chest. It was a tough moment, filled with doubt and uncertainty about what the next round of reviews might bring. In that challenging time, the words of my supervisor, Professor Xianta Jiang, echoed in my mind: "Do not let this drive you crazy; just enjoy your life and your research." His advice, to value both life and the pursuit of knowledge, has stayed with me ever since and helped me persevere.

Time flies, and I have done my best to make the most of every opportunity, even through the hardest times – when algorithms didn't work or weeks of exploration led only to dead ends. Looking back, I feel fortunate to have made it through those struggles, as they have shaped my growth.

I am deeply grateful to my supervisors for their unwavering support and guidance. Professor Xianta Jiang gave me the confidence to tackle challenges head-on; Professor Hanli Zhao emphasized the importance of precision and rigor in research; and Professor Yuanzhu Chen broadened my perspective and sharpened my critical thinking. Together, they transformed me from a hesitant student into a confident scholar.

I also want to thank Professors Minglun Gong, Xiaogang Jin, Yongliang Yang, and Zili Yi for their mentorship. Their insights enriched my knowledge and opened doors to new areas of research, fostering my growth as a researcher.

To my senior colleagues, Tao Wang, Jingjing Zheng, Qiao Kang, and Yajun Yu, thank you for your invaluable mentorship. Your advice and encouragement greatly accelerated my progress. My heartfelt gratitude also goes to my friends and colleagues (Ziying Lyu, Kaijie Shi, Jikai Wang, Lingming Su, Meng Wang, Vitaliy Zhao, etc.) at Wenzhou University and Memorial University of Newfoundland. You have been like family to me, providing not only collaboration but also personal support and encouragement throughout this journey. I must also acknowledge the co-authors of my published papers. Your collaboration and hard work have been instrumental in overcoming numerous challenges and achieving success. Thank you, from the depths of my heart, to everyone who has been part of this journey.

Finally, I owe everything to my parents. Your unwavering support and love have been my anchor. Without you, I could never have reached this point or fulfilled my dreams. This achievement belongs as much to you as it does to me.

I also would like to thank the anonymous reviewers for their constructive comments. This work was supported by the Zhejiang Provincial Natural Science Foundation of China (Grant No. LZ21F020001), Wenzhou Major Tackling Project of Science and Technology Innovation (Grant No. ZF2024001), Natural Sciences and Engineering Research Council of Canada (NSERC, Grant No. DGECR-2020-00296), Government of Canada's New Frontiers in Research Fund (NFRF, Grant No. NFRFE-2022-00407), and Digital Research Alliance of Canada.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Motivation. With the rapid development of the internet, cloud computing, and mobile communication technologies, along with the improved processing capabilities of mobile devices, various applications are reshaping our daily lives and work routines. Among these, applications related to facial images have garnered widespread attention due to their extensive use, resulting in the generation and storage of a vast number of facial images every day. These applications across multiple domains, including social entertainment, e-commerce, fashion design, visual effects in films, and criminal investigation, underline the growing need for sophisticated facial image manipulation technologies.

Facial image manipulation refers to the process of digitally altering facial images through various computational techniques to enhance, transform, or modify facial features in a photograph or image. This process encompasses a wide range of tasks, typically including face restoration (e.g., denoising, deblurring, super-resolution, and enhancement), facial image inpainting [27], facial style transfer [28], and facial attribute manipulation [29] (e.g., skin texture [16], face swapping [30], expression manipulation, age manipulation [31], hair editing [32], facial contouring, and reshaping [33]).

To enhance the whole facial features or restore the noisy part of the whole facial image, unconditional global facial manipulation (such as facial restoration) can be used to enhance the global and local details of facial features. To achieve localized enhancements, unconditional local facial manipulation (like image inpainting) allows for interactive removal and seamless replacement of undesirable visual features. However, these methods have limitations in terms of controllable editing ability and cannot provide flexible editing of facial features. To bridge this gap, conditional local facial editing (like exemplar-guided facial inpainting) empowers users to transfer specific facial attributes from a given exemplar image, enabling semantic manipulation like expression and identity transfer. To support extensive multimodal editing, multimodal facial editing stands out by utilizing semantics to edit facial image's coarse layout, sketches to refine facial structure and texture, and texts or attribute labels to adjust facial attributes, to name just a few.

With the fast-paced progress in computer vision and deep learning, deep-learning-based facial image editing has made considerable progress in recent years. These methods understand images at a semantic level, delivering remarkable results beyond the pixel-level interpretations of earlier techniques. Yet, numerous difficulties and challenges persist, affecting their wider application across various scenarios. This thesis aims to address these issues in facial editing. We explored blind face restoration for unconditional global facial editing, facial image inpainting for unconditional local facial editing, exemplar-guided facial inpainting for conditional facial editing, and tackled challenges in multimodal facial image editing.

Challenges. Starting from the unconditional global facial editing – blind face restoration, we found that prior knowledge-based methods lead the way in achieving high-quality face restoration, but still face challenges in achieving high-quality restoration performance. These methods use geometric priors to guide the restoration process, by drawing on shape information such as parsing maps [34], component heatmaps [35], and identity [36]. Nonetheless, their results often lack comprehensive detail since they focus on limited facial features [37]. To address this issue, some studies [38, 39, 37, 40] have employed facial priors inherently present in pre-trained generative models [41, 42] or vector quantization dictionary [43] for guiding restorations that retain detailed facial attributes. However, when facial images are degraded, critical features (e.g., details of expression and identity) may be lost, leading to inaccurate facial prior estimation for reconstruction. Achieving precise facial prior estimation that corresponds to high-quality images is crucial for enhanced restoration, and is still an open challenge.

For the unconditional local facial editing – facial image inpainting, we found that existing state-of-the-art methods [44, 45] require a large amount of data to train their convolutional neural network (CNN) or transformer models [46]. When these models are trained on small image datasets, there is a high possibility of overfitting and model collapse [47]. In practice, it is much easier for users if only a small number of training images is required. Furthermore, in specific image domains, such as medical imaging, art, and historical relics, collecting large datasets is often too costly or impractical, significantly limiting the application of image inpainting in real-world scenarios. Thus, designing a data-efficient method, which is capable of training on limited data while producing high-quality inpainting outcomes remains a challenge.

For conditional facial manipulation – exemplar-guided image inpainting, to achieve realistic facial inpainting guided by exemplar images, there are two main challenges: how to learn the style of facial attributes from the exemplar and how to guarantee natural transition on the mask boundary. Some works [48, 49] attempt to generate diverse image inpainting results giving users the option to select their preferred one. However, they cannot utilize user guidance to complete missing regions. Many recent methods try to employ additional landmarks [50], strokes [16], or sketches [26, 51] to guide the inpainting of facial structures and attributes. However, these methods tend to overfit the resulting images with this limited guidance information. As a result, these methods still require considerable professional skills in order to generate satisfactory target facial attributes, such as identity, expression, and gender.

For multimodal facial manipulation, there are three main challenges. The first challenge is the difficulty in preserving visual content in unedited background areas.

However, existing multimodal facial editing techniques [52, 14, 12, 13] can only edit the facial image as a whole and are prone to introduce unwanted changes to unedited background regions. When users are not satisfied with some local effects, these techniques fail to edit the local regions in an incremental manner. While state-of-the-art (SOTA) methods may perform high-quality edits, they are likely to include unwanted changes of other facial features in incremental editing scenarios, where an already edited image is subject to further modifications using different modalities. The second challenge is the need for manual annotations of paired data. For example, some existing methods [52, 14] train their models with labeled paired data across different modalities, but manual annotations of training datasets are label-intensive. The third challenge is the need for training a single modal model for each input modality. The methods based on diffusion models [12, 13] train all uni-modal models first and perform editing by integrating these pre-trained uni-modal models. When the number of modalities grows, more uni-modal models should be trained separately. How to design a multimodal manipulation method, that meets these requirements for high-quality and controllable editing, is still an open question.

Contributions. Due to advances in deep learning, generative models (such as Generative Adversarial Networks (GANs) [41], Variational Auto-Encoder (VAE) [53], and Diffusion Probabilistic Models (DMs) [54]) have become key solutions for tackling various challenges in image segmentation, image generation, image editing, etc. Leveraging the advanced learning abilities of deep network architectures, such as Convolutional Neural Networks (CNNs) and Transformers [46], these models can excel at creating images with remarkable realism and precision, offering a deeper understanding of image semantics. However, the straightforward application of these techniques results in limited effectiveness due to the absence of designs tailored to specific questions. Finding ways to harness generative models and deep neural networks for high-quality facial image manipulation remains challenging.

This thesis explores facial image manipulation, specifically focusing on overcoming the aforementioned challenges across four pivotal tasks: face restoration, face inpainting, exemplar-guided facial editing, and multimodal face editing.

For the face restoration, we explore the use of latent representations in pre-trained generative models as visual style prompts for directing the face restoration process. Specifically, there are mainly two questions: (1) how to accurately embed visual prompts from a degraded face image, and (2) how to integrate the visual prompts and facial priors within the network to boost performance. To answer these questions, we first explore the role of Diffusion Models (DMs) in improving the estimation of clean latent representations from degraded images, aiding facial feature extraction for restoration. Secondly, we evaluate how rescaling and adjusting multiscale convolutional kernels according to visual prompts can enhance feature extraction, and better utilize visual information across different receptive fields.

For the face inpainting task, we propose a novel data-efficient generative residual image inpainting framework (GRIG), which enables high-quality image inpainting on small-scale datasets. To effectively optimize inpainting results, we use iterative reasoning to more accurately and generalizably solve algorithmic reasoning tasks [55], based on residual learning [56] to incrementally refine previous estimates. By continually updating residual offsets and utilizing inpainted information from previous iterations, our model dynamically refines the input image. This approach reduces direct memorization of input-to-ground-truth mappings, effectively diminishing overfitting and enhancing visual quality. We also investigate whether combining iterative reasoning and residual learning with CNNs and transformers [46], as well as image-level and patch-level discriminators, can lead to a more robust and data-efficient method to tackle the data-efficient image inpainting task.

For exemplar-guided facial editing, we propose EXE-GAN, a novel interactive facial inpainting framework, which enables high-quality generative facial inpainting guided by exemplars. Our framework consists of four main components, including a mapping network, a style encoder, a multi-style generator, and a discriminator. Our method mixes the global style of the input image, the stochastic style generated from the random latent code, and the exemplar style of the exemplar image to generate highly realistic images. We impose a perceptual similarity constraint to preserve the global visual consistency of the image. To enable the completion of exemplar-like facial attributes, we further employ facial identity and attribute constraints on the output result. To guarantee natural transition across the boundary of inpainted regions, we devise a novel spatial variant gradient backpropagation method for the network training.

For multimodal face editing, we investigated whether incorporating generative adversarial networks (GANs) [41] can improve global consistency for multimodal local facial editing. By learning the distribution of real facial images, adversarial training enforces the model to fill plausible contents for edited regions guided by multimodalities. To minimize the dependency on paired training data, we asked the question that if we can loose the ties between the paired modalities data by aligning all modalities into a unified generative latent space to diminish the requirement for paired text, attribute label, and exemplar modalities. Instead of training a uni-modal model for each modality, we examined whether fusion and warping priors along with multimodalies in both latent and feature space, could achieve seamless integration of multimodalities.

We propose four novel frameworks:

- A visual style prompt learning framework for blind face restoration that utilizes latent representations from pre-trained generative models for guiding the restoration process.

- A data-efficient generative residual image inpainting framework (GRIG) that optimizes the use of small-scale datasets for high-quality inpainting.

- An interactive facial inpainting framework (EXE-GAN) guided by exemplars, enabling high-quality generative inpainting of facial images.

- A multimodal generative fusion framework for local facial editing (FACEMUG) that integrates various types of conditioning information for personalized editing.

Extensive experimental results and applications demonstrate the remarkable performance of our methods in enhancing the realism, detail, and personalization of facial image manipulation. The above research has led to the following published or pending works [57, 58, 27, 59]:

- Wanglong Lu, Jikai Wang, Tao Wang, Kaihao Zhang, Xianta Jiang, and Hanli Zhao. Visual style prompt learning using diffusion models for blind face restoration. Pattern Recognition, 161:111312, 2025.

- Wanglong Lu, Xianta Jiang, Xiaogang Jin, Yong-Liang Yang, Minglun Gong, Tao Wang, Kaijie Shi, and Hanli Zhao. GRIG: Few-shot generative residual image inpainting, Computational Visual Media, (In Press), 2025.

- Wanglong Lu, Hanli Zhao, Xianta Jiang, Xiaogang Jin, Yong-Liang Yang, and Kaijie Shi. Do inpainting yourself: Generative facial inpainting guided by exemplars. Neurocomputing, 617:128996, 2025.

- Wanglong Lu, Jikai Wang, Xiaogang Jin, Xianta Jiang, and Hanli Zhao. FACEMUG: A multimodal generative and fusion framework for local facial editing. IEEE Transactions on Visualization and Computer Graphics, 1-15, 2024.

Organization of the Dissertation. The remainder of this thesis is structured as follows: Chapter 2 introduces core concepts and related work on generative models for image synthesis, blind face restoration, image inpainting, and facial image manipulation. Chapter 3 discusses the visual style prompt learning framework for blind face restoration. Chapter 4 presents the data-efficient generative residual image inpainting framework (GRIG). The interactive facial inpainting framework (EXE-GAN) is explored in Chapter 5, and the multimodal generative fusion framework for local facial editing (FACEMUG) is detailed in Chapter 6. Finally, conclusions and future work are discussed in Chapter 7.

The writing style of this proposal follows the manuscript style. Chapters 3 to 6 are separate manuscripts published or under review. Some content of these sections in the manuscripts maybe duplicated.

Co-Authorship Statement: I (Wanglong Lu) am the principal author for all the published or submitted works presented in this thesis. For each work, I proposed ideas and solutions, conducted experiments, and wrote the manuscripts. My co-authors provided constructive comments, assisted with setting up experiments, and contributed to manuscript revisions.

Chapter 2

Related Work

2.1 Generative models for image synthesis

Thanks to the efficient sampling of high-resolution images with good perceptual quality [60], Generative Adversarial Networks (GANs) [41] have dominated in learning image synthesis tasks for several years, such as image style transfer [61, 62], image inapinting [63], and image super-resolution [64]. The learned latent space enables an intuitive approach to control the image generation process, thereby achieving semantic manipulation. With advancements of disentangled latent space learning, such as Fader Networks [65] and StyleGANs [42, 60, 66], the manipulation of latent codes in a pre-trained GAN model has become an active research area on image editing. Recently, Diffusion Probabilistic Models [67] have showcased remarkable potential in image synthesis quality [68]. To solve the drawback of low-speed sampling, Latent

Diffusion Models (LDMs, Stable Diffusion) [69] were proposed for achieving better visual quality on image inpainting and class-conditional image synthesis, while significantly reducing computational requirements. Even though the sampling speed can be partially solved by sampling strategies [70] and hierarchical approaches [69, 71], multiple denoising steps are still needed for high-quality synthesis.

2.2 Blind face restoration

Blind face restoration, which aims to reconstruct original faces from degraded images with unknown sources of degradation, has been a long-standing problem with wide-ranging application potentials [72, 73]. Following the achievements of Deep Neural Networks (DNN) in various research fields, DNN-based approaches have become the leading methodology in blind face restoration [74]. Since limited restoration cues are available for effective restoration, the use of auxiliary priors has become increasingly common. These include geometric priors [34], reference priors [75], generative priors [39, 76, 38, 77], and multi-priors [78]. For geometric priors, an auxiliary model is employed to predict facial structure information, including landmarks [79], parsing maps [34], and component heatmaps [35], to assist in restoration. However, the accuracy of these geometric priors is heavily dependent on the condition of the facial degradation. Reference-based approaches [75] rely on exemplars for guidance, yet their effectiveness is constrained when exemplars of the same identity as the degraded face are unavailable. Generative priors [39, 76, 38, 77], leveraging high-quality face generators or vector quantization dictionaries [43, 37, 40], have shown significant potential in blind face restoration. However, these methods do not focus primarily on the explicit estimation of representations in pre-trained models, potentially limiting their ability to provide accurate feature priors for guiding the restoration process. Moreover, they also face challenges in dynamically selecting relevant features and adjusting the kernels' receptive fields. Recent studies [80] have employed diffusion models for boosting robustness against common degradations, successfully generating high-quality facial images. Yet, these approaches, mainly operating in the pixel space, suffer from low inference speeds and are time-intensive. In contrast, we utilize a denoising process in the latent space to restore clear representations of degraded images, restoring facial features with high efficiency. Our model combines the advantages of dynamically rescaling and adjusting convolutional kernels to enhance the feature extraction of informative context and detailed patterns for better restoration.

2.3 Image inpainting

Image inpainting can be grouped into traditional image inpainting methods and deep-learning-based inpainting approaches. The former mainly relies on low-level features, while the latter leverages deep neural networks to extract semantic features, resulting in better visual quality. However, there has been limited research into training these deep-learning-based models on a small number of samples.

Early image inpainting techniques rely heavily on low-level features from pixels and image patches. Methods based on diffusion [81, 82, 83] propagate undamaged information along the boundary to the hole's center. Patch-based methods [84, 85, 86] iteratively search for and copy similar appearances from image datasets or known backgrounds. Some variants include GPU-based parallel methods [87], summarizing of non-stationary patterns [88], and inpainting with nonlocal texture similarity [89]. Because of the lack of semantic understanding of the image, these methods perform well for small-scale and narrow missing regions but fail to recover meaningful contents for large holes [90].

Deep-learning-based inpainting methods have achieved great success in semantic completion. Deep neural networks have been used extensively to improve the visual quality of inpainting [91]. These works include an auto-encoder-based architecture [92] and its variant architectures [93, 94, 95, 96]. Various sophisticated modules or learning strategies have been developed to enhance the effectiveness of image inpainting, including global and local discriminators [63], contextual attention [97, 98, 99, 100] to improve semantic understanding, methods for dealing with irregular holes [101, 26, 49], and utilization of auxiliary information (such as sketches [16], foreground contours [102], and structures [103]). Recent research has addressed issues related to high-resolution [99, 104, 44, 45, 105, 106], Transformer-based inpainting [107, 108, 109, 110], pluralistic generation [45, 49, 48, 111, 112], and large hole filling [113, 114, 115, 45, 49, 111, 105, 112, 116]. The methods discussed above aim for semantically high-quality completion, but they may overfit when trained on data with a small number of samples.

Progressive-based image inpainting methods [114, 106, 115, 113, 116] are closely related to our work. These methods primarily inpaint pixels from the hole boundary to the center in a progressive manner [114, 113, 116, 106] or employ multi-stage refinement schemes [106, 115]. For example, Zeng et al. [106] improved high-resolution inpainting by iteratively predicting a confidence map and corresponding intermediate results. Such methods reuse only a portion of the predicted information and do not change pixels with high confidence for the next iterative inpainting. Recurrent Feature Reasoning (RFR) [115] runs embedded feature maps through their feature reasoning module multiple times to generate multiple features for adaptive feature merging. RFR's final inpainted results, on the other hand, are produced from their decoder with a single forward pass, indicating that the model cannot readjust its results at the pixel level for better fine details.

2.4 Facial image editing

Facial image editing (a.k.a. image manipulation) is closely related to image inpainting, which is a significant task in computer graphics and computer vision, focusing on the modification and control of various visual features, such as skin texture [16], lighting [117, 118], structure [33], expressions [27], and identity [119].

To remove the unwanted or blemished visual features, existing image inpainting techniques [99, 45, 49, 48, 111, 112] can be applied directly to remove the pixels in masked regions and fill with plausible content. However, they have limitations in terms of controllable editing ability and cannot provide multimodal conditional editing. By utilizing auxiliary information, such as sketches [120, 16], colors [16], foreground contours [102], and structures [103], low-level facial features such as texture and geometry can be manipulated. However, these methods tend to overfit on limited guidance information. Thus, professional skills may needed when users opt to edit facial features with semantic level (e.g., facial expression and identity).

To perform semantic-level face manipulation, some methods [121, 122] use a domain label to index the mapped latent codes and can control a set of attributes. However, they are limited to pre-defined attributes, thereby restricting editing freedom. Geometry-guided face manipulation methods mainly use semantic geometry, such as sketches [18, 15, 19], semantic maps [123, 21, 20, 6, 22] to guide the generation of facial structure. However, due to information loss during the projection and reconstruction process between real photographs and corresponding latent representations, these methods might inadvertently alter fine facial details (i.e., unedited regions may be changed). Recent advancements allow the transfer of facial attributes from example images at the instance level [124, 125, 126, 127, 119, 128], manipulating attributes such as expression, identity, and decorative elements. Despite their advantages, these methods share a common drawback: users cannot flexibly select facial regions for local facial editing. Image-composition-based methods enable local editing by overlaying the foreground of a source image onto the background of a target image [3, 4, 5]. Although they can generate more diverse and realistic facial images, their method may fail when the poses in the edited and reference images differ.

Manipulating latent codes in pre-trained GAN models is an active research area in image editing. GAN inversion, a fundamental step in image manipulation, uses optimization-based [129, 2, 130] and encoder-based [131, 18, 132, 17, 133] methods to map an image back into a pre-trained GAN's latent space. The predicted latent codes can then be modified and fed back into the generator for corresponding semantic editing. Various studies have pursued semantically meaningful paths for editing latent codes through supervised methods such as annotated images [33, 32] or pre-defined semantics attribute predictors [134, 24, 135], and unsupervised methods such as closed-form factorization [136] or PCA in the latent space [137]. With StyleGAN's advancement, text descriptions, such as those employed by TediGAN [52] and Style-CLIP [138], are gaining popularity for text-based image manipulation. However, these methods rely on image inversion techniques that may unintentionally alter unedited areas due to information loss in the inversion process.

Chapter 3

Visual Style Prompt Learning for Blind Face Restoration

3.1 Introduction

Blind face restoration is dedicated to reconstructing high-fidelity facial images from a range of unidentified sources of degradation such as blurring, noise, downsampling, and compression artifacts [139, 140]. Given the complexity and unpredictability of the degradation in real-world situations, only minimal information could be utilized from the compromised face images [141, 142]. Reconstructing faithful facial features from blindly degraded images stands as a pivotal challenge in the fields of computer vision and image processing [143].

Current prior-knowledge-based methods are leading the way in achieving high-quality face restoration. These methods use geometric priors, drawing on shape information such as parsing maps [34], component heatmaps [35], and identity [36] to guide the restoration process. Nonetheless, due to their focus on limited facial features, the results often lack comprehensive detail [37]. To address this issue, some studies [38, 39, 37, 40] have investigated the use of facial priors inherently present in pre-trained generative models [41, 42] or vector quantization dictionary [43], which show promise for guiding restorations that retain detailed facial attributes. However, these techniques do not focus primarily on the explicit estimation of latent feature representations in a pre-trained model, which might not always lead to effective restoration results when the estimated representation is not correctly predicted.

Recent studies have shown that using pre-trained embedding models as visual prompts [144, 145] enhances restoration processes, highlighting the importance of visual priors for restoration quality. Yet, these features, not being from generative models, lack easy visualization and interpretation, which constrains understanding of their impact on performance. In contrast, generative models provide dense latent representations that encapsulate visual styles and attributes, offering valuable guidance for restoration. By leveraging generative models' latent representations, we can introduce a more intuitive and reliable means to direct the face restoration process. Nonetheless, research in this area remains limited.

In this paper, we explore the use of latent representations in pre-trained generative models as visual style prompts for directing the face restoration process. Specifically, there are mainly two questions: (1) how to accurately embed visual prompts from a degraded face image, and (2) how to integrate the visual prompts and facial priors within the network to boost performance. For the first question, GAN inversion methods can effectively embed a given high-quality image to the corresponding latent space. However, the optimization-based [130] is time-consuming, and the encoder-based methods [132] offer faster but sometimes less precise mapping. Recent studies on Diffusion Probabilistic Models (DMs) [54] show their potential in learning distributions effectively. Yet, their use in embedding latent representations is still not widely explored. Also, the application of these models in processing degraded face images remains insufficiently studied. For the second question, it is equally vital for the network to adeptly utilize valuable guidance to improve restoration quality. Current techniques like spatial feature transform [38] and deformable operations [146] yield promising feature extraction but are limited by the narrow receptive fields of convolutional layers, restricting global context usage. While Transformer-based methods [37, 40] excel in face restoration, they typically require substantial GPU memory and computational resources. Moreover, these methods are not designed to utilize visual prompts to enhance performance.

To tackle these challenges, we first explore the role of Diffusion Models (DMs) in improving the estimation of clean latent representations from degraded images, aiding facial feature extraction for restoration. Secondly, we evaluate how rescaling and adjusting multiscale convolutional kernels according to visual prompts can enhance feature extraction, and better utilize visual information across different receptive fields.

We thus propose a novel visual style prompt learning framework for blind face restoration. Thanks to the well-disentangled and capable StyleGAN [60] latent space in representing diverse facial attributes, we leverage the latent representations of a pre-trained StyleGAN model as visual prompts to generate potential facial features and inform the subsequent restoration process. To efficiently transform a degraded face image into high-quality latent representations, a diffusion-based style prompt module progressively performs denoising steps, refining low-quality visual prompts into higher-quality (clean) counterparts. The resulting candidate facial features and clean visual prompts are then employed to guide the restoration process within the restoration auto-encoder. Furthermore, we introduce a Style-Modulated AggRegation Transformation (SMART) layer to fully utilize visual prompts and capture both contextual information and detailed patterns for better context reasoning. Extensive comparisons and analysis compared to the state-of-the-art (SOTA) methods on four public datasets demonstrate the effectiveness of our approach in achieving high-quality blind face restoration. We also extend our algorithm to various applications, such as facial landmark detection and face emotion recognition to demonstrate the applicability of our method in helping address face-related tasks.

In summary, our paper offers four key contributions: (1) We proposed a novel diffusion-based style prompt method that explicitly predicts visual prompts from degraded images employing DMs within the latent space of a generative model. (2) A novel style-modulated aggregation transformation layer is proposed, which effectively leverages visual prompts and excels in capturing comprehensive contextual information and detailed patterns for boosting performance. (3) We introduce a novel visual style prompt learning framework for high-quality blind face restoration, which explicitly predicts visual prompts through the style prompt module and enhances the clarity and detail of restoration by employing style-modulated aggregation transformation layers. (4) We conducted various experiments to demonstrate that our method not only excels in enhancing the quality of images in both synthetic and real-world blind face restoration datasets but also benefits various applications, highlighting the effectiveness of our proposed method.

3.2 Method

3.2.1 Overview

As shown in Fig. 3.1, given a degraded facial image $\mathbf{I}_{de} \in \mathbb{R}^{h \times w \times 3}$, a ground-truth face image $\mathbf{I}_{gt} \in \mathbb{R}^{h \times w \times 3}$ (with $h \times w$ pixels and three color channels). Let $\mathcal{W}+$ denote the disentangled style latent space [60]. Our framework aims to restore the degraded facial image to achieve both visual and structural fidelity compared to the ground-truth image.

Diffusion-based style prompt module. As shown in Fig. 3.1 (a), our style

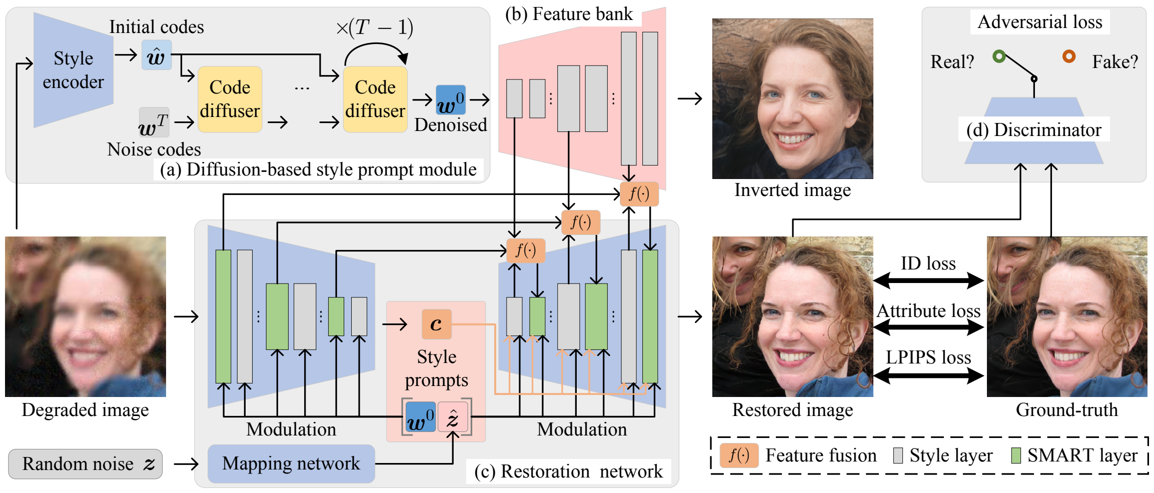


Figure 3.1: The overall pipeline of our framework: the degraded image is processed through a diffusion-based style prompt module (a) to get denoised codes $\boldsymbol{w}^0$ through T time diffusion steps, beginning from noise codes $\boldsymbol{w}^T$. Then, the restoration auto-encoder (c) processes the degraded image, using the denoised codes $\boldsymbol{w}^0$, random codes $\hat{\boldsymbol{z}}$, and a global code $\boldsymbol{c}$ as style prompts. The network also leverages prior features from the facial feature bank (b), integrating them through a fusion process $f(\cdot)$, to achieve the restored image. During training, the restoration auto-encoder is jointly optimized by loss functions.

prompt module is designed to predict denoised latent codes from a degraded facial image. First, a style encoder $E_{\theta_e}(\cdot)$ with the network parameters θ_e, embeds the given degraded image to initial latent codes $\hat{\boldsymbol{w}} \in \mathbb{R}^{512\times N} = E_{\theta_e}(\mathbf{I}_{de}) \in \mathcal{W}+$. Guided by the initial latent codes, our code diffuser $P_{\theta_p}(\cdot)$ (with the network parameters θ_p) samples Gaussian random noise codes $\boldsymbol{w}^T \in \mathbb{R}^{512\times N}$ and then gradually denoise $\boldsymbol{w}^T$ with T steps to get denoised latent codes $\boldsymbol{w}^0 \in \mathbb{R}^{512\times N} \in \mathcal{W}+$. For each step t, code diffuser gradually predict the noise: $\hat{\boldsymbol{\epsilon}}^t \in \mathbb{R}^{512\times N} = P_{\theta_p}(\boldsymbol{w}^t, \hat{\boldsymbol{w}}, t)$. The denoising step is performed by subtracting the predicted noise from $\boldsymbol{w}^t \in \mathbb{R}^{512\times N}$. N is the number of style vectors in latent codes.

Facial feature bank. As depicted in Fig. 3.1 (b), we leverage the StyleGAN generator S_{θ_s} as our facial feature bank. The generator uses denoised latent codes $\boldsymbol{w}^0$ to produce multi-scale, coarse facial features. From S_{θ_s}, we can obtain a set of facial feature maps $\mathcal{F}^s = \{\mathbf{F}_i^s \in \mathbb{R}^{\hat{h}_i\times\hat{w}_i\times\hat{c}_i} | i \in [1, N]\}$ and a inverted image $\mathbf{I}_{ve} \in \mathbb{R}^{h\times w\times 3}$, such that $(\mathcal{F}^s, \mathbf{I}_{ve}) = S_{\theta_s}(\boldsymbol{w}^0)$. The dimensions $\hat{h}_i \times \hat{w}_i \times \hat{c}_i$ represent the size of the feature maps at the i-th layer.

Restoration auto-encoder. As shown in Fig. 3.1 (c), a multi-layer fully-connected neural mapping network F_{θ_f} with the network parameters θ_f, first set a random noise vector $\boldsymbol{z} \in \mathbb{R}^{512\times 1}$ as the input to get random styles $\hat{\boldsymbol{z}} \in \mathbb{R}^{512\times 1} = F_{\theta_f}(\boldsymbol{z})$. The restoration auto-encoder G_{θ_g} leverages $\mathbf{I}_{de}$, $\boldsymbol{w}^0$, $\hat{\boldsymbol{z}}$, and $\mathcal{F}^s$ to generate an restored image $\mathbf{I}_{out} = G_{\theta_g}(\mathbf{I}_{de}, \boldsymbol{w}^0, \hat{\boldsymbol{z}}, \mathcal{F}^s) \in \mathbb{R}^{h\times w\times 3}$. The restoration auto-encoder can be further divided into an encoder G^{en} and a decoder G^{de}, i. e., $G_{\theta_g} = \{G^{en}, G^{de}\}$.

Discriminator. A discriminator network D_{θ_d} with trainable parameters θ_d, is trained to distinguish real images from generated ones (*e.g.*, $\mathbf{I}_{out}$ or $\mathbf{I}_{gt}$). For a given image $\mathbf{I}$, we have $D_{\theta_d}(\mathbf{I}) \in \mathbb{R}^{1\times 1}$.

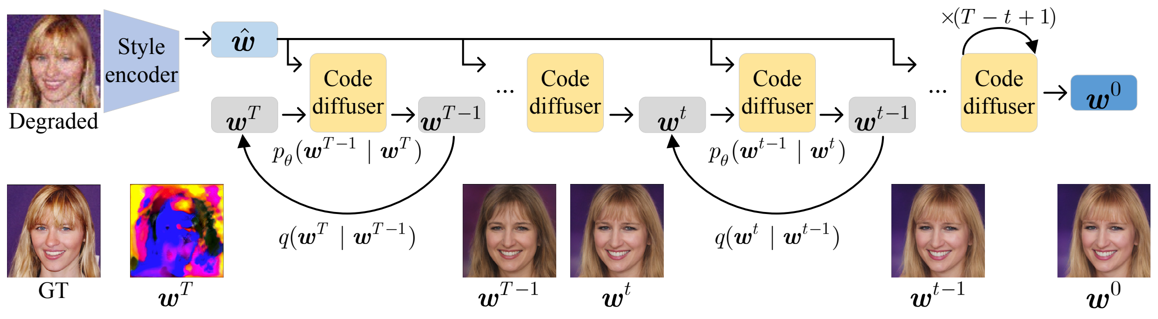

Degraded
Style encoder
$\hat{\boldsymbol{w}}$
Code diffuser
$\boldsymbol{w}^T$
$\boldsymbol{w}^{T-1}$
$p_\theta(\boldsymbol{w}^{T-1} \mid \boldsymbol{w}^T)$
$q(\boldsymbol{w}^T \mid \boldsymbol{w}^{T-1})$
$\boldsymbol{w}^t$
$\boldsymbol{w}^{t-1}$
$p_\theta(\boldsymbol{w}^{t-1} \mid \boldsymbol{w}^t)$
$q(\boldsymbol{w}^t \mid \boldsymbol{w}^{t-1})$
$\times(T-t+1)$
$\boldsymbol{w}^0$
GT

Figure 3.2: The detailed diffusion (←) and denoising (→) processes in the style latent space. We also show the corresponding inverted images of latent codes in steps.

3.2.2 Diffusion-based style prompt module

Guided by the initial codes $\hat{\boldsymbol{w}}$, we adopt diffusion models (DMs) [54] by using our code diffuser to generate denoised latent codes. DMs have a diffusion process and a denoising process, where the former is for training, and the latter is to sample latent codes from random Gaussian noise.

As shown in Fig. 3.2, the diffusion process (a.k.a. forward process) incrementally introduces Gaussian noise into the data. This process transforms the clean latent codes $\boldsymbol{w}^0$ into an approximately pure Gaussian noise $\boldsymbol{w}^T$ using a variance schedule $\beta_1, \ldots, \beta_T$. The diffusion process is defined as:

$$q(\boldsymbol{w}_t|\boldsymbol{w}_{t-1}) = \mathcal{N}(\boldsymbol{w}_t; \sqrt{1-\beta_t}\boldsymbol{w}_{t-1}, \beta_t \mathbf{I}). \tag{3.1}$$

In this process, $\boldsymbol{w}^t$ can be directly approximated by $\boldsymbol{w}^t = \sqrt{\bar{\alpha}_t}\boldsymbol{w}^0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$, with $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$, $\alpha_t = 1-\beta_t$, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

As shown in Fig. 3.2, the denoising process (a.k.a. reverse process) is designed to sample the cleaner version $\boldsymbol{w}^{t-1}$ from $\boldsymbol{w}^t$ by estimating the added noise, which can

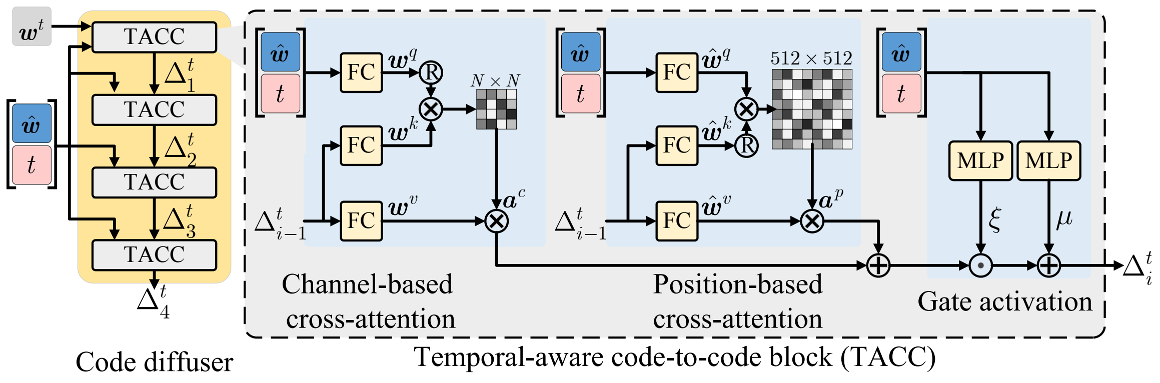


Figure 3.3: The architecture of the coder diffuser and temporal-aware code-to-code block.

be defined as:

$$p_\theta(\boldsymbol{w}_{t-1}|\boldsymbol{w}_t) = \mathcal{N}(\boldsymbol{w}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{w}_t, t), \boldsymbol{\Sigma}_\theta(\boldsymbol{w}_t, t)). \tag{3.2}$$

To sample the denoised latent codes $\boldsymbol{w}^0$ by using our code diffuser, we iteratively denoise $\boldsymbol{w}^t$ from $t = T$ to $t = 1$. The Eq. 3.2 is implemented as:

$$\boldsymbol{w}^{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{w}^t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\epsilon}}^t\right) + \sigma_t\bar{\boldsymbol{\epsilon}}, \tag{3.3}$$

where $\hat{\boldsymbol{\epsilon}}^t = P_{\theta_p}(\boldsymbol{w}^t, \hat{\boldsymbol{w}}, t)$; variance $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$ and noise $\bar{\boldsymbol{\epsilon}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

Code diffuser. Fig. 3.3 shows that our code diffuser has four temporal-aware code-to-code blocks (TACC). Our TACC block is extended from FFCLIP's semantic modulation block [29]. Each block sets Δ_{i-1}^t ($\Delta_0^t = \boldsymbol{w}^t$), initial codes $\hat{\boldsymbol{w}}$, and denoising step t as inputs, and outputs the intermediate results Δ_i^t $(i = 1, 2, 3, 4)$:

$$\begin{aligned}
&\boldsymbol{w}^q = \mathrm{FC}([\hat{\boldsymbol{w}}, t]), \boldsymbol{w}^k = \mathrm{FC}(\Delta_{i-1}^t), \boldsymbol{w}^v = \mathrm{FC}(\Delta_{i-1}^t),\\
&\boldsymbol{a}^c = \boldsymbol{w}^v \cdot \mathrm{Softmax}(\boldsymbol{w}^{q^\top} \cdot \boldsymbol{w}^k / \tau_1),\\
&\hat{\boldsymbol{w}}^q = \mathrm{FC}([\hat{\boldsymbol{w}}, t]), \hat{\boldsymbol{w}}^k = \mathrm{FC}(\Delta_{i-1}^t), \hat{\boldsymbol{w}}^v = \mathrm{FC}(\Delta_{i-1}^t),\\
&\boldsymbol{a}^p = \mathrm{Softmax}(\hat{\boldsymbol{w}}^q \cdot \hat{\boldsymbol{w}}^{k^\top} / \tau_2) \cdot \hat{\boldsymbol{w}}^v,\\
&\boldsymbol{\xi} = \sigma(\mathrm{MLP}([\hat{\boldsymbol{w}}, t])), \boldsymbol{\mu} = \phi(\mathrm{MLP}([\hat{\boldsymbol{w}}, t])),\\
&\Delta_i^t = \mathrm{LayerNorm}(\boldsymbol{a}^c + \boldsymbol{a}^p) \odot (\boldsymbol{\xi} + 1) + \boldsymbol{\mu},
\end{aligned} \tag{3.4}$$

where $[\cdot]$ is the concatenation operation; $\mathrm{FC}(\cdot)$ is a fully connected layer; $\mathrm{MLP}(\cdot)$ is a stack of two fully connected layers; $\mathrm{Softmax}(\cdot)$ is the softmax activation; $\sigma(\cdot)$ denotes the sigmoid activation function; $\phi(\cdot)$ corresponds to the LeakyReLU activation function with the negative slope of 0.2; $\mathrm{LayerNorm}(\cdot)$ is the LayerNorm layer. We set $\tau_1 = \sqrt{N}$, and $\tau_2 = \sqrt{512}$. Our blocks compute channel-based attention [147] ($\boldsymbol{a}^c$), position-based attention [147] ($\boldsymbol{a}^p$), and gated maps $\boldsymbol{\xi}$ and the bias $\boldsymbol{\mu}$ using time step t, initial codes $\hat{\boldsymbol{w}}$ and intermediate results Δ_i^t layer by layer to predict the noise $\hat{\boldsymbol{\epsilon}}^t = \Delta_4^t$.

3.2.3 Restoration auto-encoder

To produce high-quality restoration results, we develop a restoration auto-encoder that fully utilizes denoised latent codes and facial priors to recover the given degraded image $\mathbf{I}_{de}$. We introduce the SMART layer to fully capture contextual information and detailed patterns for enhancing context reasoning, while fully using latent codes

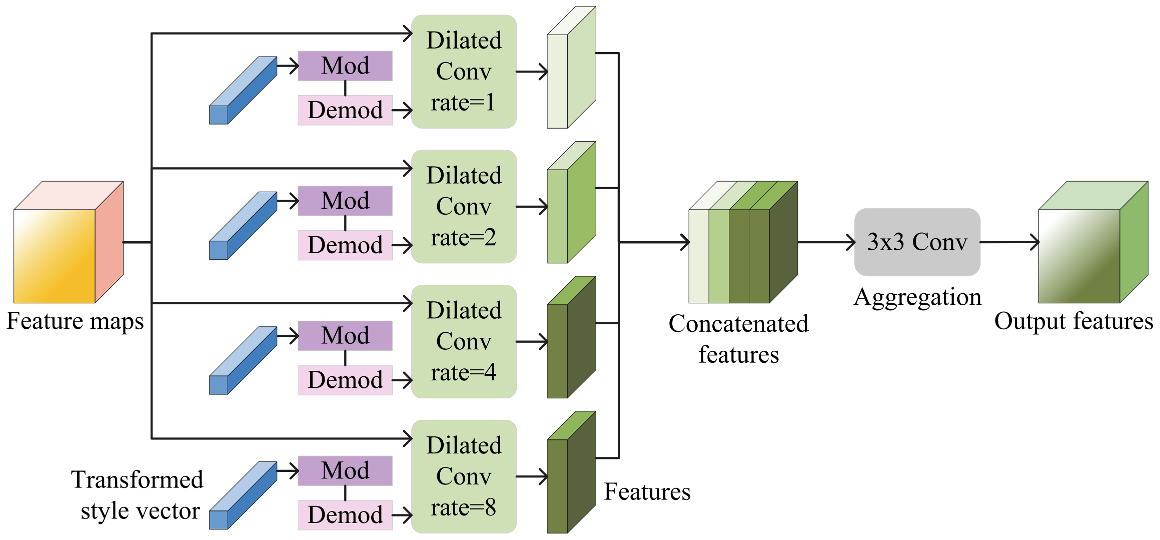


Figure 3.4: Illustration of the style-modulated aggregation transformation (SMART).

to guide the restoration process.

As shown in Fig. 3.1 (c), given denoised latent codes $\boldsymbol{w}^0$, random styles $\hat{\boldsymbol{z}}$ and facial priors $\mathcal{F}^s = \{\mathbf{F}_i^s \in \mathbb{R}^{\hat{h}_i \times \hat{w}_i \times \hat{c}_i} | i \in [1, N]\}$. The features $\mathcal{F}^{en} = \{\mathbf{F}_i^{en} \in \mathbb{R}^{\hat{h}_i \times \hat{w}_i \times \hat{c}_i} | i \in [1, N]\}$ in our restoration encoder G_{en} are defined as:

$$\mathbf{F}_i^{en} = \begin{cases} \text{SMART}(\mathbf{F}_{i-1}^{en}, [\boldsymbol{w}_i^0, \hat{\boldsymbol{z}}]), & \text{if } i \bmod 2 = 1; \\ \text{Down}(\text{SC}(\mathbf{F}_{i-1}^{en}, [\boldsymbol{w}_i^0, \hat{\boldsymbol{z}}])), & \text{otherwise,} \end{cases} \tag{3.5}$$

where SC($\cdot$) is the style layer; Down($\cdot$) indicates the the downsample operation; $\mathbf{F}_0^{en} = \text{Conv}(\mathbf{I}_{de})$. When $i = N$, we can further get the global code $\boldsymbol{c} = \text{FC}(\mathbf{F}_N^{en})$. Our restoration decoder then generate features $\mathcal{F}^{de} = \{\mathbf{F}_i^{de} \in \mathbb{R}^{\hat{h}_i \times \hat{w}_i \times \hat{c}_i} | i \in [1, N]\}$, which can be calculated as follows:

$$\mathbf{F}_i^{de} = \begin{cases} \text{SC}(\text{UP}(\mathbf{F}_{i-1}^{de}), [\boldsymbol{w}_i^0, \hat{\boldsymbol{z}}, \boldsymbol{c}]), & \text{if } i \bmod 2 = 1; \\ \text{SMART}(\hat{\mathbf{F}}_{i-1}^{de}, [\boldsymbol{w}_i^0, \hat{\boldsymbol{z}}, \boldsymbol{c}]), & \text{otherwise,} \end{cases} \tag{3.6}$$

where $\mathbf{F}_0^{de} = \text{Conv}(\mathbf{F}_N^{en})$, $\hat{\mathbf{F}}_{i-1}^{de} = f(\mathbf{F}_{i-1}^{de}, \mathbf{F}_{N-i+1}^{en}, \mathbf{F}_{i-1}^{s})$, $\mathbf{I}_{out} = \text{Conv}(\mathbf{F}_N^{de})$. We simply use the summation of input features as our fusion function $f(\cdot)$.

Style-modulated aggregation transformation. As shown in Fig. 3.4, the SMART layer is designed to leverage style prompts and various dilation factors to widen the receptive field. This enables the network to capture detailed patterns and informative distant image contexts for better reasoning.

Given input feature maps $\bar{\mathbf{F}}$ and corresponding style prompts $\bar{\boldsymbol{w}}$ ($[\boldsymbol{w}_i^0, \hat{\boldsymbol{z}}]$ for layers in restoration encoder, $[\boldsymbol{w}_i^0, \hat{\boldsymbol{z}}, \boldsymbol{c}]$ for layers in restoration decoder, where $\boldsymbol{w}_i^0 \in \mathbb{R}^{512\times1}$, $\hat{\boldsymbol{z}} \in \mathbb{R}^{512\times1}$, $\boldsymbol{c} \in \mathbb{R}^{512\times2}$). We first generate a style vector $\boldsymbol{s} \in \mathbb{R}^{512\times1}$ for the subsequent modulation and demodulation [60] to rescale Convolution kernels $\boldsymbol{k}$, *e.g.*, $\boldsymbol{k} \in \mathbb{R}^{512\times3\times3\times512}$. The Convolution layers $\text{Conv}_{d=m}^{\boldsymbol{k}'}$, with the reweighted $\boldsymbol{k}'$ and dilation factor m are then applied to extract feature maps to get results. Our SMART layer is expressed as follows:

$$\begin{aligned}
&\boldsymbol{s} = \text{FC}(\bar{\boldsymbol{w}}), \boldsymbol{k}' = \text{Demod}(\text{Mod}(\boldsymbol{k}, \boldsymbol{s})),\\
&\mathbf{F}^1 = \phi(\text{Conv}_{d=1}^{\boldsymbol{k}'}(\bar{\mathbf{F}})), \mathbf{F}^2 = \phi(\text{Conv}_{d=2}^{\boldsymbol{k}'}(\bar{\mathbf{F}})),\\
&\mathbf{F}^4 = \phi(\text{Conv}_{d=4}^{\boldsymbol{k}'}(\bar{\mathbf{F}})), \mathbf{F}^8 = \phi(\text{Conv}_{d=8}^{\boldsymbol{k}'}(\bar{\mathbf{F}})),\\
&\hat{\mathbf{F}} = \text{Aggregation}([\mathbf{F}^1, \mathbf{F}^2, \mathbf{F}^4, \mathbf{F}^8]]),
\end{aligned} \tag{3.7}$$

where the modulation operation $\text{Mod}(\cdot)$ is employed to scale the convolution weights using given transformed $\boldsymbol{s}$ style vector, while the demodulation operation $\text{Demod}(\cdot)$ is utilized to rescale kernels back to unit standard deviation for stable training. We employ a 3×3 convolution layer as the aggregation layer to merge features from the four different dilation rates, thereby achieving more distinct facial structures. Incorporating the SMART layer within the restoration network enables our model to grasp the global facial structure and local details. It also facilitates control over both global and local stylistic features using modulated styles, thereby enhancing performance.

3.2.4 Module training

We utilize the pre-trained StyleGAN generator S_{θ_s}. We first train the style encoder E_{θ_e}, and then optimize the code diffuser P_{θ_p}. Finally, we detail the training for both the restoration auto-encoder and the discriminator.

Training of style encoder. We adopt the style encoder network architecture and training objectives from e4e [132]. Throughout the training phase, degraded images are input into the style encoder $E_{\hat{\theta}_e}$.

Training of code diffuser. To learn the diffusion process for the generation of denoised latent codes, we reparameterize the learnable Gaussian transition as our code diffuser $P_{\theta_p}(\cdot)$, and use the commonly used optimization objectives, including diffusion loss, Learned Perceptual Image Patch Similarity (LPIPS) loss, and identity loss to constrain the training process.

Diffusion loss. The diffusion loss $\mathcal{L}_{dm}$ [54] is presented as:

$$\mathcal{L}_{dm}^{(\theta_p)}(\boldsymbol{\epsilon}, \hat{\boldsymbol{\epsilon}}^t) = \mathbb{E}_{t,\boldsymbol{w}^0,\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}^t\|^2\right], \tag{3.8}$$

where $\hat{\boldsymbol{\epsilon}}^t = P_{\theta_p}(\boldsymbol{w}^t, \hat{\boldsymbol{w}}, t)$. By minimizing $\mathcal{L}_{dm}$, our code diffuser can be trained to learn how to predict the added Gaussian noise at step t.

LPIPS loss. To enhance the fidelity of the denoised latent codes, we apply the LPIPS loss [148] to constrain the perceptual similarity between the inverted image $\mathbf{I}_{ve}$ and the ground-truth $\mathbf{I}_{gt}$ using StyleGAN model S_{θ_s}. When $t = T$, we can generate an inverted image $(\mathcal{F}^s, \mathbf{I}_{ve}) = S_{\theta_s}(\boldsymbol{w}^0)$ from the denoised latent codes $\boldsymbol{w}^0$. The LPIPS loss can be written as:

$$\mathcal{L}_{lpips}^{(\theta_p)}(\mathbf{I}_{ve}, \mathbf{I}_{gt}) = \| \mathrm{VGG}(\mathbf{I}_{ve}) - \mathrm{VGG}(\mathbf{I}_{gt}) \|_2, \tag{3.9}$$

where $\mathrm{VGG}(\cdot)$ is the pre-trained perceptual feature extractor VGG [149].

Identity loss. Similar to the LPIPS loss, we apply the identity loss to constrain identity similarity between the inverted image $\mathbf{I}_{ve}$ and the ground-truth $\mathbf{I}_{gt}$ in the facial embedding space. The identity loss is formulated as:

$$\mathcal{L}_{id}^{(\theta_p)}(\mathbf{I}_{ve}, \mathbf{I}_{gt}) = 1 - \cos\left(\mathrm{R}(\mathbf{I}_{ve}), \mathrm{R}(\mathbf{I}_{gt})\right). \tag{3.10}$$

$\mathrm{R}(\cdot)$ is the pre-trained face recognition ArcFace network [150].

Total loss. The total training loss of our code diffuser is expressed as:

$$O(\theta_p) = \mathcal{L}_{dm}^{(\theta_p)} + \lambda_{lpips}\mathcal{L}_{lpips}^{(\theta_p)} + \lambda_{id}\mathcal{L}_{id}^{(\theta_p)}, \tag{3.11}$$

where $\lambda_{lpips} = 0.1$ and $\lambda_{id} = 0.1$ are weights of objectives. During training, we optimize parameters θ_p by minimizing the total loss.

Training of restoration networks. We train the restoration and mapping networks using LPIPS loss, identity loss, and adversarial loss.

LPIPS loss. The LPIPS loss $\mathcal{L}_{lpips}^{(\theta_g,\theta_f)}(\mathbf{I}_{out}, \mathbf{I}_{gt})$ is applied to enforce perceptual similarity between the restored image $\mathbf{I}_{out}$ and the ground-truth $\mathbf{I}_{gt}$.

Identity loss. The identity loss $\mathcal{L}_{id}^{(\theta_g,\theta_f)}(\mathbf{I}_{out}, \mathbf{I}_{gt})$ is employed to constrain identity similarity between the restored $\mathbf{I}_{out}$ and the ground-truth $\mathbf{I}_{gt}$.

Adversarial loss. We employ the adversarial non-saturating logistic loss [41], with R_1 regularization [151]. The adversarial objective is defined as:

$$\begin{aligned}\mathcal{L}_{adv}^{(\theta_g,\theta_f,\theta_d)}(\mathbf{I}_{out}, \mathbf{I}_{gt}) &= \mathbb{E}_{\mathbf{I}_{out}}[\log(1 - D(\mathbf{I}_{out})] \\ &+\mathbb{E}_{\mathbf{I}_{gt}}[\log(D(\mathbf{I}_{gt}))] - \frac{\gamma}{2}\mathbb{E}_{\mathbf{I}_{gt}}[\|\nabla_{\mathbf{I}_{gt}} D(\mathbf{I}_{gt})\|_2^2],\end{aligned} \tag{3.12}$$

where $\gamma = 10$ is used to balance the R_1 regularization term. The restoration network G learns to produce realistic images $\mathbf{I}_{out}$, while the discriminator D tries to recognize between real $\mathbf{I}_{gt}$ and restored $\mathbf{I}_{out}$ images.

Total objective. The total training objective is expressed as:

$$\begin{aligned}O(\theta_g, \theta_f, \theta_d) &= \mathcal{L}_{adv}^{(\theta_g,\theta_f,\theta_d)}(\mathbf{I}_{out}, \mathbf{I}_{gt}) + \hat{\lambda}_{id}\mathcal{L}_{id}^{(\theta_g,\theta_f)}(\mathbf{I}_{out}, \mathbf{I}_{gt}) \\ &+ \hat{\lambda}_{lpips}\mathcal{L}_{lpips}^{(\theta_g,\theta_f)}(\mathbf{I}_{out}, \mathbf{I}_{gt}).\end{aligned} \tag{3.13}$$

We empirically set $\hat{\lambda}_{id} = 0.1$ and $\hat{\lambda}_{lpips} = 0.5$ in this work. We can obtain the optimized parameters (θ_g, θ_f), and θ_d via the alternating training phases:

$$\begin{aligned}(\theta_g, \theta_f) &= \arg\min_{\theta_g,\theta_f} O(\theta_g, \theta_f, \theta_d), \\ (\theta_d) &= \arg\max_{\theta_d} O(\theta_g, \theta_f, \theta_d).\end{aligned} \tag{3.14}$$

3.3 Experimental results and comparisons

3.3.1 Settings

Implementation. The proposed framework was developed using Python and PyTorch. We borrowed and fine-tuned the style encoder from e4e [132] on our training dataset, training 150,000 iterations with a batch size of 16. For the code diffuser, we trained 200,000 iterations with a batch size of 8, setting the denoising step $T = 4$ for both training and testing. The variance schedule β_t increased linearly from $\beta_1 = 0.1$ to $\beta_T = 0.99$. The restoration auto-encoder was trained on 500,000 iterations using a batch size of 4. The code diffuser and restoration auto-encoder used the Adam optimizer (momentum coefficients 0.5 and 0.999, learning rate 0.0002). All experiments were conducted on the NVIDIA Tesla V100 GPU.

Datasets. We trained our algorithm on the FFHQ dataset [42], which comprises 70,000 high-resolution facial images. Each image was resized to the 512×512 resolution. Utilizing the FFHQ dataset, we generated degraded images following the degradation model described in established literature [37, 36, 38]. The degradation process is $\mathbf{I}_{de} = \left\{ [(\mathbf{I}_{gt} \otimes \mathbf{k}_\sigma) \downarrow_r + \mathbf{n}_\delta]_{JPEG_q} \right\} \uparrow_r$. Here, $\mathbf{I}_{de}$ and $\mathbf{I}_{gt}$ denote a degraded image and its corresponding high-quality counterpart, respectively. $\mathbf{k}_\sigma$ means a Gaussian blur kernel with σ. $\downarrow_r$ corresponds to a bilinear downsampling operation with a scale factor r. $\mathbf{n}_\delta$ is denoted as Gaussian noise with δ. $[\cdot]_{JPEG_q}$ is a JPEG compression with quality factor q. $\uparrow_r$ represents a bilinear upsampling operation with a scale factor r

Table 3.1: Quantitative comparison across multiple metrics on the CelebA-Test dataset. **Bold** values indicate the best results.

Methods	FID† ↓	U-IDS ↑	P-IDS ↑	FID ↓	NIQE ↓	PSNR ↑	SSIM ↑
PSFRGAN [34]	22.93	0	0	43.88	4.27	24.45	0.6308
Wan *et al.* [76]	32.34	0	0	70.21	5.19	23.00	0.6189
PULSE [39]	57.92	0	0	67.75	4.71	21.61	0.6287
GPEN [77]	14.65	0.03%	0	41.99	4.25	**24.63**	0.6476
GFP-GAN [38]	17.29	0	0	42.39	4.58	24.46	**0.6684**
VQFR [146]	13.41	0.02%	0	46.77	4.19	23.76	0.6591
RestoreFormer [37]	12.42	0.06%	0	41.45	4.18	24.42	0.6404
Ours	**11.76**	**2.21%**	**0.40%**	**38.34**	**4.04**	24.60	0.6513

to resize the same size of $\mathbf{I}_{gt}$. During training, the parameters $\sigma \in [0.2, 10]$, $r \in [1, 8]$, $\delta \in [0, 20]$, and $q \in [60, 100]$ were randomly sampled from the given ranges.

For testing, our method was evaluated on both a synthetic dataset (CelebA-Test [152]) and three real-world datasets (LFW-Test [153], CelebChild-Test [38], and WebPhoto-Test [38]). The CelebA-Test [152] comprises 3,000 samples generated from CelebA-HQ images [154] using the above degradation process. LFW-Test has 1,711 images, which are extracted from the first image of each identity in LFW's [153] validation set. CelebChild-Test and WebPhoto-Test [38] contain 180 and 407 degraded facial images, respectively. Note that real-world datasets have no ground-truth images.

Table 3.2: Quantitative comparisons on three real-world datasets in terms of FID and NIQE metrics. **Bold** values indicate the best results.

	LFW-Test		CelebChild-Test		WebPhoto-Test	
Methods	FID ↓	NIQE ↓	FID ↓	NIQE ↓	FID ↓	NIQE ↓
PSFRGAN [34]	49.53	4.13	106.61	4.68	84.98	4.36
Wan *et al.* [76]	60.58	5.09	117.37	5.18	101.37	5.53
PULSE [39]	64.86	4.34	102.74	4.28	86.45	4.38
GPEN [77]	51.92	4.05	107.22	4.46	80.58	4.65
GFP-GAN [38]	49.96	4.56	111.78	4.60	87.35	5.80
VQFR [146]	49.79	**3.82**	114.79	4.42	84.78	4.67
RestoreFormer [37]	48.39	3.97	**101.22**	4.35	77.33	4.38
Ours	**46.48**	3.85	109.52	**4.12**	**74.25**	**4.28**

Metrics. We employed multiple metrics across perceptual, identity, and pixel levels. These include the Fréchet Inception Distance (FID) [155], paired and unpaired inception discriminative scores (P-IDS/U-IDS) [49], the Naturalness Image Quality Evaluator (NIQE), the Peak Signal-to-Noise Ratio (PSNR), and the Structural Similarity Index Measure (SSIM). FID and NIQE have been recognized as benchmark non-reference metrics for assessing the quality of restored images. Unless specified, we set the term FID for unpaired evaluation and FID† for paired evaluation, respectively.

3.3.2 Comparison with SOTA methods

We compared our method with SOTA facial restoration methods, including geometric-prior-based method (PSFRGAN [34]), generative-prior-based methods (Wan *et al.* [76], PULSE [39], GPEN [77], GFP-GAN [38]), and codebook-based methods (VQFR [146], RestoreFormer [37]).

Quantitative performance on synthetic datasets. Table 3.1 presents the quantitative results of compared methods on the CelebA-Test dataset. This table highlights that most generative-prior-based methods tend to outperform others, owing to the extensive facial attributes accessible from pre-trained generative models. Methods like GPEN [77] and GFP-GAN [38], which build upon StyleGAN pre-trained priors, achieve commendable results by integrating pre-trained facial features. However, as these methods implicitly predict latent codes, their performance can be impacted when the latent code predictions are inaccurate. Approaches using reconstruction-oriented dictionaries, such as VQFR [146] and RestoreFormer [37], provide a wealth of high-quality facial features, significantly enhancing restoration quality. However, the vector quantization process [43] may introduce quantization error since continuous data must be mapped to the nearest discrete code. That can result in a loss of detail, especially for subtle facial features that are important for a realistic reconstruction. In comparison, our method outperforms others in FID†, U-IDS, P-IDS, FID, and NIQE metrics, showcasing superior restoration quality. It also shows competitive performance in additional metrics, reflecting its effective handling in pixel-level

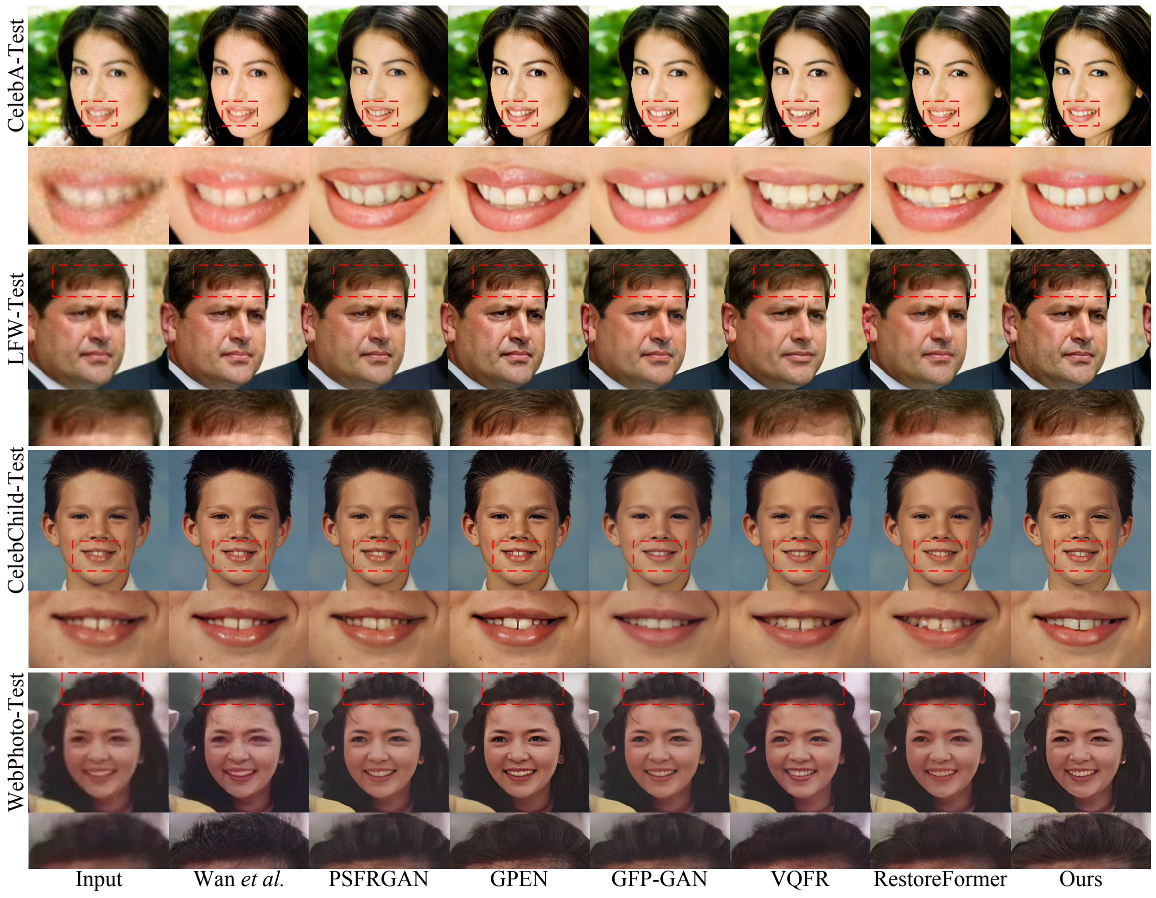


Figure 3.5: Visual comparisons of our method and the SOTA facial restoration methods.

similarity.

Quantitative performance on real-world datasets. Table 3.2 presents evaluations of methods on LFW-Test, CelebChild-Test, and WebPhoto-Test datasets to compare their generalization capabilities. Methods such as Wan *et al.* [76] and PULSE [39], which do not incorporate identity information from degraded images, may experience a drop in performance. GPEN [77] and GFP-GAN [38] apply generative facial priors, yet their local feature fusion might constrain their effectiveness.

Approaches like VQFR [146] and RestoreFormer [37] rely on a fixed codebook, which may restrict their ability to capture the fine-grained facial attributes. Our method outperforms the most compared methods in FID and NIQE metrics on three datasets, thanks to its strong visual style prompt learning and aggregated feature fusion across a large receptive field, demonstrating superior restoration quality.

Qualitative performance. Fig. 3.5 presents the visual comparisons across four datasets. While most existing approaches can produce high-fidelity faces from degraded images, our method attains more detail in critical regions, such as the hair and mouth. The results indicate that methods by Wan *et al.* [76] sometimes struggle to preserve the identity presented in the original images. GPEN [77] and GFP-GAN [38] demonstrate improved performance, yet they occasionally introduce visual artifacts on facial features. VQFR [146] and RestoreFormer [37] exhibit clearer facial features. In contrast, our approach excels in restoring facial details with clearer facial features and fewer artifacts, owing to our visual style prompt learning that precisely predicts style prompts from degraded images and our SMART layer that captures informative distant image contexts for better reasoning.

3.3.3 Ablation study

We conducted ablation studies to show the impact of individual components within our framework. This involved selectively omitting each component and evaluating its influence on restoration performance. Note that we used the same training settings

Table 3.3: Ablation study of the framework components: (a) removing our coder diffuser, (b) replacing our SMART layers with style layers, (c) removing the modulation of style codes, (d) removing facial priors in fusion processes, and the full configurations (ours). **Bold** values indicate the best results.

	LFW-Test		CelebChild-Test		WebPhoto-Test	
Methods	FID $\downarrow$	NIQE $\downarrow$	FID $\downarrow$	NIQE $\downarrow$	FID $\downarrow$	NIQE $\downarrow$
(a)	46.58	4.13	111.33	4.26	75.89	4.66
(b)	48.42	**3.85**	110.18	4.14	74.99	**4.20**
(c)	47.30	3.88	109.76	4.16	75.89	4.32
(d)	48.58	3.87	109.68	4.22	76.15	4.29
Ours	**46.48**	**3.85**	**109.52**	**4.12**	**74.25**	4.28

as mentioned in Subsection 3.3.1.

Table 3.3 presents the quantitative outcomes for various configurations. In most cases, our fully-equipped model surpasses all alternative variants in FID and NIQE metrics, demonstrating its good restoration capabilities. When removing our coder diffuser (a), the restoration network utilizes the initial codes as the style prompts. Since the initial codes might not accurately represent clean facial attributes from degraded images, we observed a noticeable deterioration in metric scores. When replacing the SMART with the style layer (b), the network shows limited capabilities to dynamically grasp the global facial structure and local details, leading to worse

FID and NIQE performance. Removing style code modulation (c) implies that the model is limited to leveraging guidance information from the style prompts to rescale convolution kernels, leading to suboptimal performance and showing the significance of style prompt modulation. Omitting facial priors in the fusion process (d) effectively degrades the fusion mechanism to a basic Unet skip connection, with the resultant decline in FID and NIQE scores. It shows that the guidance of facial priors helps to achieve better performance. Our model excels by integrating finely-tuned style prompts via our coder diffuser, adeptly capturing both local and global facial features with our SMART layer, and enriching visual quality through the utilization of spatial facial features from the restoration network and facial priors.

Figure 3.6 illustrates the visual performance corresponding with each alternative model. Removing our coder diffuser (a) leads to the appearance of noticeable artifacts in key areas, including the mouth and decorative features. Replacing the SMART layers with style layers (b) slightly compromises the facial structures and details. For instance, the restoration of the cloth in the second row appears blurred, likely due to the restricted receptive field of the style layer. The absence of style code modulation (c) leads to a decline in image quality. Specifically, in the first row, model (c) exhibits color discrepancies between the hair and sunglasses. This may caused by the missing modulation of the consistent styles. Removing facial priors from the fusion process impacts the fidelity of facial structures. For instance, as observed in the third row (d), the restored glasses' structure is not symmetrical. In

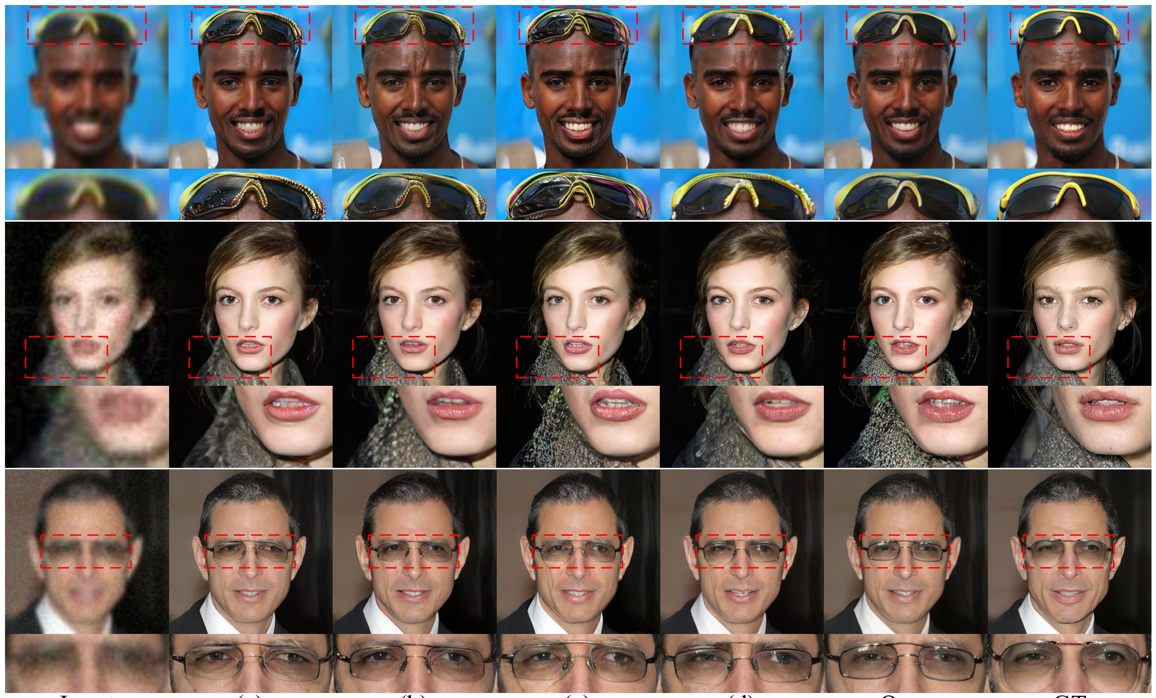


Figure 3.6: Visual examples from the ablation study with (a) removing the coder diffuser, (b) replacing our SMART layers with style layers, (c) removing the modulation of style codes, (d) removing facial priors in fusion processes, and the full configurations (ours).

contrast, our full model successfully recovers structures and details in critical areas, showcasing superior restoration capabilities.

Analysis of style prompt learning. Here, we assessed the quality of the denoised latent codes by feeding them into the pre-trained StyleGAN generator to produce inverted images on the CelebA-Test dataset. The ground-truth latent codes were generated by feeding high-quality (ground-truth) images into the style encoder. As shown in Fig. 3.7, we visualized these latent codes, degraded images, restored

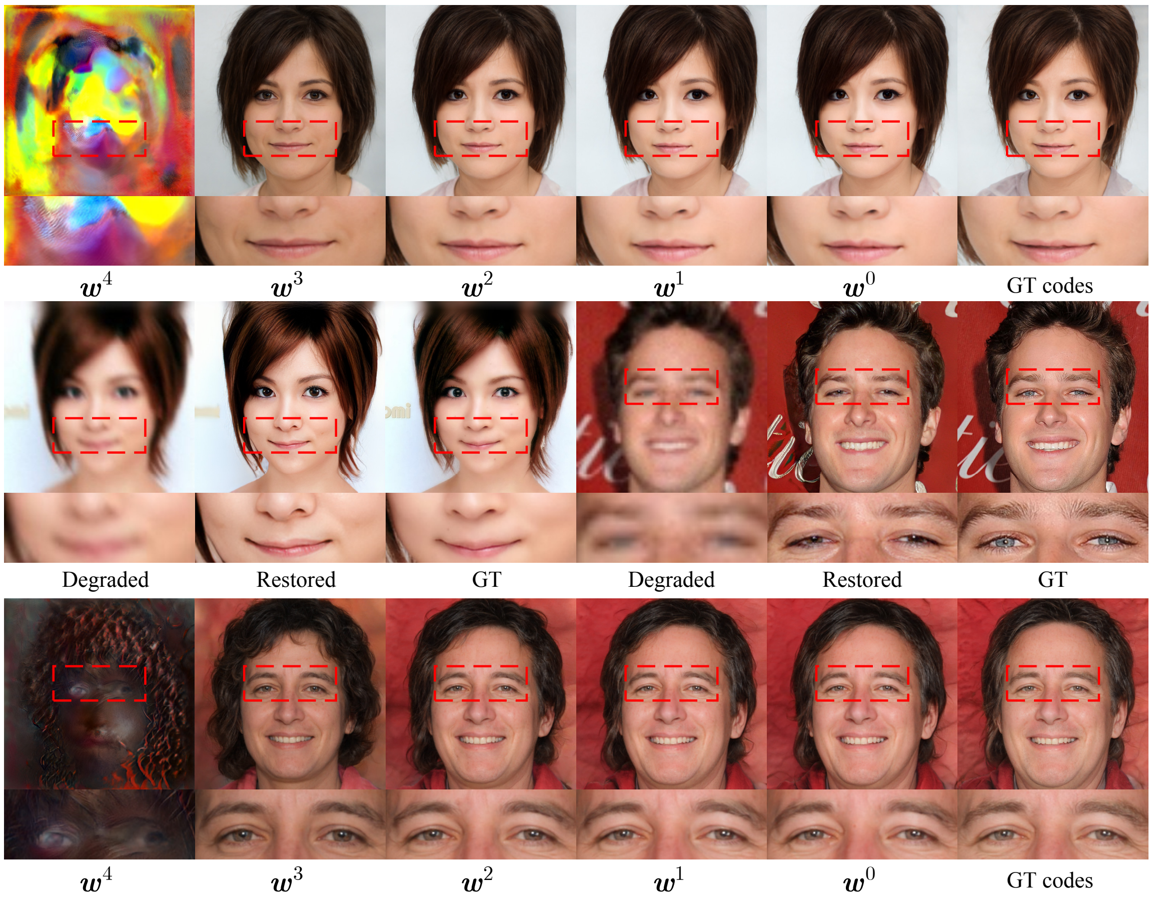


Figure 3.7: Visualization of denoised latent codes and restored images. Through the diffusion process, the denoised latent codes $\boldsymbol{w}^t$ incrementally reveal more clear facial features.

images, and ground-truth images for better analysis. With more diffusion steps employed, the denoised latent codes increasingly reflect more distinct facial features and exhibit greater fidelity. For instance, attributes such as facial pose, expression, and hairstyle were closely aligned with the ground-truth latent codes, underscoring the efficacy of visual style prompt learning in providing consistent coarse facial features. Utilizing our restoration auto-encoder, the restored images showcase high-quality out-

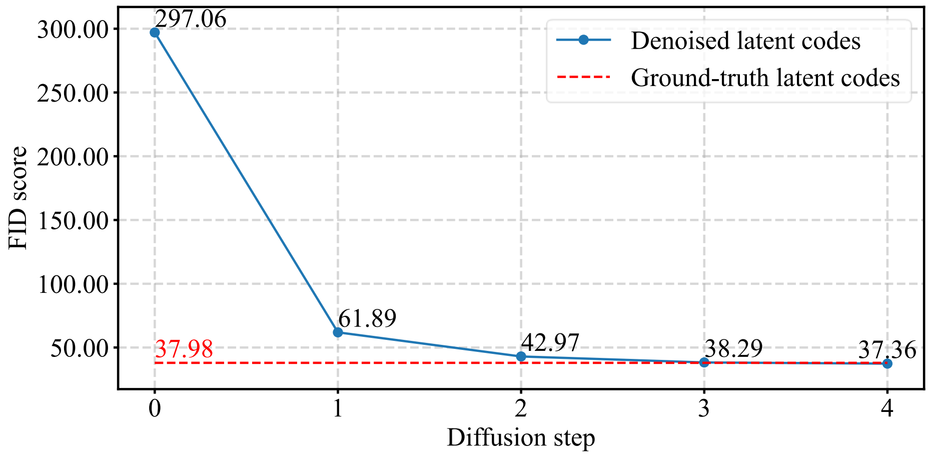


Figure 3.8: FID comparisons between denoised latent codes and ground-truth latent codes.

comes that closely resemble the ground-truth images. Table 3.3 (a) and Fig. 3.7 show the effectiveness of our diffusion-based style prompt module in improving restoration.

Fig. 3.8 shows FID scores for inverted images, demonstrating consistent improvements in latent codes with more diffusion steps, but saturating at step 4. At this step, the denoised codes even surpass the ground-truth FID scores, showing our code diffuser's ability to produce high-quality style prompts.

Analysis of SMART layer. As illustrated in Fig. 3.9, we further explored the impact of the SMART layer across varying resolutions during restoration. We visualized the feature maps generated at dilation rates of 1, 2, 4, and 8, alongside their aggregation layers, spanning resolutions from 32 to 256. A dilation rate of 1 essentially converts the layer back to an original style layer. Our observations demonstrate that at varying scales, lower dilation rates (1 and 2) typically emphasize local

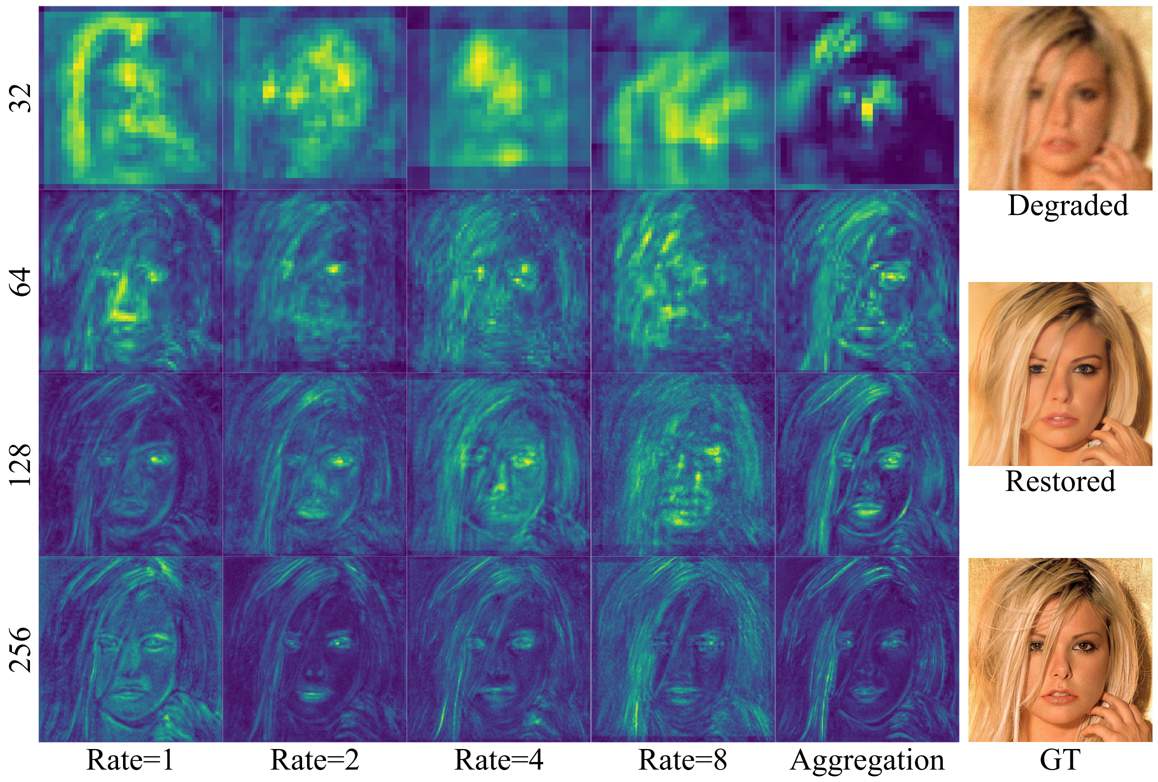


Figure 3.9: Visualization of feature maps from SMART layers at resolutions ranging from 32 to 256. We show results across different dilation rates (1, 2, 4, 8) and aggregation layers.

features, whereas higher dilation rates shift the network's focus towards more global attributes. The aggregation process further assigns higher importance to the more informative and detailed facial features while reducing the impact of less informative features. For example, at a 64 resolution, feature maps produced from a dilation rate of 1 mainly focus on the nose and eye regions. In contrast, those from a dilation rate of 8 encompass the global facial features, even though with some noise. The aggregation process integrates features across these four dilation rates, resulting in clearer

Table 3.4: Quantitative comparisons of facial landmark detection in detection success rate (DSR) and normalized mean error (NME). **Bold** values indicate the best results.

Method	DSR ↑	NME ↓
FAN	99.73%	6.08%
Our improved	**100.00%**	**2.43%**

facial structures. Table 3.3 (b) and visual results demonstrate the SMART layer's effectiveness in producing key facial features for restoring natural facial attributes.

3.4 Applications

We show our method's effectiveness in improving facial landmark detection and emotion recognition. We tested on the CelebA-Test dataset.

3.4.1 Facial landmark detection.

By incorporating our method into a face alignment network (FAN [156]), we built our improved facial landmark detection algorithm (Ours + FAN). For comparison, we set the degraded images as inputs. The landmarks detected on the ground-truth images were designated as ground-truth landmarks. We recorded the detection success rate (DSR%) for the face images and the normalized mean error (NME%) for landmarks.

Table 3.4 shows quantitative comparisons between FAN and our improved version. Incorporating our facial restoration method significantly enhances detection

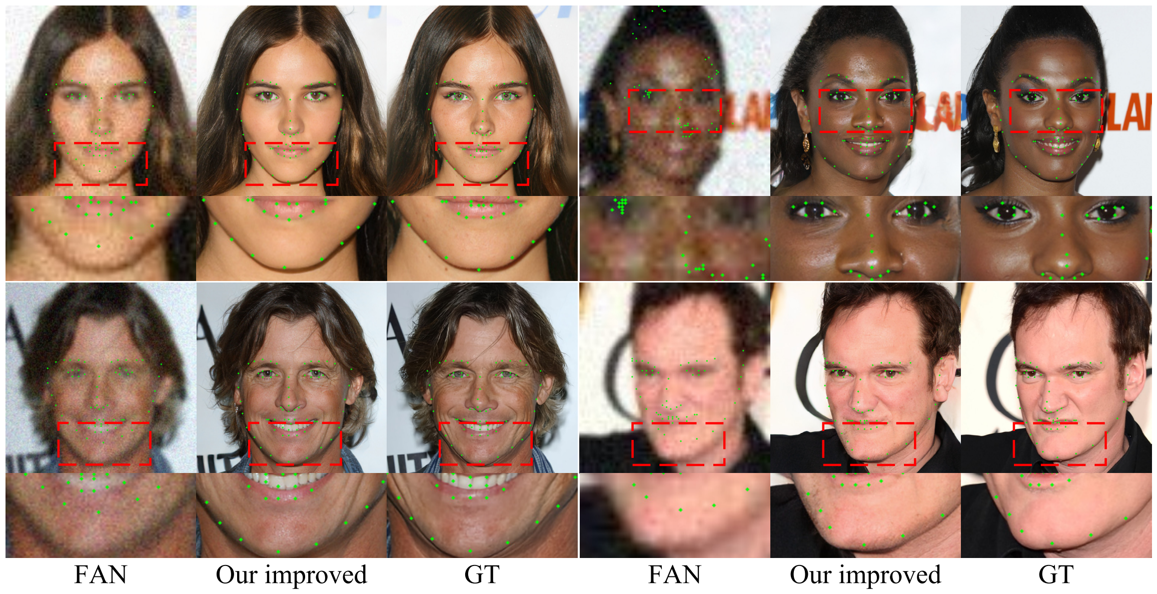


Figure 3.10: Visualization of facial landmark detection on degraded images, restored images, and ground-truth images. Green points correspond to visualized facial landmarks.

outcomes, improving the detection success rate to 100% and reducing the NME score to get a relative improvement of 3.65%. This indicates that our improved version markedly enhances the detection process by restoring clearer facial features for more accurate identification.

Figure 3.10 further visualizes the detection comparisons. It shows that performing facial landmark detection directly on degraded images often leads to inaccurate results. After incorporating our restoration method, the results in recovered face images display distinct facial features, which significantly enhance the subsequent facial landmark detection process.

Table 3.5: Face emotion recognition accuracy using HSEmotion and our enhanced version. **Bold** values indicate the best results.

	HSEmotion	Our enhanced
Accuracy ↑	79.73%	**86.03%**

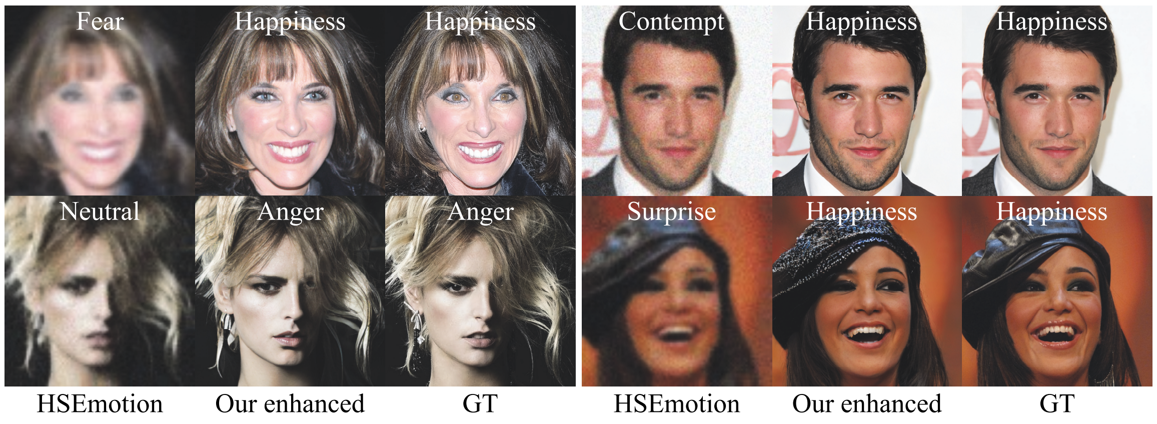


Figure 3.11: Visualization of face emotion recognition using HSEmotion and our enhanced.

3.4.2 Face emotion recognition.

By incorporating our face restoration into a face emotion recognition algorithm (HSEmotion) [157], we built our enhanced algorithm (Ours + HSEmotion). For comparison, we set the degraded images as inputs and set the emotions recognized from the ground-truth images as the ground-truth labels. The recognition accuracy (%) was evaluated.

Table 3.5 illustrates the recognition comparison between HSEmotion and our improved algorithm. While HSEmotion works well in some cases, its recognition accuracy remains capped at 79.73%. In contrast, our enhanced emotion recognition achieved a higher accuracy of 86.03%, marking a relative improvement of 6.30% over HSEmotion.

Fig. 3.11 presents additional visual comparisons for emotion recognition. These examples illustrate that even the SOTA emotion recognition method struggles with degraded images. By incorporating our facial restoration into the recognition pipeline, our enhanced algorithm is able to accurately classify the true expressions. This demonstrates the critical role of our facial restoration in improving the accuracy of facial emotion recognition.

3.5 Summary

In this chapter, we have explored a novel visual style prompt learning framework for blind face restoration. To get high-quality visual cues for guiding the restoration process, we proposed a diffusion-based style prompt module to predict visual prompts by employing DMs within the latent space of pre-trained generative models. A code diffuser was implemented to effectively denoise the given noised codes. We designed a SMART layer to utilize visual prompts and improve the extraction ability of informative contexts and detailed patterns. We build our restoration auto-encoder based on our SMART layer to achieve high-quality restoration. The extensive experiments and applications have demonstrated the effectiveness of the proposed method.

Limitations and future work. Although our method can achieve high-quality blind facial restoration, it comes with certain limitations. Like most blind facial restoration algorithms, our model may not work well on highly complex backgrounds. That might be caused by the lack of sufficiently diverse background data in the training set. Designing a more sophisticated technique to decouple the foreground and background restoration would be a promising direction. We would like to adapt our method for a wider range of downstream applications, thereby broadening its practical utility.

Chapter 4

Data-Efficient Generative Residual Image Inpainting

4.1 Introduction

Image inpainting is a fundamental task in computer graphics and computer vision [158, 159, 160, 161]. It has been employed in many downstream applications, such as image restoration [76], and image manipulation [162]. Recently proposed image inpainting methods have achieved impressive results [44, 45] on both realistic and facial images [163]. However, these methods have an overlooked limitation: they require a large amount of data to train their convolutional neural network (CNN) or transformer models [46]. When these models are trained on small image datasets, there is a high possibility of overfitting and model collapse [47]. In practice, it is much easier for users if only a small number of training images is required. Furthermore, in certain image domains (e.g., medical, art, and historical relics), large image datasets are either too expensive or infeasible to collect. This has severely restricted the use of image inpainting in real-world scenarios. Moreover, real-world data often come with their own set of challenges, including privacy concerns, data security, and data quality. These issues can significantly limit the amount of usable data. Improving the data efficiency of model training can be a crucial factor in expediting the adoption and application of image inpainting, thereby increasing its applicability to various fields that face limitations of data availability. Moreover, small-scale training samples demand less processing power and memory, which enhances the feasibility of training models in resource-limited environments.

Achieving high-quality image inpainting on small-scale datasets is still a challenging and open problem. Most existing methods [49, 45, 26] rely on single-pass inferencing which may generate ambiguous results in inpainted sub-regions. Some methods [106, 115] perform inpainting in a progressive fashion by reusing parts of previously inpainted features from early refinement stages. However, these methods do not fully utilize the inpainted pixels as useful information for the next iteration. Moreover, existing inpainting methods are not designed specifically for data-efficient learning and may not work well with a limited number of training samples. Domain-related prior knowledge [164] or lightweight generative models [165, 166] may be employed in image inpainting to mitigate overfitting. However, such approaches do not perform well when there is a large domain gap between two tasks or the reduced network capacity affects the inpainting quality.

Thus, we propose a novel data-efficient *generative residual image inpainting* framework (GRIG), which enables high-quality image inpainting on small-scale datasets. To effectively optimize inpainting results, we use iterative reasoning to more accurately and generalizably solve algorithmic reasoning tasks [55], based on residual learning [56] to incrementally refine previous estimates. By continually updating residual offsets and utilizing inpainted information from previous iterations, our model dynamically refines the input image. This approach reduces direct memorization of input-to-ground-truth mappings, effectively diminishing overfitting and enhancing visual quality. We also investigate whether combining iterative reasoning and residual learning with CNNs and transformers [46], as well as image-level and patch-level discriminators, can lead to a more robust and data-efficient method to tackle the data-efficient image inpainting task.

We have implemented our framework using three components: a generator, a projected discriminator, and a forged-patch discriminator. The generator uses CNNs to extract shallow features of edges and textures and transformer blocks to capture global interactions between feature contexts at each iterative step. To accelerate network convergence and reduce overfitting, we decouple image distribution learning by using image-level and patch-level discriminators. We first build the projected discriminator to capture the whole image-level distribution. We then use a forged-patch discriminator to enhance the patch-level details of generated images, as the projected discriminator has difficulty in capturing fine details in inpainted images. The inpainting process is carried out in several forward passes by feeding the generator with the output of the previous iteration and the corresponding mask. Experimental results on ten small-scale and four large-scale datasets show that our method is superior to state-of-the-art methods in terms of data-efficient and high-quality image inpainting.

In summary, this chapter proposes a novel data-efficient generative residual image inpainting framework with the following contributions:

- A novel algorithm for data-efficient image inpainting, which integrates CNNs and transformers, as well as employing image-level and patch-level discriminators for iterative residual reasoning.
- A forged-patch discriminator that assists the generative network to improve the fine details of generated images and prevent overfitting for data-efficient image inpainting.
- State-of-the-art performance on ten small-scale benchmark datasets with varying contents and characteristics, including facial, photorealistic, animal, medical, cartoon, and artistic images.

Iterative reasoning, which involves applying underlying computations to the outputs of previous reasoning steps repeatedly, has the potential to more accurately and generalizably solve algorithmic reasoning tasks [55], whereas residual learning [56, 167]

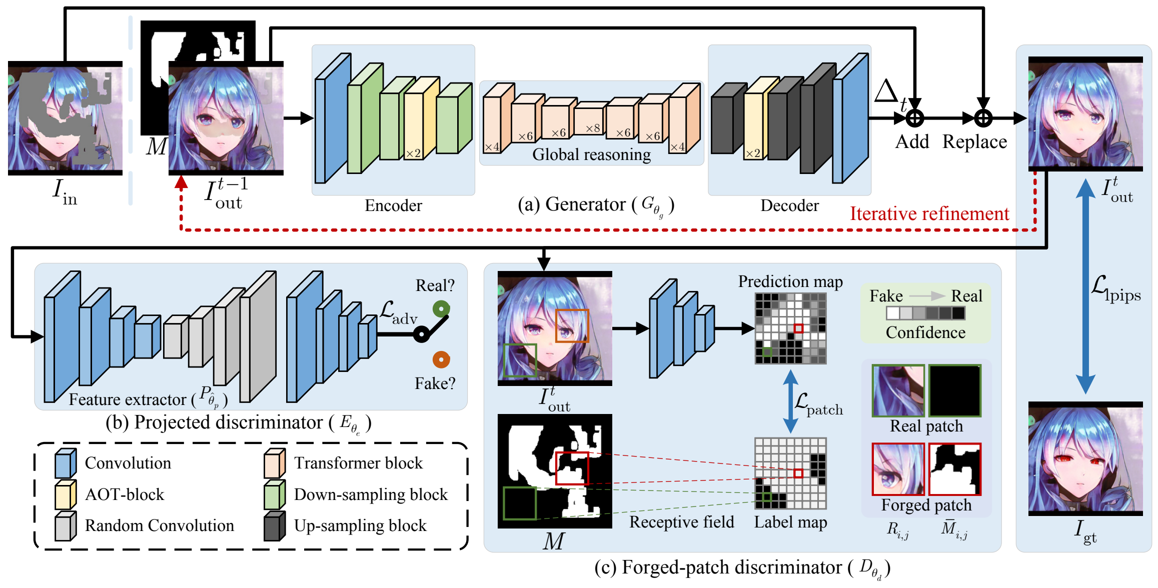


Figure 4.1: Pipeline of our data-efficient generative residual image inpainting framework (GRIG). In each t-th residual reasoning step, the generator (a) utilizes the $(t-1)$-th inpainted image I_{out}^{t-1} to generate the residual image Δ_t. In the first residual reasoning step $(t = 1)$, we set $I_{\text{out}}^{0} = I_{\text{in}}$. With the inpainted image I_{out}^{t-1} and the initial input image I_{in}, add and replace operations (refer to Eq. 4.1) are performed to obtain I_{out}^{t} for the next iterative refinement. During adversarial training, the inpainted image is fed into the projected discriminator (b) and forged-patch discriminator (c), respectively. At each iterative reasoning step, the loss functions and corresponding back-propagation are re-computed. During the test phase, a similar multi-step prediction is performed without the loss functions and back-propagation. For simplicity, down- and up-sampling operations are omitted.

facilitates the progressive optimization of previously predicted results. By iteratively predicting residual offsets and reusing previously predicted information within the

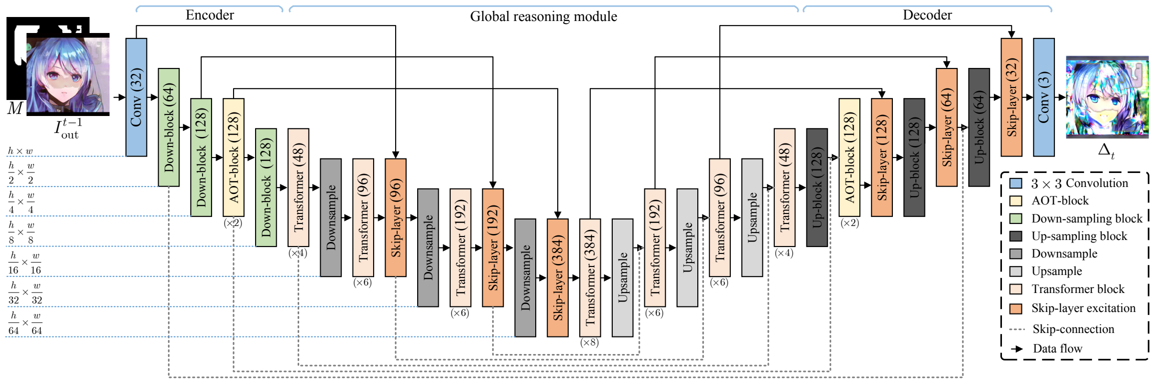


Figure 4.2: Network architecture of our generator. We show the output number of channels and dimensions for each layer/block at each scale.

same generator during each reasoning step, our GRIG can dynamically learn to refine the input image at each step. This method avoids memorizing the mapping between the inputs and their ground-truth images, thereby preventing overfitting and improving visual quality for data-efficient inpainting. In addition, to efficiently learn the global distribution of images, we utilize prior knowledge from pre-trained representations to build a compact classifier as an image-level discriminator for improving data efficiency. While this classifier excels in robust feature-based classification, it may not always capture intricate details. Recognizing this potential problem, we introduce a forged-patch discriminator that is trained to recognize real and inpainted patches based on the receptive field of the discriminator. The synergy of two discriminators mitigates the risk of overlooking fine details while also avoiding overfitting, a common challenge in data-efficient training.

As Fig. 4.1 shows, our framework consists of three main parts: a generator, a projected discriminator, and a forged-patch discriminator. Given a ground-truth image $I_{\mathrm{gt}} \in \mathbb{R}^{h\times w\times 3}$ and a binary mask $M \in \mathbb{R}^{h\times w\times 1}$ (with 1 for unknown and 0 for known pixels), the masked image $I_{\mathrm{in}} \in \mathbb{R}^{h\times w\times 3}$ is obtained as $I_{\mathrm{in}} = I_{\mathrm{gt}} \odot (1 - M)$, where $\odot$ denotes the Hadamard product. The goal of GRIG is to automatically inpaint a realistic image $I_{\mathrm{out}}^{T} \in \mathbb{R}^{h\times w\times 3}$ with T steps of iterative reasoning, where $T > 1$ denotes the iterative reasoning steps during training. For each t-th iterative residual reasoning step, a previously inpainted image I_{out}^{t-1} is fed into the generator to obtain a residual prediction. At the first residual reasoning step ($t = 1$), we set $I_{\mathrm{out}}^{0} = I_{\mathrm{in}}$. Then, addition and replacement operations are performed to produce a new image completion I_{out}^{t}. Adversarial training is conducted at each iterative step with the network weights updated accordingly via back-propagation.

4.2 Method

Iterative reasoning, which involves applying underlying computations to the outputs of previous reasoning steps repeatedly, has the potential to more accurately and generalizably solve algorithmic reasoning tasks [55], whereas residual learning [56, 167] facilitates the progressive optimization of previously predicted results. By iteratively predicting residual offsets and reusing previously predicted information within the same generator during each reasoning step, our GRIG can dynamically learn to refine the input image at each step. This method avoids memorizing the mapping between the inputs and their ground-truth images, thereby preventing overfitting and improving visual quality for data-efficient inpainting. In addition, to efficiently learn the global distribution of images, we utilize prior knowledge from pre-trained representations to build a compact classifier as an image-level discriminator for improving data efficiency. While this classifier excels in robust feature-based classification, it may not always capture intricate details. Recognizing this potential problem, we introduce a forged-patch discriminator that is trained to recognize real and inpainted patches based on the receptive field of the discriminator. The synergy of two discriminators mitigates the risk of overlooking fine details while also avoiding overfitting, a common challenge in data-efficient training.

As Fig. 4.1 shows, our framework consists of three main parts: a generator, a projected discriminator, and a forged-patch discriminator. Given a ground-truth image $I_{\text{gt}} \in \mathbb{R}^{h \times w \times 3}$ and a binary mask $M \in \mathbb{R}^{h \times w \times 1}$ (with 1 for unknown and 0 for known pixels), the masked image $I_{\text{in}} \in \mathbb{R}^{h \times w \times 3}$ is obtained as $I_{\text{in}} = I_{\text{gt}} \odot (1 - M)$, where $\odot$ denotes the Hadamard product. The goal of GRIG is to automatically inpaint a realistic image $I_{\text{out}}^{T} \in \mathbb{R}^{h \times w \times 3}$ with T steps of iterative reasoning, where $T > 1$ denotes the iterative reasoning steps during training. For each t-th iterative residual reasoning step, a previously inpainted image I_{out}^{t-1} is fed into the generator to obtain a residual prediction. At the first residual reasoning step ($t = 1$), we set $I_{\text{out}}^{0} = I_{\text{in}}$. Then, addition and replacement operations are performed to produce a new image completion I_{out}^{t}. Adversarial training is conducted at each iterative step with the network weights updated accordingly via back-propagation.

4.2.1 Network architectures

Generator. Taking the previous iteration's inpainted results as input, the generator is designed to combine CNNs and transformers [46, 168, 169] for efficient iterative residual reasoning in data-efficient image inpainting. The generator G_{θ_g} consists of an encoder, a global reasoning module with a stack of Restormer's Transformer blocks [170], and a decoder (see Fig. 4.1a). The CNN-based encoder and decoder excel at feature extraction, whereas the transformer blocks excel at dynamic attention, global context integration, and generalization. This combination helps the generator generalize effectively on small-scale training samples.

Details of our generator network are shown in Fig. 4.2. To extract features and enlarge the receptive field for capturing both informative distant image contexts and rich patterns of interest, we first stack a convolution layer, several residual down-sampling blocks [165], and AOT-blocks [104]. The extracted features are then fed into a Restormer's Transformer block stack for global context reasoning. Meanwhile, skip-layer excitation modules (SLE) [165] are utilized for a shortcut gradient flow, and skip connections are employed for collecting the multi-resolution feature maps in the decoder. The decoder is then built using up-sampling blocks [165], AOT-blocks [104], and a convolution layer. The decoder generates the intermediate prediction Δ_t by utilizing the multi-resolution feature maps output by the encoder and global reasoning module. For stable adversarial training, we apply spectral normalization [171] to all convolution layers of the networks.

At each t-th iterative reasoning step, the inpainted image from the previous iteration I_{out}^{t-1} and its corresponding mask M (see Fig. 4.1a) are fed into a generative network G_{θ_g} with the learnable network parameters θ_g. G_{θ_g} generates the intermediate residual inpainting $\Delta_t = G_{\theta_g}(I_{\text{out}}^{t-1}, M) \in \mathbb{R}^{h \times w \times 3}$. Then, the t-th inpainted image I_{out}^{t} is calculated as:

$$I_{\text{out}}^{t} = (I_{\text{out}}^{t-1} + \Delta_t) \odot M + I_{\text{in}} \odot (1 - M). \tag{4.1}$$

Projected discriminator. To stabilize GAN training and improve data efficiency, we use prior knowledge from pre-trained representations to train a compact classifier for learning the global distribution of small-scale images. The projected discriminator (see Fig. 4.1b) learns to assign high confidence scores to feature maps extracted from real images while assigning low scores to synthetic ones. Initially, feature maps are extracted from the input image I (i.e., I_{out}^{t} or I_{gt}) using a U-net-like projector $P_{\hat{\theta}_p}$ with the pre-trained network parameters $\hat{\theta}_p$. $P_{\hat{\theta}_p}$ is implemented by a pre-trained EfficientNet-Lite1 [172] with cross-channel mixing and cross-scale mixing mechanisms [166]. Subsequently, the projected discriminator E_{θ_e} with learnable network parameters θ_e maps the extracted feature maps to a scalar. Here we selected the discriminator with the largest scale of feature projections (i.e., removing the other three small-scale discriminators) from Projected GAN [166].

Forged-patch discriminator. Because the projected discriminator is primarily focused on extracting global image features for robust classification, it is possible that some fine detail features may be overlooked in these projected features. To help the

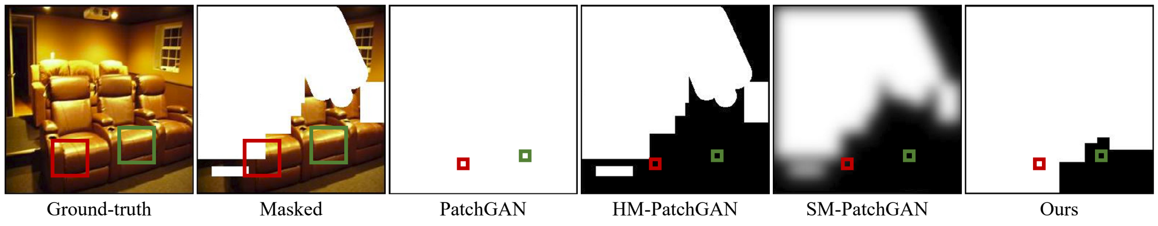


Figure 4.3: Differences between the discriminators of PatchGAN, HM-PatchGAN, SM-PatchGAN, and our algorithm. The boxes represent patches with the size of the discriminator's receptive field (left two images) and corresponding projected positions (right four images) in the resultant label maps over the (red) masked and (green) unmasked regions; Pixel values in the label maps indicate labels for fake (white) and real (black) patches.

generator produce faithful fine-grained textures and avoid overfitting in data-efficient training, we propose a forged-patch discriminator that learns to identify real and inpainted patches based on the receptive field [173] of the discriminator.

As shown in Fig. 4.1c, the forged-patch discrimination network D_{θ_d} with learnable network parameters θ_d learns to recognize real or forged image patches from a given image I (i.e., I^t_{out} or I_{gt}). The discriminator D_{θ_d} maps I to a prediction map, where each unit indicates a confidence score for each image patch based on the receptive field. In this work, we adopted the network architecture for D_{θ_d} from PatchGAN [174]. The patch-level receptive field in neural networks has also been studied as a means of overfitting avoidance in interactive video stylization [175] and improving diversity and generalizability in image generation [176].

4.2.2 Objective functions

GRIG is trained to optimize the learnable network parameters θ_g, θ_e, and θ_d using the objective functions explained below. **LPIPS loss.** At each iterative reasoning step, we use the Learned Perceptual Image Patch Similarity (LPIPS) metric [148] to constrain the perceptual similarity between the inpainted image I^t_{out} and the ground-truth image I_{gt}:

$$\begin{aligned} &\mathcal{L}_{\text{lpips}}(\theta_g) = \\ &\sum_l \frac{1}{H_l W_l} \sum_{u,v} \| w_l \odot (F_l(I^t_{\text{out}})_{u,v} - F_l(I_{\text{gt}})_{u,v}) \|_2^2 , \end{aligned} \tag{4.2}$$

where H_l and W_l represent the height and width of the feature map for layer l, respectively, u and v are spatial indices in the feature maps, w_l is the weight assigned to the feature map for layer l, $F(\cdot)$ is the pre-trained perceptual feature extractor, $F_l(\cdot)$ is the feature map for layer l; we use VGG-16 in our work [149]. This can assist our generative network in learning to maintain higher visual quality.

Projected adversarial loss. For fast convergence, the projected adversarial loss utilizes pre-trained classification models to extract prior knowledge (see Fig. 4.1b). We employ the hinge loss [166] to optimize the projected discriminator E_{θ_e} and generative network G_{θ_g}, respectively. The objective function can be formulated as:

$$\begin{aligned} \mathcal{L}^E_{\text{adv}}(\theta_e) &= \mathbb{E}_{I_{\text{gt}}}[\text{ReLu}(1 - E_{\theta_e}(P_{\hat{\theta}_p}(I_{\text{gt}})))] \\ &\quad + \mathbb{E}_{I^t_{\text{out}}}[\text{ReLu}(1 + E_{\theta_e}(P_{\hat{\theta}_p}(I^t_{\text{out}})))], \\ \mathcal{L}^G_{\text{adv}}(\theta_g) &= -\mathbb{E}_{I^t_{\text{out}}}[E_{\theta_e}(P_{\hat{\theta}_p}(I^t_{\text{out}}))]. \end{aligned} \tag{4.3}$$

The projected discriminator is constrained to assign low scores to inpainted images and high scores to real images, while the generator G_{θ_g} is supervised by the projected discriminator to inpaint the masked input based on the distribution of real images.

Adversarial forged-patch loss. As shown in Fig. 4.1c, we use the forged-patch discriminator to distinguish forged patches from real patches in a given image. We achieve this by constructing the corresponding label map $X \in \mathbb{R}^{h' \times w'}$ to supervise the discriminator. Specifically, we partition I and M into $h' \times w'$ pairs of partially overlapping patches ($R_{i,j}$ and $\overline{M}_{i,j}$) based on the receptive field of forged-patch discriminator D_{θ_d}. Here, $1 \leq i \leq h'$ and $1 \leq j \leq w'$ are horizontal and vertical indices, and the sizes of $R_{i,j}$ and $\overline{M}_{i,j}$ are equal to the receptive field $N \times N$. The label map is expressed as follows:

$$X_{i,j} = \begin{cases} 0 & \text{if } \|\overline{M}_{i,j}\|_0 = 0; \\ 1 & \text{otherwise,} \end{cases} \tag{4.4}$$

where $\|\overline{M}_{i,j}\|_0$ is defined as the L0 norm of the sub-region mask $\overline{M}_{i,j}$. If $\|\overline{M}_{i,j}\|_0$ is not zero, it indicates that there are some masked pixels in this sub-region mask, and the image patch $R_{i,j}$ contains inpainted pixels. Thus, we set $X_{i,j} = 1$, which means that the sub-region $R_{i,j}$ is a forged patch. Otherwise, it is a real patch. The hinge version of adversarial forged-patch loss is expressed as:

$$\begin{aligned}\mathcal{L}^{D}_{\text{patch}}(\theta_d) = {} & \mathbb{E}_{I_{\text{gt}}}[\text{ReLu}(1 - D_{\theta_d}(I_{\text{gt}}))] \\ & + \mathbb{E}_{I^t_{\text{out}}}[\text{ReLu}(1 - D_{\theta_d}(I^t_{\text{out}})) \odot (1 - X)] \\ & + \mathbb{E}_{I^t_{\text{out}}}[\text{ReLu}(1 + D_{\theta_d}(I^t_{\text{out}})) \odot X], \\ \mathcal{L}^{G}_{\text{patch}}(\theta_g) = {} & -\mathbb{E}_{I^t_{\text{out}}}[D_{\theta_d}(I^t_{\text{out}}) \odot X].\end{aligned} \tag{4.5}$$

Fig. 4.3 illustrates the differences between the proposed forged-patch discriminator and other closely related discriminators. PatchGAN's discriminator [174] directly assigns all patches in inpainted images as fake patches, which can confuse the discriminator when extracted patches do not have any generated pixel. HM-PatchGAN and SM-PatchGAN [104] aim to segment synthesized patches of missing regions according to inpainting masks. Since the inpainting masks have to be downsampled first to agree with the spatial size of the discriminator's output, the constraints around the mask boundaries may be unclear. For example, downsampling inpainting masks results in information loss of the precise location of inpainted pixels. SM-PatchGAN tries to identify the generated and real patches, whereas our discriminator goes one step further to consider whether generated pixels are consistent with surrounding real pixels in a given patch. Our discriminator constructs the label map based on the receptive field and treats all patches with any inpainted pixels as fake patches, which gives more constraints than PatchGAN and SM-PatchGAN.

Total objective. The total training objective of the generator is expressed as:

$$\mathcal{L}^{G}_{\text{total}} = \lambda_{\text{lpips}}\mathcal{L}_{\text{lpips}} + \lambda_{\text{adv}}\mathcal{L}^{G}_{\text{adv}} + \lambda_{\text{patch}}\mathcal{L}^{G}_{\text{patch}}, \tag{4.6}$$

Algorithm 1 GRIG training procedure

```
1: while $G_{\theta_g}$, $E_{\theta_e}$, and $D_{\theta_d}$ have not converged do
2:     Sample batch images $\mathcal{I}_{\text{gt}}$ from the training set
3:     Create random masks $\mathcal{M}$ for $\mathcal{I}_{\text{in}}$
4:     Get inputs $\mathcal{I}_{\text{in}} \leftarrow \mathcal{I}_{\text{gt}} \odot (1 - \mathcal{M})$
5:     Set inputs $\mathcal{I}_{\text{out}}^{0} \leftarrow \mathcal{I}_{\text{in}}$
6:     for iterative residual reasoning step $t = 1$ to $T$ do
7:         Get $\Delta_t \leftarrow G_{\theta_g}(\mathcal{I}_{\text{out}}^{t-1}, \mathcal{M})$
8:         Get $\mathcal{I}_{\text{out}}^{t} \leftarrow (I_{\text{out}}^{t-1} + \Delta_t) \odot \mathcal{M} + \mathcal{I}_{\text{in}} \odot (1 - \mathcal{M})$
9:         Update $G_{\theta_g}$ with $\mathcal{L}_{\text{total}}^{G}$
10:        Update $E_{\theta_e}$ with $\mathcal{L}_{\text{adv}}^{E}$
11:        Update $D_{\theta_d}$ with $\mathcal{L}_{\text{patch}}^{D}$
```

where λ_{lpips}, λ_{adv}, and λ_{patch} weight corresponding losses. During training, we alternately optimize parameters θ_g, θ_e, and θ_d.

4.2.3 Iterative residual reasoning

The iterative residual reasoning for image inpainting can be formulated as an optimization process over adversarial generative networks. This enables the generator to implicitly learn to leverage previously predicted results and focus on residual information in order to achieve high quality and better generality.

We introduce a generative network $G_{\theta_g}(I_{\text{out}}^{t-1})$ as an explicit function to predict

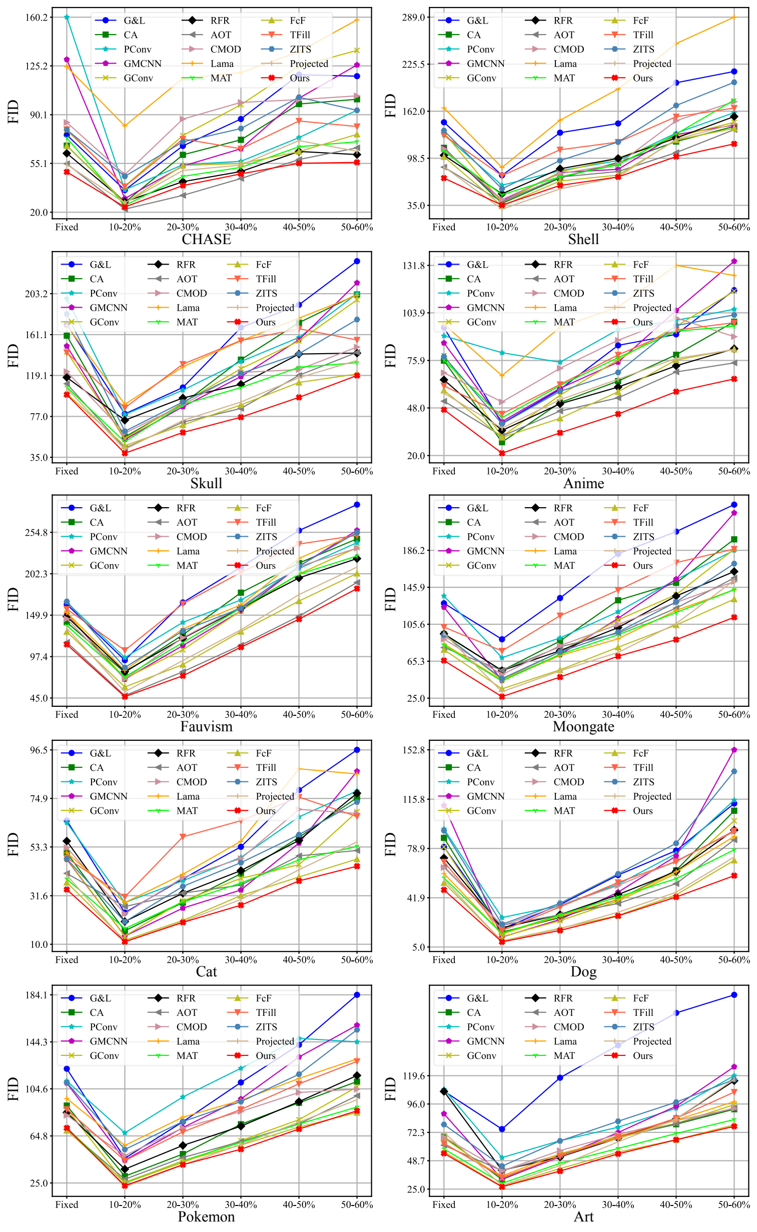


Figure 4.4: Quantitative comparisons of GRIG with various state of the art image inpainting methods on ten small-scale datasets, using FID evaluation metric. In each graph, the horizontal axis indicates mask ratios; 'Fixed' denotes the fixed center 25% rectangular mask.

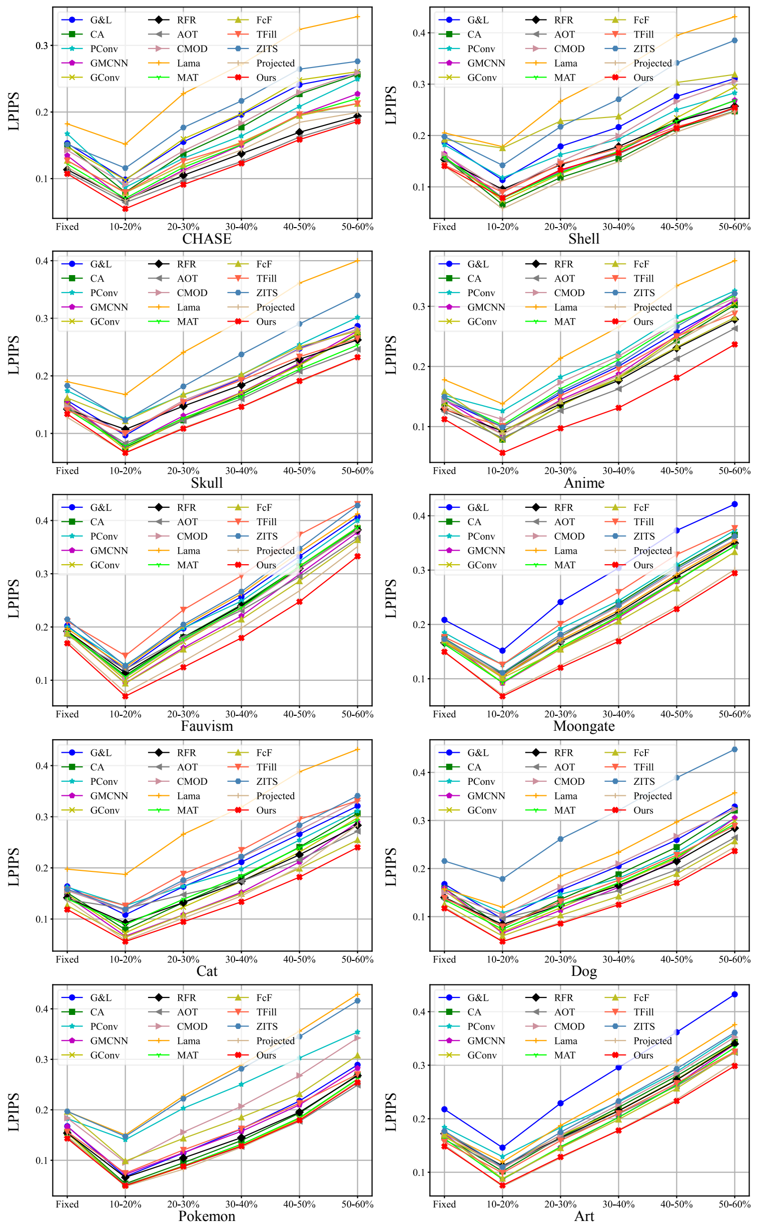


Figure 4.5: Quantitative comparisons of GRIG with various state of the art image inpainting methods on ten small-scale datasets, using LPIPS evaluation metric. In each graph, the horizontal axis indicates mask ratios; 'Fixed' denotes the fixed center 25% rectangular mask.

residual information (see Eq. 4.1). At each iterative reasoning step t, the generator G_{θ_g} is trained to maximize the confidence values of $E_{\theta_e}(I^t_{\text{out}})$ and values in the prediction map of $D_{\theta_d}(I^t_{\text{out}})$ while minimizing the perceptual similarity difference between I^t_{out} and I_{gt}. Thus, θ_g is solved by $\theta_g = \arg\min_{\theta_g} \mathcal{L}^G_{\text{total}}$. Simultaneously, E_{θ_e} and D_{θ_d} are trained to distinguish images (fake or real) and patches, respectively, where $\theta_e = \arg\min_{\theta_e} \mathcal{L}^E_{\text{adv}}$ and $\theta_d = \arg\min_{\theta_d} \mathcal{L}^D_{\text{patch}}$. The generative network G_{θ_g} directly predicts the residual information while its parameters θ_g are supervised by the discriminators as well as I_{gt}. Pseudocode for the GRIG training procedure is given in Algorithm 1.

4.3 Experimental results and comparisons

4.3.1 Experimental setting

Python and PyTorch were used to build the proposed framework. We set $\lambda_{\text{lpips}} = 1.5$, $\lambda_{\text{adv}} = 1$, $\lambda_{\text{patch}} = 1$, and $T = 3$ for all experiments in both training and testing phases, unless otherwise specified. We used the Adam optimizer with first momentum coefficient $\beta_1 = 0.5$, second momentum coefficient $\beta_2 = 0.999$, and learning rate 0.0002. Our masks were created with the CMOD mask generation algorithm [49]. Our generator contains 31.76M parameters and achieves around 21 FPS for each residual reasoning step on an NVIDIA GeForce RTX 2080 GPU with 8 GB memory.

We compared GRIG to various state of the art image inpainting methods: Globally&Locally (G&L) [63], Contextual Attention (CA) [100], Partial Convolutions

(PConv) [101], GMCNN [93], Gated Convolution (GConv) [26], Recurrent Feature Reasoning (RFR) [115], AOT-GAN (AOT) [104], Co-mod-GAN (CMOD) [49], Lama [44], MAT [45], FcF [177], TFill [105], and ZITS [178]. We also compared GRIG to an inpainting model (Projected) based on the light-weight Projected GAN [166] to further demonstrate the superiority of GRIG for data-efficient image inpainting. The publicly available MMEditing framework [179], an open-source image and video editing toolbox based on PyTorch, implements the models of G&L, CA, PConv, and GConv. We used the authors' codes for GMCNN, RFR, AOT, Lama, MAT, FcF, TFill, and ZITS. We used the authors' TensorFlow-based code to create a PyTorch-based version of CMOD. To implement the Projected model, we added a mirrored encoder of Projected GAN with skip connections and perceptual similarity $\mathcal{L}_{\text{lpips}}$. This Projected model was created using PyTorch with the same hyper-parameters as GRIG with $\lambda_{\text{lpips}} = 1.5$.

Experiments were conducted on ten small-scale datasets (CHASE [180], Shell [165], Skull [165], Anime [165], Fauvism [165], Moongate [165], Cat [181], Dog [181], Pokemon (pokemon.com), and Art (wikiart.org)) and four large-scale image datasets (including CelebA-HQ [154], FFHQ [42], Paris Street View (PSV) [182], and Places365 [183]). Details of the sizes of the datasets are given in Table 4.1. We used the original training and testing splits from the PSV and Places365 datasets, while other datasets were split using random sampling. To ensure fairness, we used the same training/testing splits for all experiments.

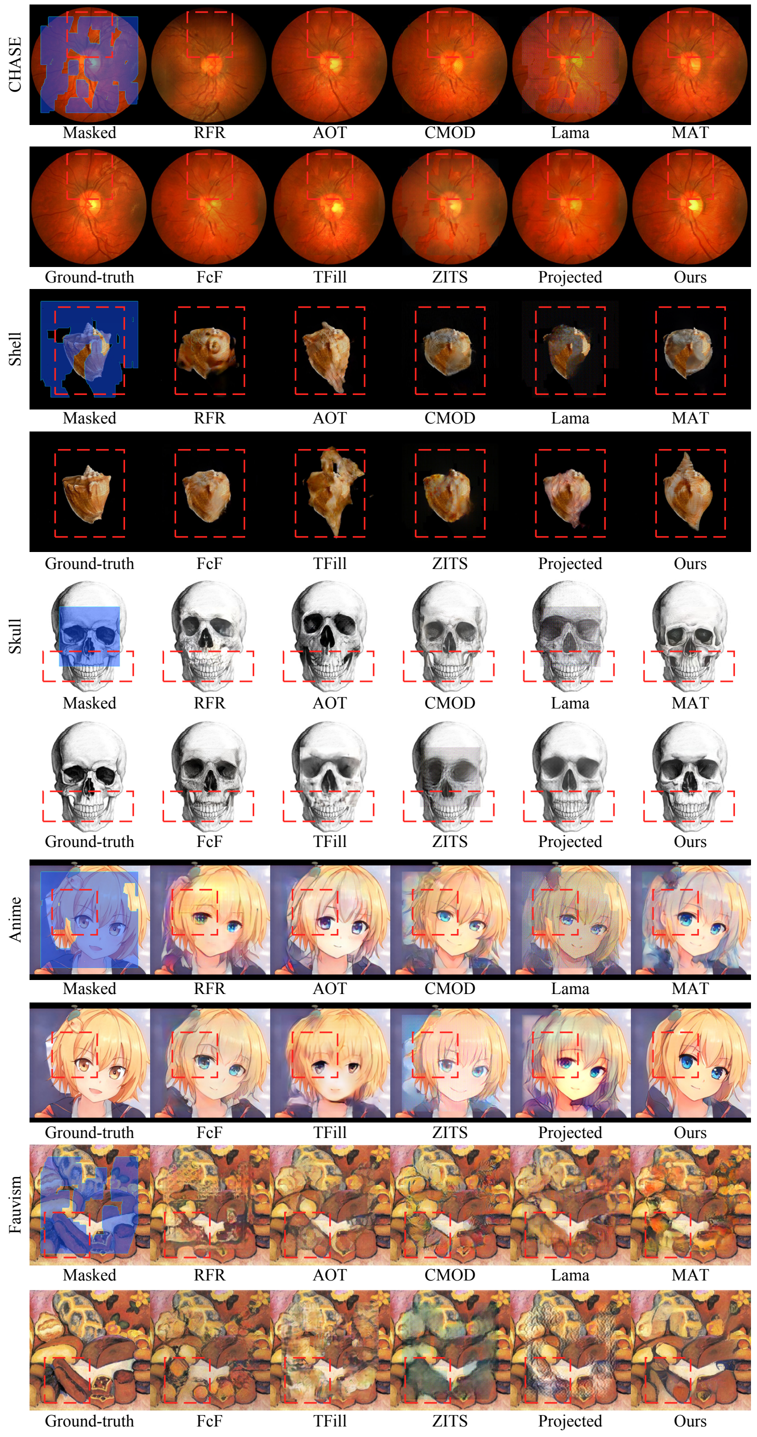


Figure 4.6: Results of GRIG and other state of the art image inpainting methods on small-scale datasets (CHASE, Shell, Skull, Anime, Fauvism).

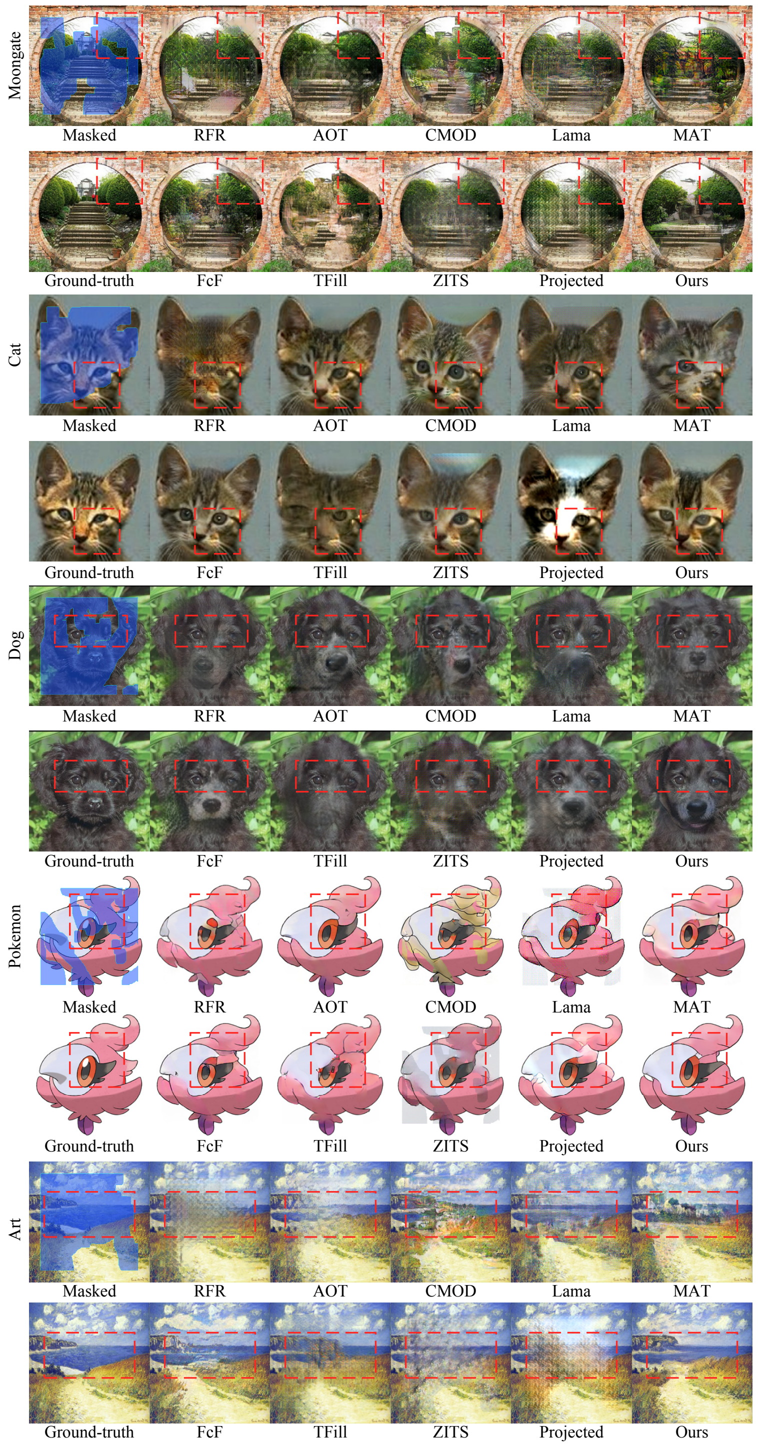


Figure 4.7: Results of GRIG and other state of the art image inpainting methods on small-scale datasets (Moongate, Cat, Dog, Pokemon, Art).

Table 4.1: Details of the ten small-scale and four large-scale image datasets.

Type	Dataset	# Training set	# Test set
Small-scale	CHASE	18	10
	Shell	48	16
	Skull	72	25
	Anime	90	30
	Fauvism	94	30
	Moongate	106	30
	Cat	120	40
	Dog	309	80
	Pokemon	633	200
	Art	750	250
Large-scale	CelebA-HQ	28K	2K
	FFHQ	60K	10K
	PSV	14.9K	100
	Places365	1.8M	36.5K

All images were resized to a resolution of 256×256. All compared models were retrained on the datasets mentioned in the chapter, using a batch size of 8, unless otherwise noted. During testing, various irregular masks with different mask ratios [101] and a fixed center 25% (128×128) rectangular mask were used to simulate different situations for all experiments. All methods in our evaluation replaced the unmasked known regions with the original image. All models were trained and tested on NVIDIA V100 GPUs (with 32 GB memory).

Since L1 distance, PSNR, and SSIM all heavily prefer blurry results [49], we used Fréchet inception distance (FID) [155] and LPIPS metrics for quantitative evaluation following established practice in recent literature [44].

4.3.2 Comparison on small-scale datasets

To evaluate the performance on ten small-scale datasets (see Table 4.1), all models were trained with $400,000$ image batches. We implemented the early-stopping technique for each method, ensuring that each model is adequately trained without overfitting and achieved optimal performance for evaluation. Fig. 4.4 and Fig. 4.5 quantitatively compare GRIG to the other state of the art inpainting methods on the ten small-scale datasets. To compare the performance of image inpainting, various irregular masks with different mask ratios [101], as well as a fixed center 25% (128×128) rectangular mask, were used to simulate various scenarios. In the small-scale setting, differentiable data augmentation [184] was applied for all compared methods when sampling images in the training phase. As Fig. 4.4 and Fig. 4.5 show, GRIG outperforms all baselines in terms of FID and LPIPS metrics by large margins on most benchmarks for various kinds of masks. For most datasets, significant gains were obtained by our method. Notably, for a 50-60% mask ratio, GRIG achieves a relative improvement of FID to the second-best methods of 9.12% (CHASE), 13.98% (Shell), 1.26% (Skull), 12.65% (Anime), 4.06% (Fauvism), 14.86% (Moongate), 6.64% (Cat), 16.40% (Dog), and 1.60% (Art).

Fig. 4.6 and Fig. 4.7 present inpainted results for the compared methods. It reveals that most methods fail to produce plausible contents for datasets with fewer than 100 training samples (for example, CHASE, Shell, Skull, and Anime) due to overfitting to features and patterns from a small number of samples. When trained on datasets with more than 500 samples (such as Pokemon and Art), some methods may be able to fill more semantic content within masked areas. However, artifacts can still be seen under close inspection. When the masked area is large, RFR is prone to producing repetitive image patches in inpainted regions, and while AOT, Lama, and ZITS can inpaint structures in missing regions, they leave artifacts in fine detail. We also noticed that Lama and ZITS have similar blurring phenomena in the inpainted regions when trained on small-scale data, which may be because the Fast Fourier Convolution [185] (FFC) overfits the limited global repeating patterns [177], harming subsequent feature extraction. CMOD, MAT, and TFill tend to overfit the training data due to their large numbers of learnable parameters. Projected GANs can handle the semantic structure, but they may introduce color inconsistency around mask boundaries. By combining the benefits of FFC and stochasticity, FcF shows robust performance on both textural and structural image inpainting. Fig. 4.4, Fig. 4.5, Fig. 4.6, and Fig. 4.7 demonstrate that GRIG can achieve better performance on

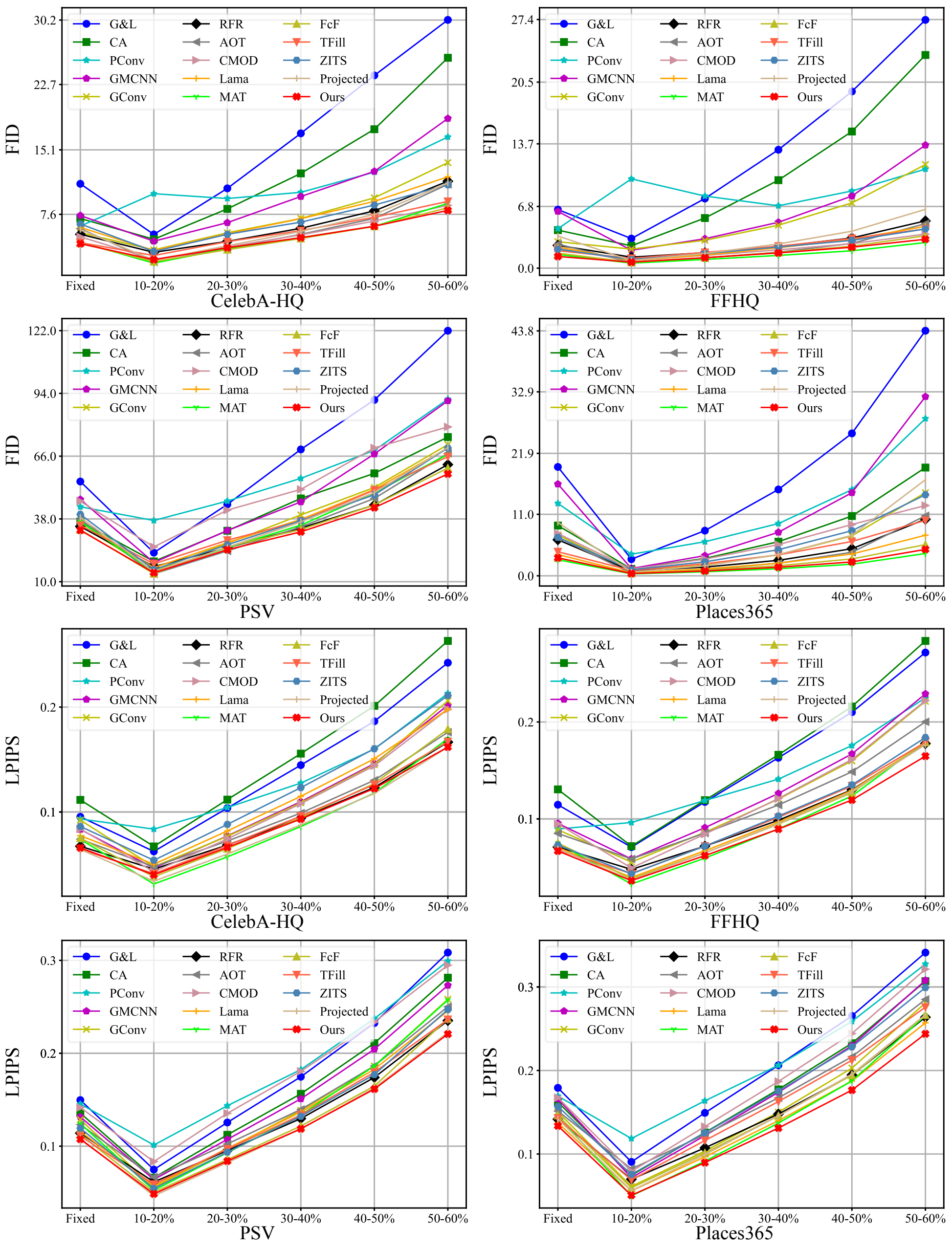


Figure 4.8: Quantitative comparisons of GRIG with various state of the art image inpainting methods on four large-scale datasets, for FID and LPIPS evaluation metrics. In each graph, the horizontal axis indicates mask ratios; 'Fixed' denotes the fixed center 25% rectangular mask.

quantitative metrics and visual quality, even though our method has more learnable parameters (31.76M) than those of GConv (4.0M) and is trained on limited samples. GRIG demonstrates strong generalization capabilities in various small-scale datasets with differing numbers of training samples, and produces images with higher visual quality.

We believe that there are three reasons for the better generalization performance and inpainting quality achieved on data-efficient image inpainting. Firstly, our iterative residual reasoning strategy enables the generator to use information learned in previous iterations while also increasing the diversity of inputs to improve performance. Secondly, the self-attention mechanism in Transformers [46] has advantages in leveraging existing information for further context reasoning. In our generator, the encoder and decoder are used to extract local features, while the Restormer Transformer blocks [170] are used for global context reasoning. Thirdly, the projected discriminator and forged-patch discriminator, with 2.829M and 2.765M learnable parameters, respectively, help improve the generality of our method. The projected discriminator focuses on images at the semantic level based on the generality of pre-trained features. The forged-patch discriminator focuses on learning patch-level consistency to capture patch statistics and distinguishing between real and inpainted patches to prevent overfitting by avoiding the need to memorize the entire image.

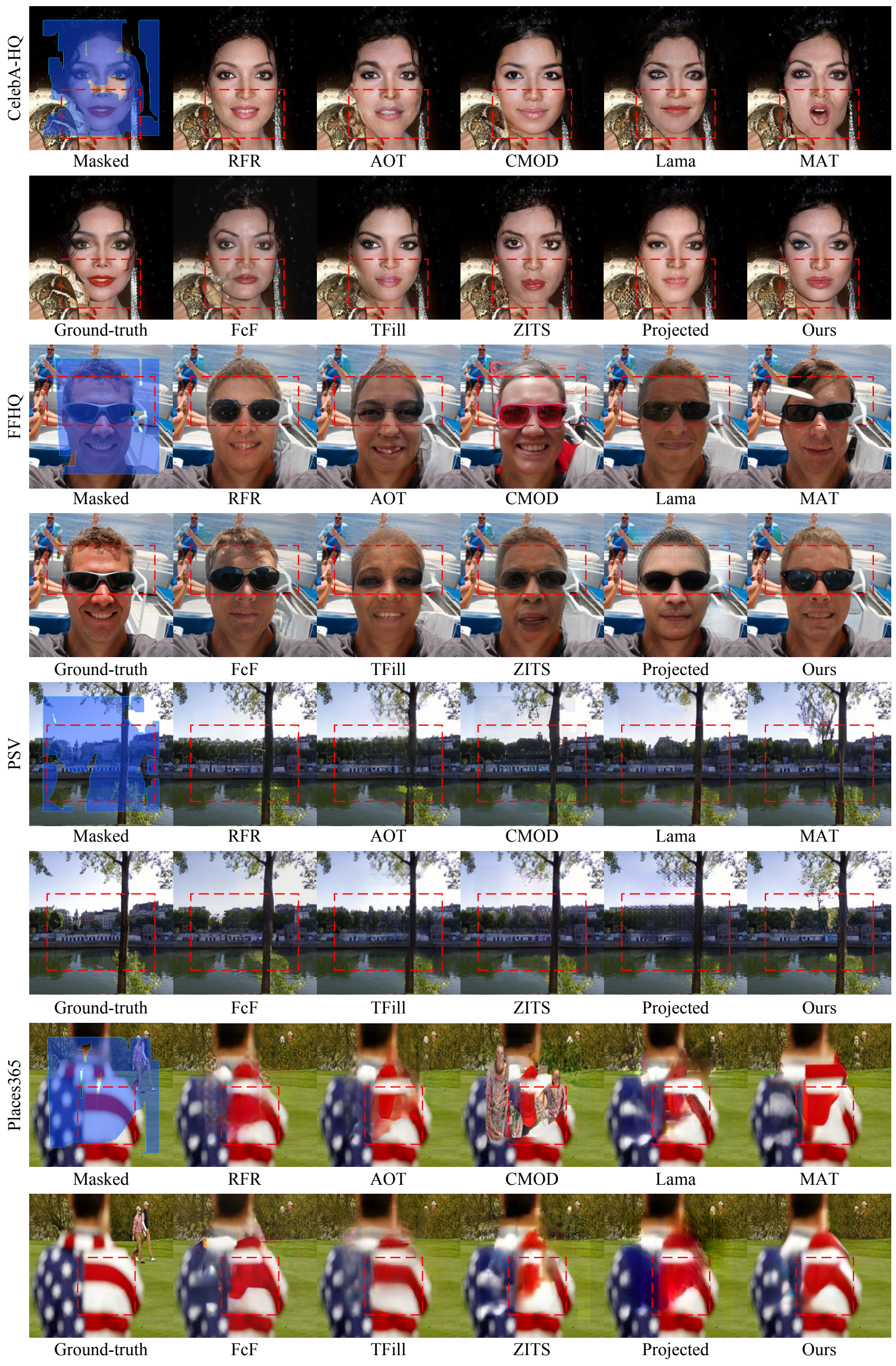


Figure 4.9: Visual comparison of GRIG and state of the art image inpainting methods on large-scale datasets.

4.3.3 Comparison on large-scale datasets

We also compared our method to the same inpainting methods on four large-scale datasets. All methods were trained with their default settings to ensure fair comparisons. Our model was trained with $1,000,000$ image batches on CelebA-HQ, FFHQ, and PSV, respectively, and $2,000,000$ image batches on Places365.

The quantitative results in Fig. 4.8 show that GRIG outperforms the majority of the SOTA inpainting methods in terms of FID and LPIPS metrics on large-scale datasets. In particular, GRIG achieves the best FID scores on PSV, and the best LPIPS scores on PSV and Places365. MAT has the best FID scores on FFHQ and Places365. Overall, GRIG performs comparably to MAT on the other large-scale datasets while containing many fewer learnable weights (31.76M) than MAT (62.0M). Our iterative residual learning effectively assists the networks in decomposing the inpainting process into multiple reasoning steps with the progressive refinement of inpainting results. Moreover, the decoupling of image distribution learning into image-level and patch-level constraints with our projected discriminator and forged-patch discriminator helps our GRIG model achieve excellent performance in both data-efficient scenarios and large datasets.

Fig. 4.9 shows a corresponding qualitative performance evaluation. It demonstrates that semantic inpainting on large masks remains difficult for most inpainting methods. RFR produces repetitive image patches in inpainted regions because iterative refinement in feature space may overlook fine details in image space. AOT

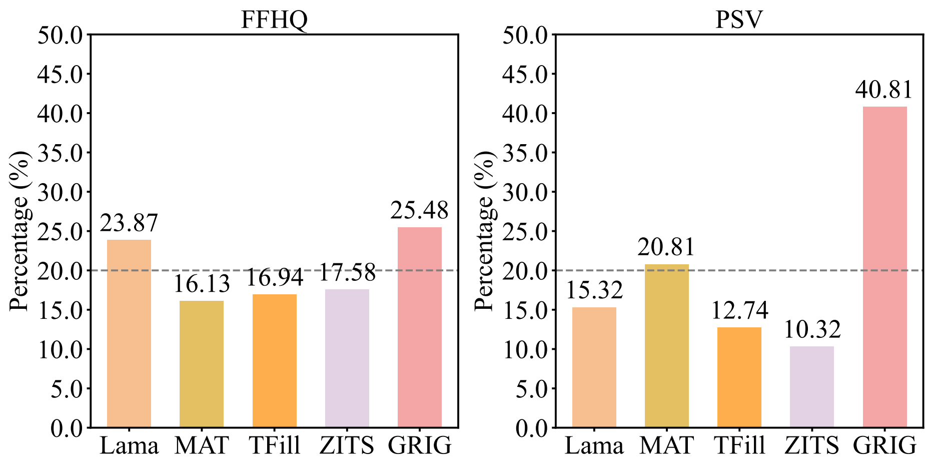


Figure 4.10: User study results on the FFHQ and PSV datasets using state of the art methods (Lama, MAT, TFill, and ZITS). We give percentages of cases in which each method is ranked first over others.

and CMOD perform well on these datasets. However, with complex backgrounds, they struggle with larger masked areas in some cases. MAT and FcF handle texture and structure inpainting well and generalize well to different types of datasets. Because one-time inferencing cannot re-adjust inpainted results, complex backgrounds are likely to have negative impacts on MAT and FcF inpainting quality. With their multi-stage inpainting processes, TFill and ZITS utilize Transformer architectures to notably enhance the visual quality of inpainted pixels. However, their performance may be influenced when previous networks in the process do not perform optimally. Because fine details are easily overlooked in projected features, projection-based models [166] tend to produce blurred results. Our GRIG can inpaint plausible contents in complex structures with high mask ratios.

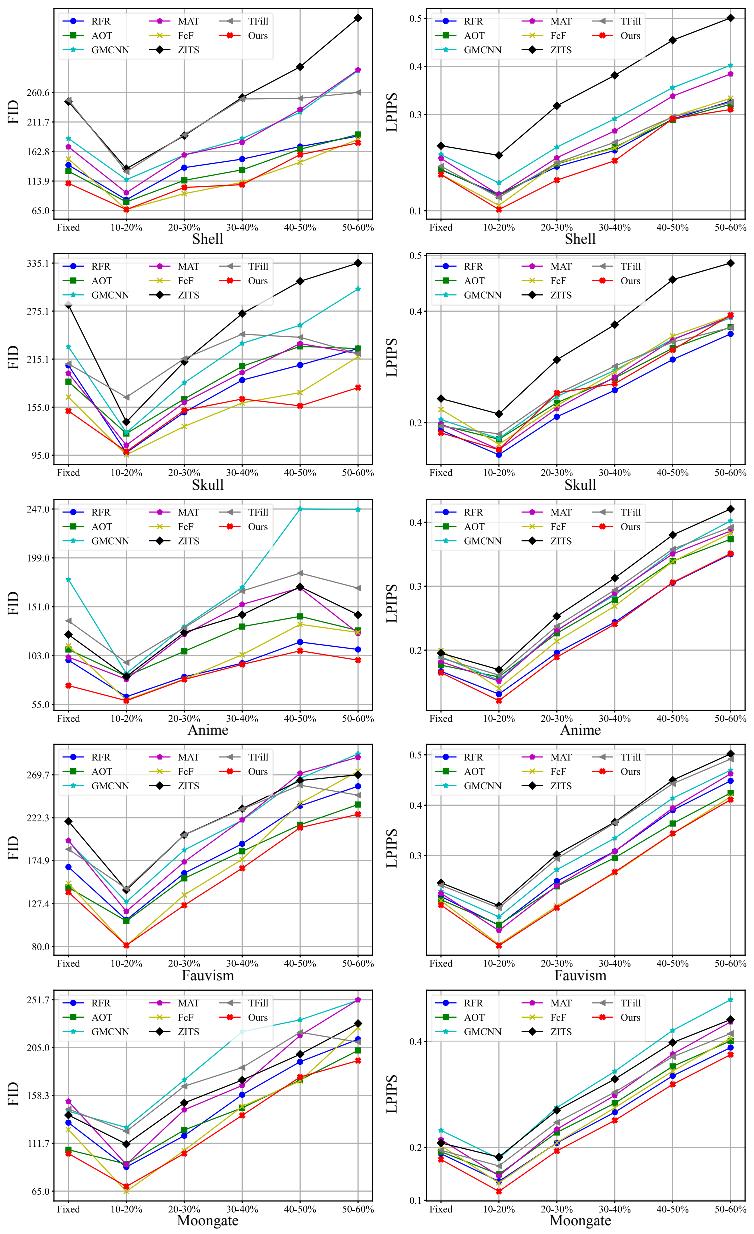


Figure 4.11: Quantitative comparisons between models (RFR, AOT, GMCNN, MAT, FcF, TFill, ZITS, and ours) trained on the 5-shot setting of Shell, Skull, Anime, Fauvism, and Moongate datasets. In each graph, the horizontal axis indicates mask ratios; 'Fixed' denotes the fixed center 25% rectangular mask.

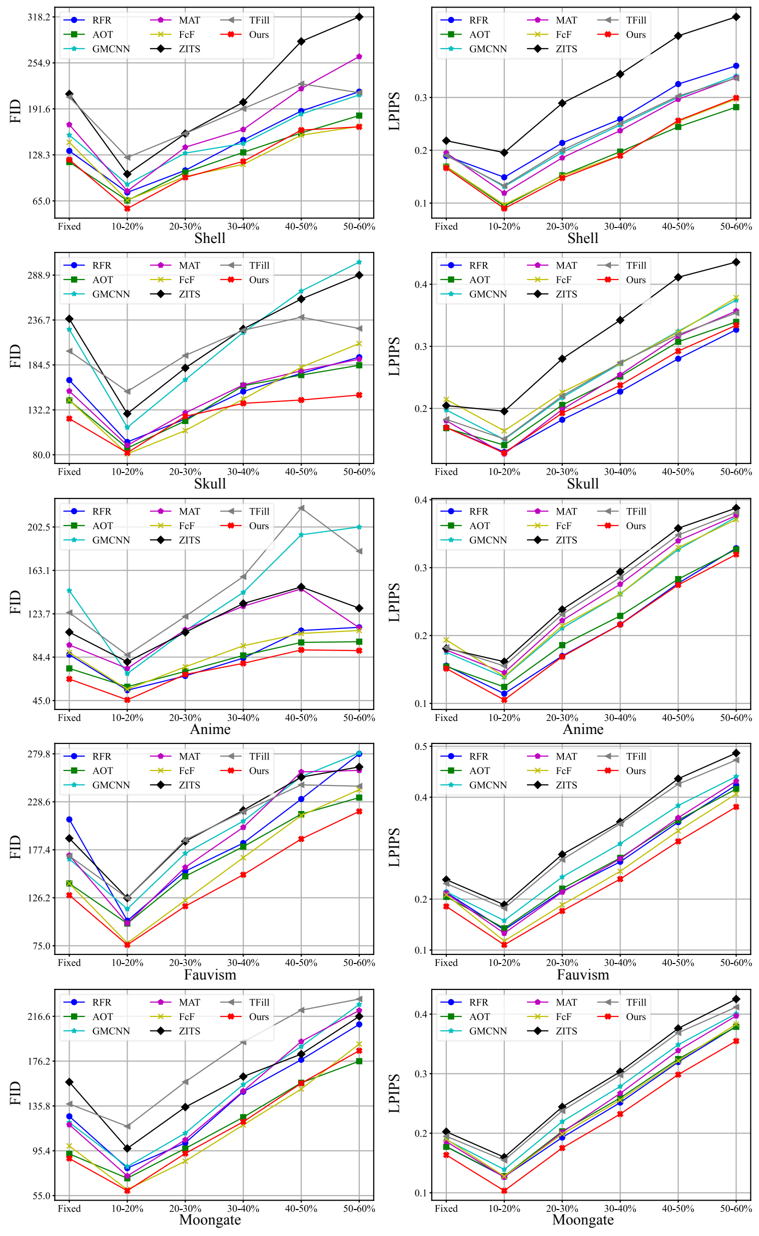


Figure 4.12: Quantitative comparisons between models (RFR, AOT, GMCNN, MAT, FcF, TFill, ZITS, and ours) trained on the 10-shot setting of Shell, Skull, Anime, Fauvism, and Moongate datasets. In each graph, the horizontal axis indicates mask ratios; 'Fixed' denotes the fixed center 25% rectangular mask.

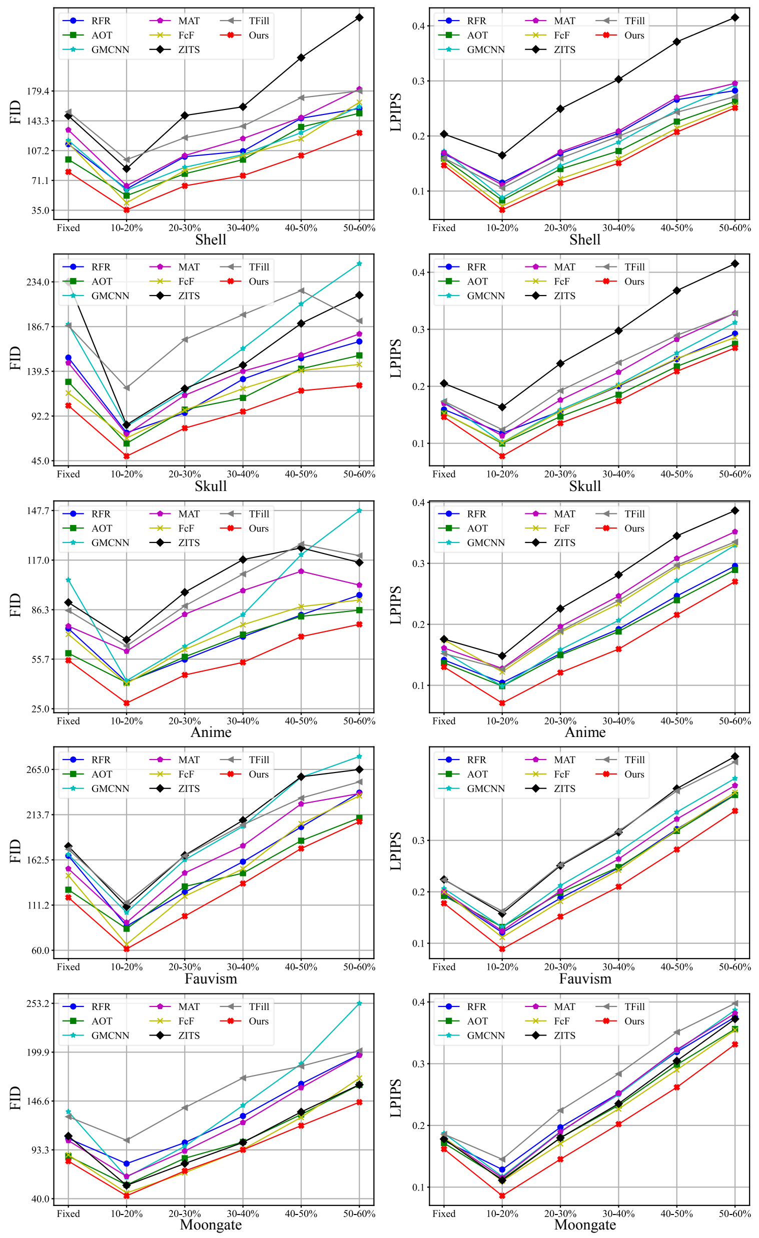


Figure 4.13: Quantitative comparisons between models (RFR, AOT, GMCNN, MAT, FcF, TFill, ZITS, and ours) trained on the 30-shot setting of Shell, Skull, Anime, Fauvism, and Moongate datasets. In each graph, the horizontal axis indicates mask ratios; 'Fixed' denotes the fixed center 25% rectangular mask.

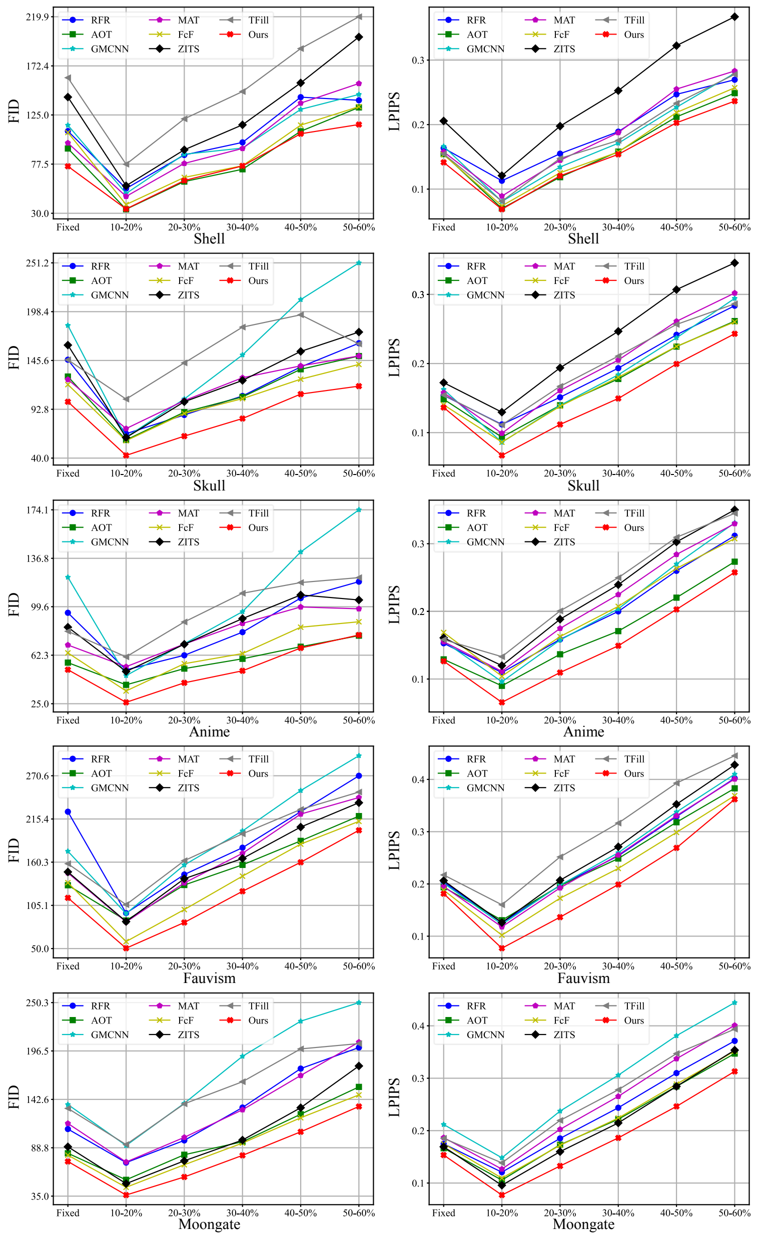


Figure 4.14: Quantitative comparisons between models (RFR, AOT, GMCNN, MAT, FcF, TFill, ZITS, and ours) trained on the 50-shot setting of Shell, Skull, Anime, Fauvism, and Moongate datasets. In each graph, the horizontal axis indicates mask ratios; 'Fixed' denotes the fixed center 25% rectangular mask.

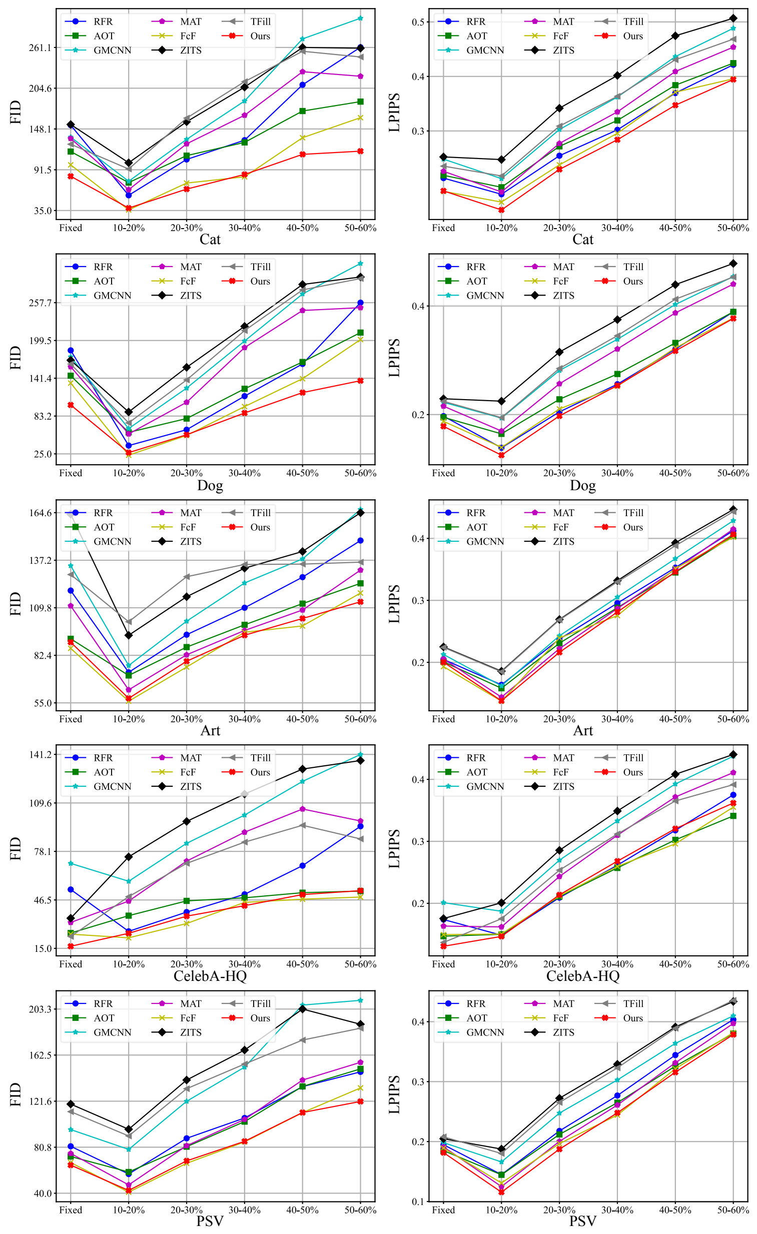


Figure 4.15: Quantitative comparisons between models (RFR, AOT, GMCNN, MAT, FcF, TFill, ZITS, and ours) trained on the 5-shot setting for Cat, Dog, Art, CelebA-HQ, and PSV datasets. In each graph, the horizontal axis indicates mask ratios; 'Fixed' denotes the fixed center 25% rectangular mask.

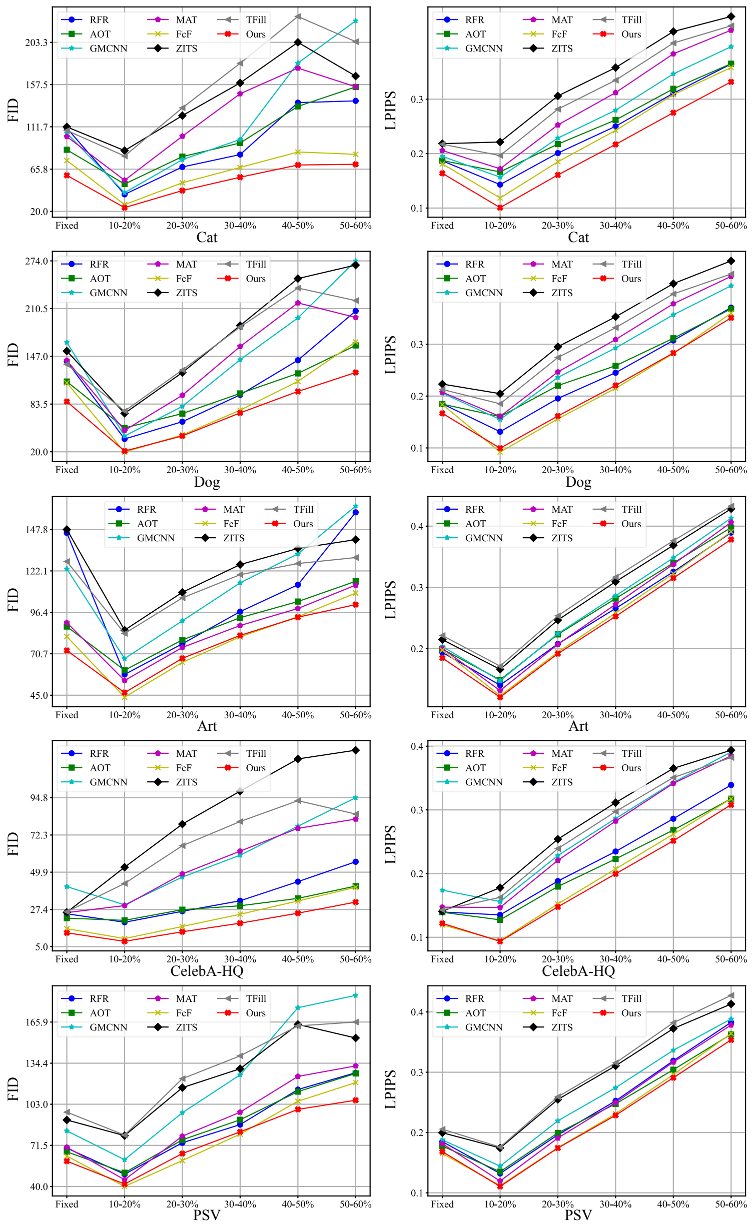


Figure 4.16: Quantitative comparisons between models (RFR, AOT, GMCNN, MAT, FcF, TFill, ZITS, and ours) trained on the 10-shot setting for Cat, Dog, Art, CelebA-HQ, and PSV datasets. In each graph, the horizontal axis indicates mask ratios; ‘Fixed’ denotes the fixed center 25% rectangular mask.

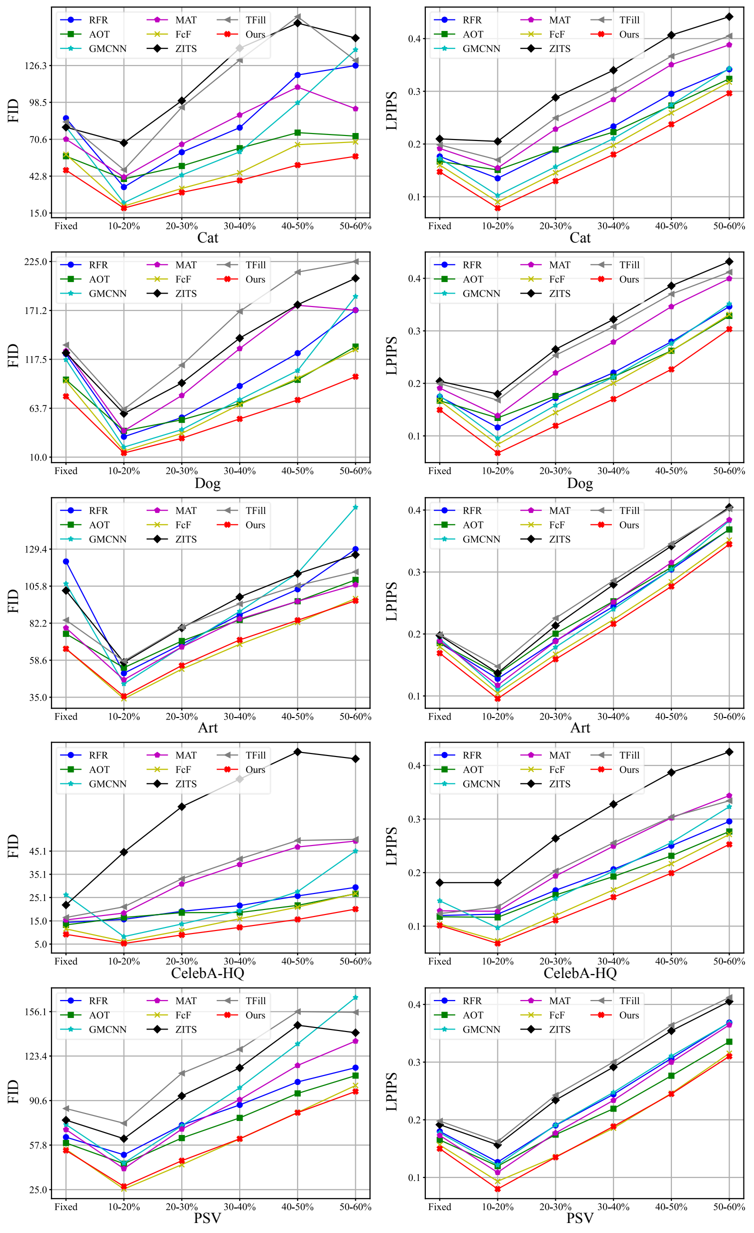


Figure 4.17: Quantitative comparisons between models (RFR, AOT, GMCNN, MAT, FcF, TFill, ZITS, and ours) trained on the 30-shot setting for Cat, Dog, Art, CelebA-HQ, and PSV datasets. In each graph, the horizontal axis indicates mask ratios; ‘Fixed’ denotes the fixed center 25% rectangular mask.

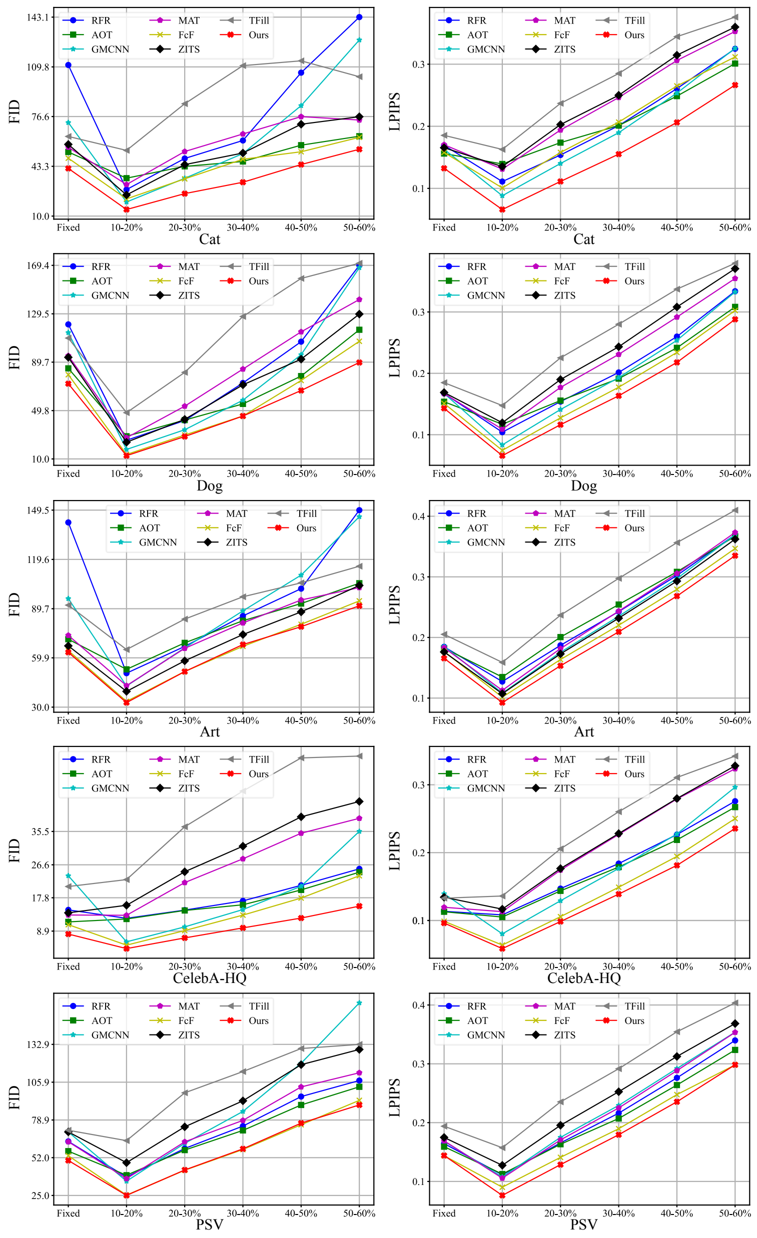


Figure 4.18: Quantitative comparisons between models (RFR, AOT, GMCNN, MAT, FcF, TFill, ZITS, and ours) trained on the 50-shot setting for Cat, Dog, Art, CelebA-HQ, and PSV datasets. In each graph, the horizontal axis indicates mask ratios; 'Fixed' denotes the fixed center 25% rectangular mask.

Table 4.2: User study results: average rankings of compared methods on the FFHQ and PSV datasets. **Bold** indicates best results.

Dataset	Lama	MAT	TFill	ZITS	GRIG
FFHQ	2.88	3.16	3.10	2.99	**2.87**
PSV	2.97	3.13	3.25	3.39	**2.26**

We conducted a user study using various state of the art methods (Lama, MAT, TFill, and ZITS) on the FFHQ and PSV datasets to demonstrate GRIG's inpainting performance on large-scale datasets. For each dataset, we randomly sampled 100 images from the testing set, then randomly selected and assigned 20 of those images to each participant. Each question contained a masked image, a ground-truth image, and shuffled inpainted images from the five compared methods. The users were asked to rank the compared methods based on visual quality and realism. We recruited 31 participants, totaling 620 votes for each method on each dataset.

Fig. 4.10 shows the percentage of time each method achieved the top rank on the FFHQ and PSV datasets. Our GRIG had the highest percentages at 25.48% on FFHQ and 40.81% on PSV. Table 4.2 displays the average rankings for each compared method. All average rankings are within the range of [2.0, 3.5], indicating comparable performance for these methods. Notably, our GRIG had the best average rankings on FFHQ and PSV, of 2.87 and 2.26 respectively. The user study results show that our GRIG produces high-quality image inpainting results.

Table 4.3: Comparisons of mean FID and LPIPS scores across all mask ratios and few-shot settings for each dataset. **Bold** indicates best results.

Metrics	Methods	Shell	Skull	Anime	Fauvism	Moongate	Cat	Dog	Art	CelebA-HQ	PSV
FID	RFR	127.31	144.71	83.72	183.50	138.11	104.10	109.63	103.29	31.92	86.17
	AOT	112.06	140.73	80.66	160.93	115.87	85.38	95.46	87.04	26.99	82.93
	GMCNN	139.87	192.34	130.63	201.42	164.46	111.20	128.53	106.64	49.19	113.25
	MAT	146.67	147.96	101.18	185.71	142.61	104.36	129.27	85.53	46.63	88.87
	FcF	111.09	130.80	81.87	157.42	110.28	63.21	82.13	76.01	23.65	73.67
	TFill	176.16	190.69	122.77	196.17	149.54	134.01	156.56	106.38	52.08	122.49
	ZITS	190.74	193.72	108.92	197.22	134.08	125.43	147.07	106.96	68.22	117.37
	Ours	**100.12**	**114.77**	**66.49**	**140.03**	**105.21**	**53.45**	**68.02**	**74.83**	**19.27**	**69.17**
LPIPS	RFR	0.2163	0.2122	0.2074	0.2652	0.2426	0.2413	0.2279	0.2508	0.2093	0.2391
	AOT	0.1875	0.2141	0.2074	0.2610	0.2385	0.2449	0.2305	0.2547	0.1992	0.2274
	GMCNN	0.2191	0.2292	0.2307	0.2824	0.2785	0.2530	0.2496	0.2550	0.2344	0.2492
	MAT	0.2190	0.2296	0.2416	0.2683	0.2576	0.2766	0.2677	0.2509	0.2402	0.2338
	FcF	0.2842	0.2442	0.2337	0.2448	0.2374	0.2188	0.2061	0.2334	0.1990	0.2172
	TFill	0.2085	0.2034	0.2478	0.3147	0.2688	0.2989	0.2923	0.2867	0.2480	0.2867
	ZITS	0.2998	0.2296	0.2614	0.3108	0.2582	0.3135	0.3001	0.2709	0.2722	0.2753
	Ours	**0.1773**	**0.2028**	**0.1869**	**0.2266**	**0.2106**	**0.2023**	**0.1983**	**0.2298**	**0.1774**	**0.2051**

4.3.4 Comparison on various few-shot settings

We conducted comparisons on various few-shot settings on small-scale and large-scale datasets. The term "n-shot" means that n images in each training set in Table 4.1 were selected for training and the test sets were kept unchanged.

The quantitative results of FID scores are shown in Fig. 4.11, Fig. 4.12, Fig. 4.13, Fig. 4.14, Fig. 4.15, Fig. 4.16, Fig. 4.17, and Fig. 4.18. FID and LPIPS scores decrease as the number of training samples increases (e.g., 50-shot images), implying that more

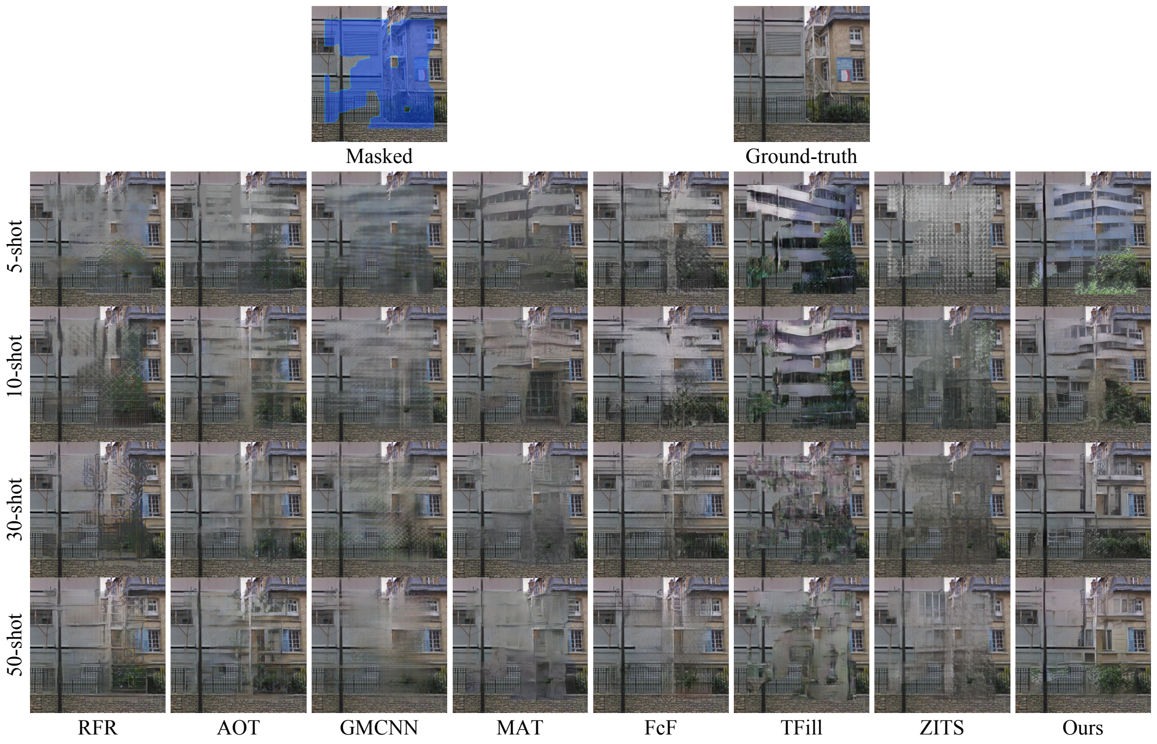


Figure 4.19: Visual comparison of results of models (RFR, AOT, GMCNN, MAT, FcF, TFill, ZITS, and ours) trained on various few-shot settings.

training samples could improve inpainting quality. We further calculated the mean scores of FID and LPIPS across all masked ratios and few-shot settings for each dataset, as shown in Table 4.3. It highlights our GRIG's superiority in few-shot settings. For example, when trained on the Dog dataset, our GRIG achieved a mean FID score of 68.02, indicating a 17.18% relative improvement over the second-best method FcF (with 82.13). The results demonstrate that our method can improve the performance on few-shot scenarios.

Fig. 4.19 presents visual comparisons on various few-shot settings. The results reveal that GRIG achieves greater visual fidelity compared to the SOTA methods.

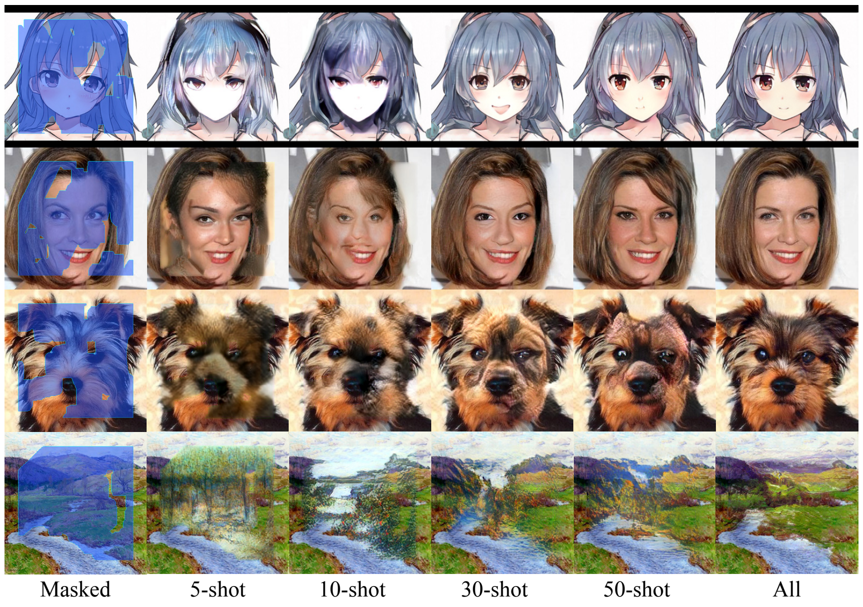


Figure 4.20: Visual results of our GRIG models trained on various few-shot settings. "All" means the training sets mentioned in Table 4.1.

For instance, when trained on 30 and 50 samples, GRIG produces sharp structural and clear texture contents, while compared methods show more blurry results. Fig. 4.20 presents more inpainted examples of our GRIG. The quality of inpainted images drops quickly when models were trained on fewer samples. For example, models trained on 5-shot images are unable to inpaint semantic structures within masked areas; while models trained on 10-shot and 30-shot images can inpaint more plausible contents, some output results still show obvious color inconsistency around mask boundaries. A similar phenomenon is also shown on CMOD and MAT in Fig. 4.6. In contrast, models trained on 50-shot settings produce sharper results with more complex textures and rich colors. We can find that the more training samples the

Table 4.4: Network complexity of various image inpainting methods, including GRIG. **Bold** indicates best.

Method	RFR	AOT	CMOD	Lama	MAT	FcF	TFill	ZITS	GRIG$_{T=1}$	GRIG$_{T=3}$
# Parameters (M)	30.59	**15.20**	79.17	27.05	62.0	70.33	109.45	68.16	31.76	31.76
FLOPs (G)	206.17	72.88	90.25	42.87	139.11	40.26	45.45	270.08	**20.47**	61.41

models trained on, the better their performance on both quantitative and qualitative evaluations. Our method produces more plausible contents when trained on few-shot settings.

4.3.5 Network complexity of recent SOTA methods

In Table 4.4, we show the model complexity of our GRIG and the state of the art methods in terms of number of parameters, and FLOPs needed for a resolution of 256×256. Our GRIG model achieves optimal efficiency with the lowest number of FLOPs at an iterative reasoning step of $T = 1$, and is ranked as the fourth most efficient at $T = 3$. While our model does not have the fewest parameters, its strong performance on small-scale datasets highlights a different strength. This success is not due to a reduced risk of overfitting from fewer parameters; instead, it is attributable to the effectiveness of our proposed framework for data-efficient image inpainting. The robust capacity of our network also plays a pivotal role in securing competitive results on larger-scale datasets. Additionally, our superior image inpainting performance on small-scale datasets, large-scale datasets, and various few-shot settings demonstrate that GRIG shows a good trade-off between image inpainting quality and computational resources.

Table 4.5: Ablation study providing FID scores for (A) GRIG without the forged-patch discriminator, (B) GRIG without the projected discriminator, (C) GRIG with forged-patch discriminator replaced by PatchGAN's discriminator, (D) GRIG with forged-patch discriminator replaced by SM-PatchGAN's discriminator, (E) GRIG with Transformer blocks replaced by down-sampling and up-sampling blocks, and full GRIG . Results were evaluated on 50–60% mask ratios. **Bold** indicates best results.

Dataset	(A)	(B)	(C)	(D)	(E)	GRIG
CHASE	73.96	57.00	56.89	64.17	59.16	**55.84**
Anime	77.49	68.56	66.03	71.12	69.96	**65.05**
Dog	65.33	62.92	61.17	59.83	61.87	**58.49**
Art	96.84	79.19	79.16	78.83	77.35	**77.32**
CelebA-HQ	10.14	8.92	8.41	8.96	8.62	**8.06**
PSV	60.53	61.76	59.62	61.07	61.45	**58.08**

4.3.6 Ablation study

Ablation study on components. We further analyzed the effects of the components of GRIG. To analyze the effects of discriminators in GRIG, we individually removed each discriminator and replaced our forged-patch discriminator with PatchGAN [174] and SM-PatchGAN [104], in turn. All compared discriminators used the same network architecture of 70×70-sized PatchGAN. To demonstrate that incorporating Transformer blocks can further improve the inpainting quality, we tested replacing Transformer blocks [170] with down-sampling and up-sampling blocks [165]. We evaluated inpainting performance to show the impact of these changes. Table 4.5 shows quantitative results of the compared variants. GRIG outperforms all variants in terms of FID score on various small-scale and large-scale datasets. The FID scores increase dramatically when removing either the forged-patch discriminator (model A) or the projected discriminator (model B). Replacing our forged-patch discriminator with other discriminators (models C and D) also leads to higher FID scores. These results indicate that removing our discriminators or replacing the proposed forged-patch discriminator causes a significant degradation in inpainting performance. The best FID scores of our GRIG on various datasets validate the effectiveness of our forged-patch discriminator for performance boosting and mitigating overfitting on small-scale image inpainting. Additionally, without the global context integration of Transformers (model E), the model performs worse. Our generator leverages both advantages of shallow feature extraction and global context reasoning to enhance the

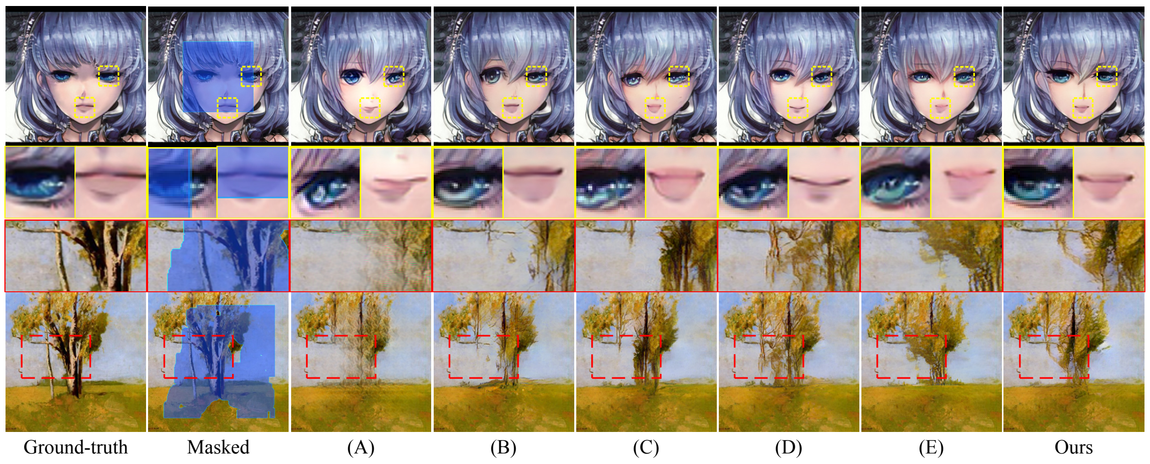


Figure 4.21: Examples from the ablation study with (A) GRIG without the forged-patch discriminator, (B) GRIG without the projected discriminator, (C) GRIG's forged-patch discriminator replaced by PatchGAN's discriminator, (D) GRIG's forged-patch discriminator replaced by SM-PatchGAN's discriminator, (E) GRIG with Transformer blocks replaced by down-sampling and up-sampling blocks, and full GRIG (ours).

visual quality of inpainted images.

Fig. 4.21 shows corresponding visual results. When removing the forged-patch discriminator, the inpainted results show noticeable artifacts around mask boundaries, and the produced textures are blurred, as shown in Fig. 4.21(A). When the projected discriminator is removed, both quantitative performance and visual quality suffer. It is more difficult to maintain the semantic structure of outputs in this case, e.g., the asymmetrical Anime face, as shown in Fig. 4.21(B). The alignment between generated pixels and known pixels may be influenced when we replace our forged-patch discrim-

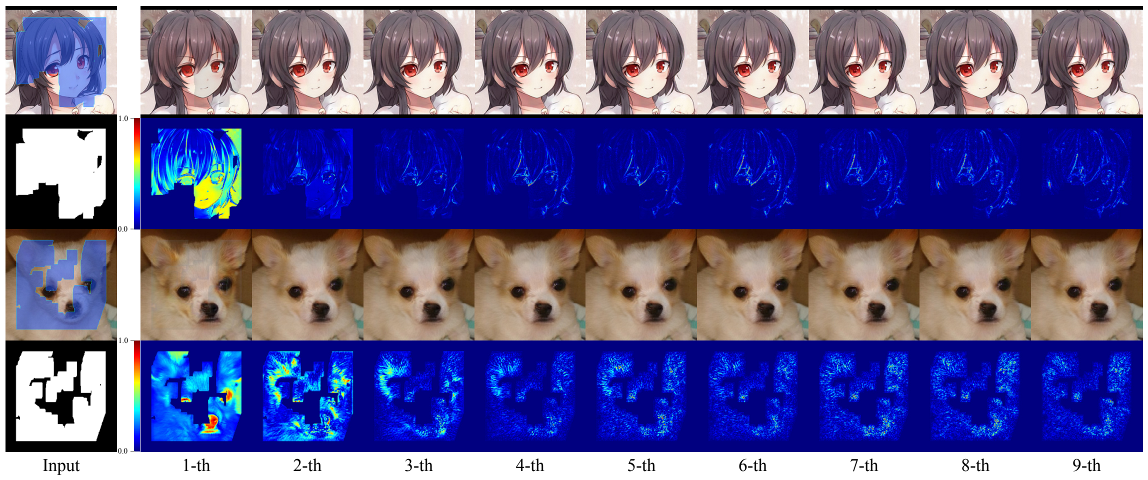
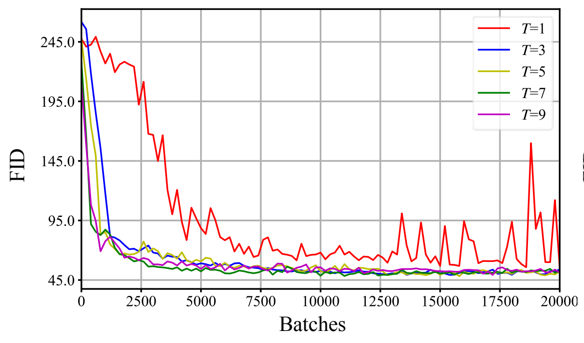


Figure 4.22: Inpainting performance at each iterative reasoning step. For each group: Above-left: masked image. Below-left: input binary mask. Above-right: inpainted images. Below–right: heatmaps of residual outputs Δ_t. Colors red–blue in heatmaps represent higher–lower change for the corresponding pixel.

inator with a Patch-GAN discriminator, as shown in Fig. 4.21(C). When we replace our forged-patch discriminator with an SM-PatchGAN discriminator, it can create plausible content, but consistency with known areas is poor, as seen in Fig. 4.21(D). After replacing Transformer blocks with CNN-based blocks, the trained model excels at inpainting texture and detailed contents, but may not be good at capturing structure information, as shown in Fig. 4.21(E). GRIG shows the best performance on both quantitative and qualitative measures.

Number of iterative reasoning steps. To evaluate the effectiveness of the iterative reasoning in GRIG, we varied the number of iterative reasoning steps T and tested corresponding FID scores on 50–60% mask ratios: see Table 4.6. Each



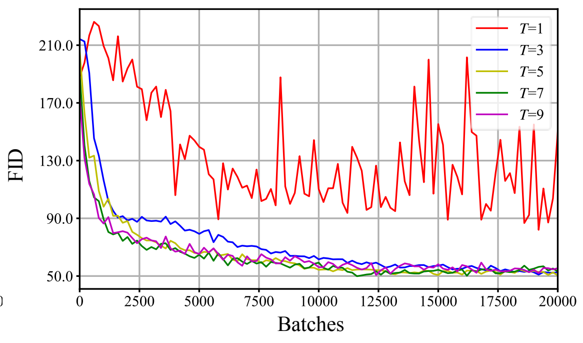


Figure 4.23: Comparisons of FID scores for each training iteration on Anime (left) and Dog (right) datasets. Results were evaluated on a fixed center 25% rectangular mask.

test used the same number of iterative reasoning steps as the corresponding training phase. Compared to models trained for $T = 1$, models trained for $T > 1$ have large performance gains. For example, on the CHASE dataset, the model trained on $T = 3$ has 6.92 lower FID score than that trained for $T = 1$ (55.84 vs 62.76). When $T > 5$, the performance gains saturate or decrease to some extent, but the inpainting performance is still better than for $T = 1$ in most cases. The results indicate that GRIG can produce satisfactory inpainting outcomes in the early steps, while the residual offsets may fluctuate in subsequent steps, potentially leading to variations in inpainting quality. However, GRIG effectively balances the number of steps and the improvement in inpainting quality, achieving superior performance in the data-efficient image inpainting task. In this chapter, we used $T = 3$ to strike a balance between computational cost and visual quality. Fig. 4.22 visualizes the residual output Δ_t for each step t for the model trained with $T = 3$. The masked images were gradually

Table 4.6: FID scores for models trained with various numbers of iterative reasoning steps T. Results were evaluated on 50–60% mask ratios. **Bold** indicates best results.

Dataset	$T = 1$	$T = 3$	$T = 5$	$T = 7$	$T = 9$
CHASE	62.76	55.84	56.58	**53.39**	57.84
Anime	69.55	**65.05**	68.05	66.50	69.20
Dog	63.64	**58.49**	59.66	62.09	61.47
Art	78.40	**77.32**	78.04	78.29	78.23

inpainted. The model prioritizes semantic features in the early steps and fine details in the later steps.

Fig. 4.23 shows an evaluation of GRIG on a fixed center 25% rectangular mask for models trained with $T = 1, 3, 5, 7, 9$, respectively. The FID scores on Anime and Dog datasets show that models trained with more iterative reasoning steps T converge faster than those with fewer T, and models trained with $T = 1$ do not readily converge. Specifically, models trained with $T > 1$ converge for around $10,000$ image batches, whereas models trained with $T = 1$ are far from convergence and fluctuate drastically even after $10,000$ image batches. This shows that our framework can effectively help networks to converge faster.

4.4 Summary

We have taken a first step toward solving data-efficient image inpainting in this chapter. By introducing iterative residual reasoning with decoupled image-level and patch-level discriminators, we have presented a novel data-efficient generative residual image inpainting framework. The proposed generator effectively utilizes CNNs for feature extraction and Transformers for global reasoning. To assist the generative network in learning image fine details, a forged-patch discriminator was introduced. Furthermore, we have established new state-of-the-art performance on multiple small-scale datasets, and extensive experiments have demonstrated the efficacy of the proposed method.

Our method has some limitations. The approach can effectively perform high-fidelity image inpainting on small-scale datasets. However, GRIG cannot directly utilize conditional information for guidance-based image inpainting. Introducing a more sophisticated scheme or module to guide the inpainting process would be interesting for controllable small-scale image completion. Moreover, GRIG is not specialized in diverse image inpainting. Using a mapping network to embed random style codes into the generator could be a good solution for diversity of data-efficient image inpainting.

Chapter 5

Generative Facial Inpainting Guided by Exemplars

5.1 Introduction

Faces are widely recognized as the most representative and expressive aspect of human beings [186, 187, 188]. With the advancement of digital imaging and mobile computing techniques, facial photographs may now be readily collected and distributed. This increases the need for effective and fast facial image altering in a convenient manner while keeping authenticity.

In this chapter, we aim to solve a new face image manipulation problem. The goal is to seamlessly fill in the missing region of an input image by referring to the corresponding content of an exemplar image. This can largely help to generate a satisfactory face image that would favor various application scenarios, including recovering faces occluded by face masks, sunglasses, etc.; synthesizing faces of interest for person identification; designing personalized hairstyles according to existing examples; and generating face makeup for visual effects, to name just a few.

Many face image manipulation methods can achieve impressive manipulation of facial attributes based on guidance information, such as geometries [123, 189], semantics [122], and exemplars [128]. However, these methods often introduce unwanted changes to unedited regions and thus cannot guarantee visual information of known regions unchanged.

Facial inpainting plays an important role in facial image editing for filling missing or masked regions [190]. To achieve realistic facial inpainting guided by exemplar images, there are two main challenges: how to learn the style of facial attributes from the exemplar and how to guarantee natural transition on the mask boundary. Some works [48, 49] attempt to generate diverse image inpainting results allowing users to select a desired one. However, they cannot complete missing regions with user guidance. Many recent methods try to employ additional landmarks [50], strokes [16], or sketches [26, 51] to guide the inpainting of facial structures and attributes. However, these methods tend to overfit the resulting images with this limited guidance information. As a result, these methods still require considerable professional skills in order to generate satisfactory target facial attributes, such as identity, expression, and gender.

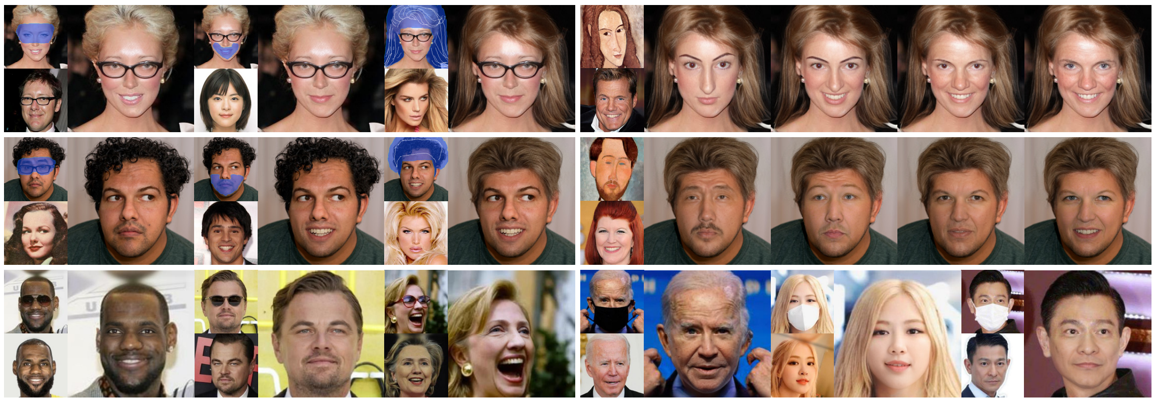


Figure 5.1: Facial inpainting examples using our method. Top two rows: starting with the input image (the top-left sub-image with mask), our method gradually edits the eye style (left), the mouth style (middle left), the hair style (middle), and the facial styles (right) from exemplars. Hairstyles can be edited with the insertion of basic sketches (middle). Real-world and artistic face photos can both be used to direct the inpainting of (blended) facial features in the locally edited regions without affecting the visual content of the rest of the image. Bottom row: for occluded portraits with eyeglasses and masks, we perform guided facial image recovery from exemplars.

To this end, we propose EXE-GAN, a novel interactive facial inpainting framework, which enables high-quality generative facial inpainting guided by exemplars. Our framework consists of four main components, including a mapping network, a style encoder, a multi-style generator, and a discriminator. Our method mixes the global style of the input image, the stochastic style generated from the random latent code, and the exemplar style of the exemplar image to generate highly realistic images. We impose a perceptual similarity constraint to preserve the global visual consistency of the image. To enable the completion of exemplar-like facial attributes, we further employ facial identity and attribute constraints on the output result. To guarantee natural transition across the boundary of inpainted regions, we devise a novel spatial variant gradient backpropagation method for the network training. We compare our method to the state-of-the-art (SOTA) methods to validate its advantages. Experimental results show that our method outperforms competitive methods in terms of visual quality. We also demonstrate several applications that could benefit from our framework, including local facial attribute transfer, guided facial style mixing, hairstyle editing, and guided facial image recovery (see Fig. 5.1).

In summary, this method makes the following contributions:

- A novel interactive facial inpainting framework for high-quality generative inpainting of facial images with facial attributes guided by exemplars.
- A novel self-supervised attribute similarity metric to encourage the generative network to learn the style of facial attributes from exemplars.
- A novel spatial variant gradient backpropagation method for network training to guarantee realistic inpainting with natural transition on the boundary.
- Several applications benefiting from the proposed facial inpainting approach, including local facial attribute transfer, guided facial style mixing, hairstyle editing, and guided facial image recovery.

Our work is also closely related to *image embedding* which enables image synthesis

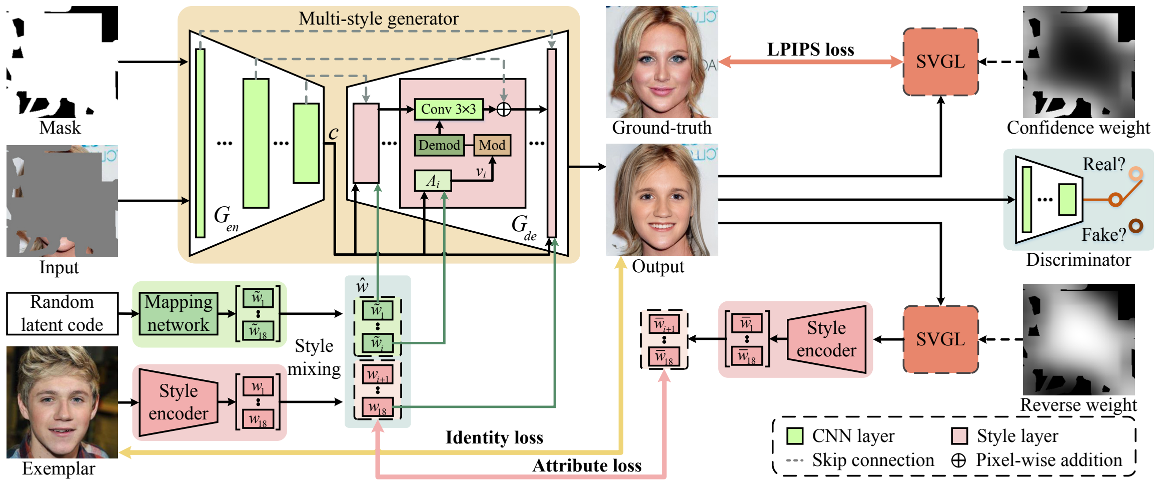


Figure 5.2: Overview of our EXE-GAN framework. We employ style mixing on stochastic and exemplar style codes, and modulate them with the global style code of input image into the multi-style generator for facial inpainting. The adversarial, identity, LPIPS, and attribute losses are integrated into the overall training objective. Spatial variant gradient layers (SVGL) are utilized for natural transition across the filling boundary. Once trained, EXE-GAN can be applied to various application scenarios, such as local facial attribute transfer, guided facial style mixing, hairstyle editing, and guided facial image recovery.

from latent space [7, 42, 60]. StyleGANs [42, 60] and SemanticStyleGAN [6] enable direct scale-specific control of image synthesis with disentangled intermediate latent style space and can produce plausible results for unconditional face image synthesis. Optimization-based embedding [129, 2] and encoder-based embedding [131, 18, 132, 191, 32, 133] methods perform image manipulation by inverting an image to the latent space [192]. Recently, EditingInStyle [193] and StyleFusion [194] show the impressive performance of image editing by semantically manipulating the style latent space. Although image embedding has a strong capability in presenting image styles, these methods may change unedited regions because of information losses in inversion process. For instance, Richardson et al. [18] presented results of inpainting using a pixel2style2pixel framework but failed to preserve the visual contents of unmasked parts. In this work, we propose a novel facial inpainting framework by taking advantage of style latent codes while keeping the unmasked region.

5.2 Method

5.2.1 Overview

The overall structure of our EXE-GAN framework is shown in Fig. 5.2. Given a ground-truth face image $I_{gt} \in \mathbb{R}^{h \times w \times 3}$, an exemplar image $I_{exe} \in \mathbb{R}^{h \times w \times 3}$, and a binary mask $M \in \mathbb{R}^{h \times w \times 1}$ (with value 1 for unknown and 0 for known pixels), the input image $I_{in} \in \mathbb{R}^{h \times w \times 3}$ is obtained by $I_{in} = I_{gt} \odot (1 - M)$, where $\odot$ denotes the Hadamard product. The goal of our EXE-GAN framework is to automatically generate a realistic face image I_{out}, where the inpainting of the masked regions in I_{in} is guided by the facial attributes of I_{exe} while the known regions remain unchanged. The proposed EXE-GAN consists of four main components, including a mapping network, a style encoder, a multi-style generator, and a discriminator.

Mapping network. A multi-layer fully-connected neural network f linearly maps a random latent code $z \in \mathbb{R}^{512\times1}$ to a stochastic style code $\tilde{w} = \{\tilde{w}_i \in \mathbb{R}^{512\times1} | i \in T\} \in \tilde{W}+$, where $\tilde{W}+$ denotes the extended stochastic style latent space and $T = \{1, 2, ..., 18\}$ denotes the index set. Let θ_f be the learnable network parameters in f, we have $\tilde{w} = f(z; \theta_f)$.

Style encoder. A style encoder E directly maps an image to a disentangled style latent space $W+$. Given the network parameters $\hat{\theta}_e$, the style encoder extracts the exemplar style code $w = \{w_i \in \mathbb{R}^{512\times1} | i \in T\} = E(I_{exe}; \hat{\theta}_e)$ and the inpainted style code $\overline{w} = \{\overline{w}_i \in \mathbb{R}^{512\times1} | i \in T\} = E(I_{out}; \hat{\theta}_e)$. Therefore, $w \in W+$ and $\overline{w} \in W+$.

Multi-style generator. A generative network G that leverages multiple representations (i.e., I_{in}, M, and $\hat{w}$) to generate an intermediate result $I_{pred} \in \mathbb{R}^{h\times w\times3}$, where $\hat{w} = \{\hat{w}_i \in \mathbb{R}^{512\times1} | i \in T\}$ is the mixed style code of w and $\tilde{w}$. Let θ_g be the learnable network parameters of G, we have $I_{pred} = G(I_{in}, M, \hat{w}; \theta_g)$. The multi-style generator can be further divided into an encoder G_{en} and a decoder G_{de}, i. e., $G = \{G_{en}, G_{de}\}$.

Discriminator. A discriminative network D learns to judge whether an image is a real or fake image. Let θ_d be the learnable network parameters of D, the discriminative network maps the inpainted image I_{out} to a scalar $D(I_{out}; \theta_d) \in \mathbb{R}^{1\times1}$.

5.2.2 Multi-style modulation

To leverage the global style of the input image, the stochastic style generated from the random latent code, and the exemplar style of exemplar image to perform generative facial inpainting, we propose a multi-style generator by also incorporating the exemplar style and mixing it with other styles based on carefully designed style modulation. The proposed multi-style generator can not only preserve the global visual consistency of the input image, but also embed exemplar facial attributes to the local facial inpainting. In addition, it has the good property of inherent stochasticity with the stochastic style latent code.

First of all, the mixed style code $\hat{w}$ is obtained by style mixing of the stochastic and exemplar styles. Specifically, each layer of $\hat{w}$ is defined as:

$$\hat{w}_i = \begin{cases} w_i & \text{if } \phi_i = 1; \\ \tilde{w}_i & \text{otherwise,} \end{cases} \tag{5.1}$$

where $i \in T = \{1, 2, ..., 18\}$ and $\phi \in \mathbb{R}^{18\times1}$ is a binary vector to indicate which style is modulated for each layer. Coarse-resolution layers correspond to high-level facial attributes and fine-resolution layers could change small-scale features. We empirically set $\phi = [0, 0, 0, 0, 1, 1, ..., 1]$ by balancing the stochastic and exemplar styles in this chapter.

Secondly, the encoder G_{en} takes I_{in} and M as input, and outputs a global style code $c \in \mathbb{R}^{2\times512\times1}$ as well as the corresponding multi-resolution feature maps.

Then, as illustrated in Fig. 5.2, the global style code c and the mixed style code $\hat{w}$ are transformed to multi-style vectors v for subsequent modulation within the style layers of the decoder G_{de}. For each i-th style layer, the transformation is defined as [60]:

$$v_i = A_i([c, \hat{w}_i]), \tag{5.2}$$

where $[\cdot]$ refers to the concatenation operator, A_i is a learned affine transformation within the i-th style layer, and v_i is a linearly learned style representation conditioned on the input style representations.

Next, the decoder G_{de} utilizes the multi-style vectors v and the multi-resolution feature maps output by G_{en} to generate the intermediate inpainting I_{pred}. The decoder contains two style layers in each resolution. In each i-th style layer, the multi-style vector v_i is then used for weight modulation and demodulation [60]. As shown in Fig. 5.2, skip connections are used for collecting the multi-resolution feature maps in the decoder G_{de}.

Finally, the inpainted image I_{out} is generated as follows:

$$I_{out} = I_{in} \odot (1 - M) + I_{pred} \odot M. \tag{5.3}$$

5.2.3 Training objectives

EXE-GAN is trained to optimize the learnable network parameters θ_g, θ_f, and θ_d using the following objectives.

Adversarial loss. We use the adversarial non-saturating logistic loss [41] with R_1 regularization [151]. Specifically, the adversarial objective is defined as:

$$\begin{aligned} \mathcal{L}_{adv}(I_{out}, I_{gt}) = \mathbb{E}_{I_{out}}[\log(1 - D(I_{out})] \\ + \mathbb{E}_{I_{gt}}[\log(D(I_{gt}))] - \frac{\gamma}{2}\mathbb{E}_{I_{gt}}[\|\nabla_{I_{gt}} D(I_{gt})\|_2^2], \end{aligned} \tag{5.4}$$

where γ is used to balance the R_1 regularization term. We empirically set $\gamma = 10$. The generative network G learns to generate a visually realistic image I_{out} while the discriminative network D tries to distinguish between the ground-truth I_{gt} and the generated image I_{out}. G and D are trained in an alternating manner.

Identity loss. We constrain identity similarity between the output image I_{out} and the exemplar image I_{exe} in the embedding space. The identity loss is formulated as follows:

$$\mathcal{L}_{id}(I_{out}, I_{exe}) = 1 - \cos\left(R(I_{out}), R(I_{exe})\right), \tag{5.5}$$

where $R(\cdot)$ is a pre-trained ArcFace network [195] for face recognition.

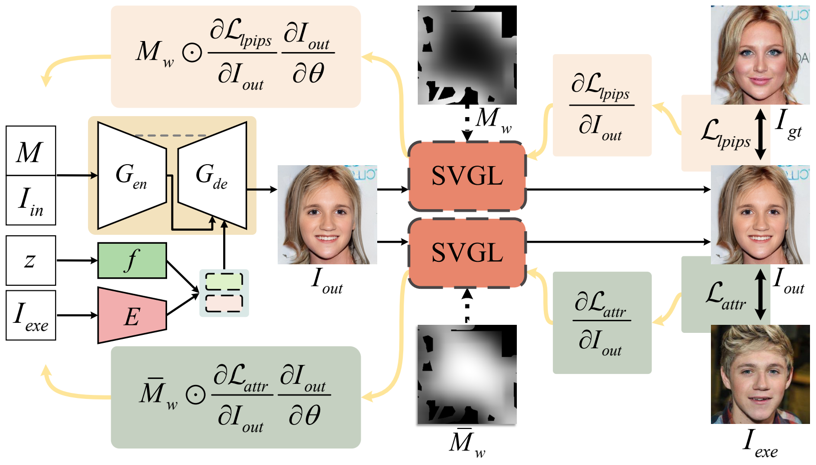
LPIPS loss. We employ the Learned Perceptual Image Patch Similarity (LPIPS) loss [196] to constrain the perceptual similarity between the output image I_{out} and the ground-truth I_{gt}:

$$\mathcal{L}_{lpips}(I_{out}, I_{gt}) = \begin{cases} \|F(I_{out}) - F(I_{gt})\|_2 & \text{if } I_{gt} = I_{exe}; \\ 0 & \text{otherwise,} \end{cases} \tag{5.6}$$

where $F(\cdot)$ is the pre-trained perceptual feature extractor and we adopt VGG [149] in our work. Note that $\mathcal{L}_{lpips}$ is applied only when I_{gt} and I_{exe} are sampled from the same image (see Section 5.3.1 for the detailed settings).

Attribute loss. In order to learn the style of facial attributes from the exemplar image, we introduce a novel self-supervised attribute similarity metric to measure the consistency between facial attributes of the inpainted result I_{out} and the exemplar I_{exe} in the style latent space:

$$\mathcal{L}_{attr}(I_{out}, I_{exe}) = \frac{1}{\|\phi\|_0} \sum_{i \in T} \phi_i \cdot \|\overline{w}_i - \hat{w}_i\|_2, \tag{5.7}$$



$M_w \odot \frac{\partial \mathcal{L}_{lpips}}{\partial I_{out}} \frac{\partial I_{out}}{\partial \theta}$
M
I_{in}
G_{en}
G_{de}
z
f
I_{exe}
E
I_{out}
SVGL
SVGL
M_w
$\bar{M}_w$
$\frac{\partial \mathcal{L}_{lpips}}{\partial I_{out}}$
$\mathcal{L}_{lpips}$
I_{gt}
$\frac{\partial \mathcal{L}_{attr}}{\partial I_{out}}$
$\mathcal{L}_{attr}$
I_{out}
I_{exe}
$\bar{M}_w \odot \frac{\partial \mathcal{L}_{attr}}{\partial I_{out}} \frac{\partial I_{out}}{\partial \theta}$

Figure 5.3: Illustration of the SVGL on LPIPS and attribute losses. In forward-propagation, SVGL does not change any information for I_{out}. In backpropagation, gradients are re-weighted based on the spatial variant M_w and $\overline{M}_w$, respectively.

where the L0 norm $\|\cdot\|_0$ indicates the number of non-zeros.

Total objective. The total training objective can be expressed as:

$$\begin{aligned} O(\theta_g, \theta_f, \theta_d, \hat{\theta}_e) = \mathcal{L}_{adv}(I_{out}, I_{gt}) + \lambda_{id}\mathcal{L}_{id}(I_{out}, I_{exe}) \\ + \lambda_{lpips}\mathcal{L}_{lpips}(I_{out}, I_{gt}) + \lambda_{attr}\mathcal{L}_{attr}(I_{out}, I_{exe}), \end{aligned} \tag{5.8}$$

where λ_{id}, λ_{lpips}, and λ_{attr} are weights of corresponding losses, respectively. We empirically set $\lambda_{id} = 0.1$, $\lambda_{lpips} = 0.5$, and $\lambda_{attr} = 0.1$ in this work. During training, we can obtain the optimized parameters θ_g, θ_f, and θ_d via the minimax game iteratively:

$$\begin{aligned} (\theta_g, \theta_f) = \arg\min_{\theta_g, \theta_f} O(\theta_g, \theta_f, \theta_d, \hat{\theta}_e), \\ (\theta_d) = \arg\max_{\theta_d} O(\theta_g, \theta_f, \theta_d, \hat{\theta}_e). \end{aligned} \tag{5.9}$$

5.2.4 Spatial variant gradient backpropagation

It is expected that the inpainted facial attributes close to the filling center are more similar to those of the exemplar image. Moreover, the inpainted values close to the boundary should be perceptually more similar to those of the input image, and the visual contents should be naturally transited on the boundary. To generate naturally looking inpainting, we further exert constraint based on spatial location.

From Eqs. 5.6 and 5.7, we can find that the LPIPS loss and attribute loss are defined over the entire inpainted image. GMCNN [93] applies the spatial constraint to the pixel-wise reconstruction loss. However, we cannot directly impose GMCNN's spatial constraint on our loss functions. The reason is that our losses (e.g., attribute loss) are defined in the embedding space and the dimensions of embedding features do not match those of the spatial space. In our work, a novel spatial variant gradient layer (SVGL) is designed to impose the spatial constraint on loss gradients in backpropagation.

As illustrated in Fig. 5.3, SVGL has no parameter but relies on a spatial weight mask. During forward-propagation, SVGL acts as an identity transform, which does not change any information from the input. During backpropagation, it collects gradients from subsequent layers, re-weights the gradients based on the spatial weight mask, and passes the re-weighted gradients to the preceding layers.

Mathematically, given an input feature x and a spatial weight mask M_x, we can treat SVGL as a "pseudo-function" $P(x, M_x)$. The forward-propagation and backpropagation behaviors of SVGL are defined below:

$$\begin{aligned} P(x, M_x) &= x, \\ \frac{\partial P(x, M_x)}{\partial x} &= M_x \odot \mathbf{I}, \end{aligned} \tag{5.10}$$

where $\mathbf{I}$ represents an identity matrix.

Then, we apply SVGL to the spatial variant LPIPS and attribute losses. Specifically, we equipped the network with an SVGL $P(\cdot, M_w)$ for the spatial variant LPIPS loss and an SVGL $P(\cdot, \overline{M}_w)$ for the spatial variant attribute loss, respectively, where the confidence weight mask $M_w \in \mathbb{R}^{h \times w \times 1}$ is obtained with Gaussian smoothing on the masked region of M and the reverse weight mask $\overline{M}_w = (1 - M_w) \odot M \in \mathbb{R}^{h \times w \times 1}$. As shown in Fig. 5.3, both SVGLs are added right after the layer of generating I_{out}. Our SVGL is general and can be used to apply spatial constraints to any loss functions with spatial variant backpropagation. Note that the values of non-masked regions are zeros. With the spatial variant gradient layers, the training objective is computed with Eq. 5.8 during forward-propagation while its gradients are computed in a spatial variant manner.

5.2.5 Implementation details

Algorithm 2 lists the pseudo-code for our EXE-GAN framework's training procedure. The threshold $\tau \in [0, 1]$ was used to control the probability that the sampled ground-truth image and exemplar image were the same. We set threshold $\tau = 0.1$ in

Algorithm 2 Training procedure of EXE-GAN

```
1: while $f$, $G$, and $D$ have not converged do
2:     Sample batch images $\mathcal{I}_{gt}$ from training data
3:     Sample random latent vectors $\mathcal{Z}$
4:     Sample a random number $r \in [0, 1]$
5:     if $r >$ threshold $\tau$ then
6:         Sample batch exemplars $\mathcal{I}_{exe}$ from training data
7:     else
8:         Set batch exemplars from ground-truth $\mathcal{I}_{exe} \leftarrow \mathcal{I}_{gt}$
9:     Create random masks $\mathcal{M}$ for $\mathcal{I}_{in}$
10:    Get confidence weight masks $\mathcal{M}_w$ for $\mathcal{L}_{lpips}$
11:    Get reverse weight masks $\overline{\mathcal{M}}_w$ for $\mathcal{L}_{attr}$
12:    Get inputs $\mathcal{I}_{in} \leftarrow \mathcal{I}_{gt} \odot (1 - \mathcal{M})$
13:    Get $\hat{w} \leftarrow mixing(E(\mathcal{I}_{exe}), f(\mathcal{Z}))$
14:    Get $\mathcal{I}_{pred} \leftarrow G(\mathcal{I}_{in}, \mathcal{M}, \hat{w})$
15:    Get outputs $\mathcal{I}_{out} \leftarrow \mathcal{I}_{in} \odot (1 - \mathcal{M}) + \mathcal{I}_{pred} \odot \mathcal{M}$
16:    Update $f$ and $G$ with $\mathcal{L}_{adv}$, $\mathcal{L}_{id}$, $\mathcal{L}_{lpips}$, and $\mathcal{L}_{attr}$
17:    Update $D$ with $\mathcal{L}_{adv}$
```

this chapter. Our framework was implemented using Python and PyTorch. We employed the mapping network f and discriminator D architectures from StyleGANv2 [60], and borrowed the pre-trained the style encoder E from pSp [18]. We kept E frozen for training f, G and D. All weights of f, G, and D are trained from scratch. Following the settings of StyleGANv2 [60], we employed the Adam optimizer with the first momentum coefficient of 0.5, the second momentum coefficient of 0.99, and the learning rate of 0.002. Mixing regularization [60] with a probability of 0.5 was employed to generate stochastic style codes during training. For computing the confidence weight mask in SVGL, the Gaussian kernel is with size 64×64 and its standard deviation is 40. The free-form mask sampling strategy was adopted for training by simulating random brush strokes and rectangles. The brush strokes were generated using the algorithm presented in GConv [26] with maxVertex of 20, maxLength of 100, maxBrushWidth of 24, and maxAngle of 360. Multiple up-to-half-size rectangles and up-to-quarter-size rectangles were generated randomly. The numbers of up-to-half-size rectangles and up-to-quarter-size rectangles were uniformly sampled within $[0, 5]$ and $[0, 10]$, respectively.

5.3 Experiments

5.3.1 Settings

Experiments were conducted on two publicly available face image datasets CelebA-HQ [154] and FFHQ [42]. For CelebA-HQ [154], we randomly selected 28,000 images for training and remained 2,000 images for testing. For FFHQ [42], we randomly selected 60,000 images for training and the rest 10,000 images for testing. Each image was resized to 256×256. We trained the networks for 800,000 iterations with batch size of 8. Unless specified, all experiments were conducted on the NVIDIA Tesla V100 GPU. The training time was around three weeks.

The performance was quantitatively evaluated using the Fréchet inception distance (FID) [155] and the paired/unpaired inception discriminative score (P-IDS/U-IDS) [49]. FID has been proven to correlate well with human perception for the visual quality of generated images. P-IDS and U-IDS are robust assessment measures for the perceptual fidelity of generative models.

5.3.2 Comparisons

Comparison to free-form inpainting. We compared EXE-GAN on CelebA-HQ and FFHQ datasets to the SOTA free-form inpainting methods, including Contextual Attention (CA) [100], Partial Convolutions (PConv) [101], Globally & Locally (G&L) [63], Gated Convolution (GConv) [26], EdgeConnect [51], GMCNN [93], CMOD [49], ZITS [178], TFill [105], Stable Diffusion [69], RGTSI [9], and TransRef [10].

We used the publicly available MMEditing framework [179] for Contextual Attention (CA) [100], Partial Convolutions (PConv) [101], Globally & Locally (G&L) [63], and Gated Convolution (GConv) [26]. MMEditing is an open-source image and video editing toolbox based on PyTorch. We used the official codes for EdgeConnect [51], GMCNN [93], ZITS [178], TFill [105], Stable Diffusion [69], RGTSI [9], and TransRef [10]. For CMOD [49], we used the official TensorFlow-based code to implement a PyTorch-based version. Except for Stable Diffusion [69], all the compared free-form inpainting models were trained using CelebA-HQ and FFHQ datasets, respectively. We used the pre-trained Stable Diffusion [69] sd-v1.5-inpainting model for inference. The positive prompts included "best quality, extremely detailed, real human face, human face, high fidelity face, high-quality face." To avoid low-quality images, we also employed negative prompts such as "bad anatomy, bad hands, missing fingers, extra digit, fewer digits, cropped, worst quality, low quality." For reference-guided methods, RGTSI [9] and TransRef [10], input-reference pairs with high similarity are crucial for inpainting results. However, the CelebA-HQ and FFHQ datasets lack paired ground-truth and exemplar images with the same attributes (identity, expression, decorative goods, etc.). To address this, we adapted the training strategy by copying the ground-truth image and applying random color jittering, translation, flipping, and bilinear scaling operations [184] to create the exemplar image for each iteration. This approach ensures effective training and reasonable results, aligning with the strategies used by RGTSI and TransRef for natural images. Our EXE-GAN does not require paired data due to the proposed self-supervised attribute similarity metric. We randomly chose exemplar images during training and used a reversed batch of input images to guide the inpainting during testing. For a fair comparison, we used the same training/testing splits for all experiments.

Fig. 5.4 shows the quantitative performance comparisons between our method and

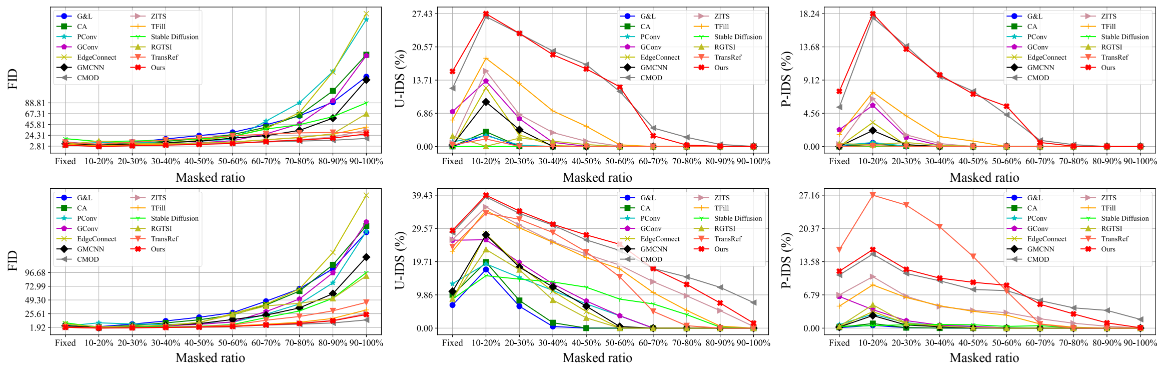


Figure 5.4: Quantitative comparisons of our method to SOTA free-form inpainting methods on the CelebA-HQ (top) and FFHQ (bottom) datasets.

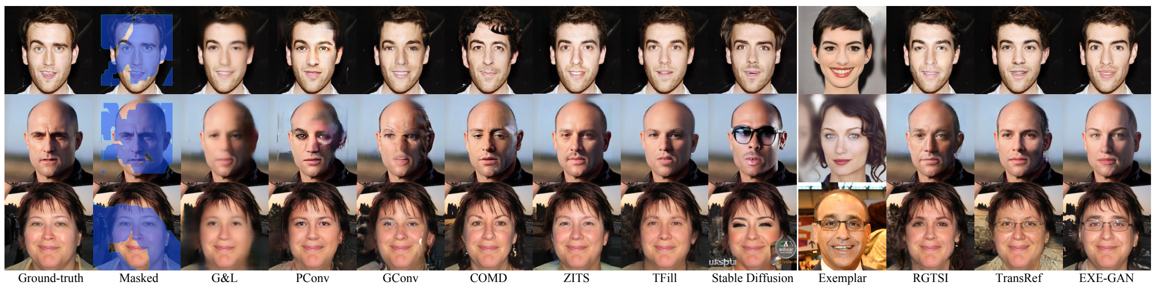


Figure 5.5: A qualitative comparison of our method to current free-form inpainting methods.

the SOTA free-form methods on CelebA-HQ and FFHQ datasets. For easier comparison, we also show the corresponding FID scores in Table 5.1. To simulate various scenarios, we used a fixed center rectangle mask of 128 × 128 and irregular mask templates from PConv [101] for mask ratios ranging from 10-20% to 50-60%. For evaluating large-scale mask inpainting, we generated mask templates using the mask generation procedure of CMOD [49] for ratios ranging from 60-70% to 90-100%. Each range includes 2,000 mask templates. Quantitative results show that our method can

Table 5.1: FID scores for our method and SOTA's free-form inpainting methods on the CelebA-HQ (top) and FFHQ (bottom) datasets. The best is bolded, while the second-best is underlined.

Methods	Fixed	10-20%	20-30%	30-40%	40-50%	50-60%	60-70%	70-80%	80-90%	90-100%
G&L	11.15	5.26	10.62	17.02	23.76	30.22	45.17	63.06	90.06	140.83
CA	7.12	4.70	8.23	12.36	17.50	25.81	40.35	64.27	112.61	184.60
PConv	6.23	10.00	9.44	10.16	12.51	16.60	52.59	89.00	151.03	254.59
GConv	4.46	3.46	5.45	7.12	9.52	13.61	27.08	47.17	93.10	182.71
EdgeConnect	9.53	3.96	6.76	9.84	14.07	22.51	37.53	69.44	150.17	266.11
GMCNN	7.46	4.47	6.63	9.67	12.60	18.75	23.58	34.11	58.48	134.48
CMOD	4.99	2.35	3.79	5.21	6.82	8.64	**11.15**	**12.87**	**14.53**	**18.12**
ZITS	**4.31**	2.81	4.42	5.71	7.26	9.11	11.94	15.34	21.07	34.73
TFill	6.48	3.28	5.27	6.69	8.68	11.04	15.44	20.11	26.66	40.58
Stable Diffusion	17.67	11.95	12.98	14.57	16.85	22.35	36.72	45.43	61.38	88.80
RGTSI	6.03	13.20	8.09	7.40	8.71	11.13	15.35	19.75	27.32	67.33
TransRef	10.39	6.71	10.35	14.61	18.98	23.18	26.69	29.18	30.01	29.10
Ours	4.66	**2.10**	**3.50**	**4.80**	**6.22**	**8.16**	11.86	14.78	18.71	27.15
G&L	6.45	3.26	7.67	13.05	19.47	27.36	47.16	68.56	103.23	166.28
CA	4.19	2.46	5.52	9.70	15.06	23.49	40.67	64.71	110.03	177.22
PConv	4.35	9.84	7.95	6.87	8.50	10.92	24.66	42.43	78.97	167.49
GConv	**1.92**	2.10	3.08	4.73	7.19	11.42	29.29	50.96	96.21	184.05
EdgeConnect	7.92	1.68	3.33	5.83	10.43	25.75	41.80	68.52	131.25	230.16
GMCNN	4.68	1.81	3.45	5.46	8.53	15.56	23.15	35.63	60.19	123.32
CMOD	2.19	0.82	1.41	2.02	2.70	3.76	**6.04**	**7.57**	**9.66**	**14.85**
ZITS	2.09	0.95	1.60	2.33	3.06	4.32	6.56	9.25	13.95	26.09
TFill	**1.92**	1.09	1.74	2.43	3.27	4.32	7.42	10.88	17.45	32.12
Stable Diffusion	10.03	1.98	2.38	2.89	3.77	6.87	18.97	30.33	52.16	96.67
RGTSI	7.78	3.02	4.35	5.21	10.44	24.14	39.09	43.55	51.95	90.94
TransRef	2.84	1.35	1.63	2.24	3.30	6.04	14.37	20.77	30.60	45.17
Ours	2.22	**0.75**	**1.30**	**1.88**	**2.50**	**3.57**	6.27	8.77	13.44	24.18

compete with SOTA methods, despite the fact that its inpainting was guided by exemplars, which was thought to be difficult to maintain image quality [49]. Furthermore, when the masked ratios increase and the useful information from unmasked regions decreases, G&L [63], PConv [101], GConv [26], EdgeConnect [51], and GMCNN [93] often fail to generate high-fidelity results due to their limited generative capacity. For large-scale masks, ZITS [178], TFill [105], and CMOD [49] perform better thanks to their more powerful generative capabilities, but they still have performance limitations. Stable Diffusion [69] has the capability to produce high-quality inpainted results by benefiting from extremely large-scale training data. However, since the highly diverse results may not align with the ground-truth images, the quantitative scores could be affected. RGTSI [9] and TransRef [10] rely on exemplar facial features to guide inpainting, but may struggle when exemplars and ground truths differ in poses, lighting, or extreme styles. Exemplar-guided methods are expected to extract useful exemplar facial features and precisely fill the corresponding facial features to align with the inpainting boundaries. On FFHQ and CelebA-HQ with various masks in terms of FID, U-IDS, and P-IDS, our EXE-GAN achieves overall better quantitative performance than exemplar-guided methods and competitive performance than SOTA free-form inpainting methods.

Fig. 5.5 shows the quantitative performance of compared methods. Although all the methods can compatibly fill in the missing regions, G&L [63] tends to produce blurry inpainting while PConv [101] and GConv [26] fail to inpaint large-scale missing regions. CMOD [49], ZITS [178], and TFill [105] well handle textures and structures inpainting. However, with large masked regions, CMOD would produce a little artifact on masked boundaries (e.g., the first row). Some blurred contents could be found under close inspection in TFill's results. Stable Diffusion [69] can control the inpainting contents via text prompts, but some facial features, such as identity, are difficult to describe using prompts. RGTSI [9] and TransRef [10] produce plausible inpainted results. However, their results do not always accurately reflect exemplar facial features, such as the hazy glasses structure shown in the third row. Our EXE-GAN can generate high-quality inpainted results. Notably, the inpainting of facial attributes cannot be controlled with CMOD, ZITS, and TFill. In contrast, with the help of exemplar facial attributes, the inpainting of facial attributes with our EXE-GAN can be controlled easily.

Comparison to guidance-based inpainting. We compared EXE-GAN on Celeba-HQ dataset to the SOTA guidance-based facial inpainting methods, including sketch-and-color-based facial inpainting SC-FEGAN [16], landmark-based face inpainting LaFIn [50], reference-based inpainting RGTSI [9] and TransRef [10]. For SC-FEGAN and LaFIn, we generated corresponding guided information from exemplar images and alternately took one facial image as the exemplar and the other one as the masked image for facial attribute inpainting. We used the exemplar image directly for inpainting in RGTSI, TransRef, and EXE-GAN. In addition, we used ground-truth structure images as input for RGTSI, following the methodology of the

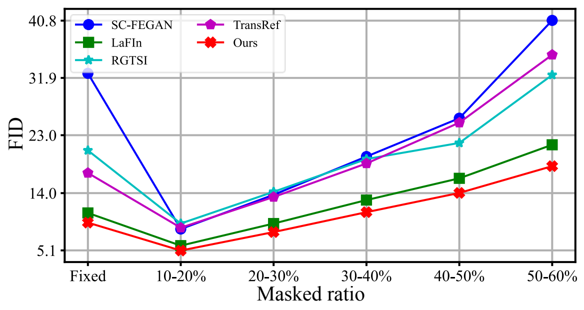


Figure 5.6: On the Celeba-HQ dataset, we performed a quantitative comparison of our method to SOTA guidance-based facial inpainting methods.

original paper.

In this experiment, we used the officially released pre-trained SC-FEGAN [16] and LaFIn [50] models, as well as the trained RGTSI and TransRef models described in Subsection 5.3.2. SC-FEGAN [16] uses sketches and color as guidance to generate missing pixels. Therefore, we leveraged the Canny edge detector to automatically generate sketches from the exemplar image. To avoid inconsistency of color in the inpainted pixels, we did not introduce color information of the exemplar into missing regions. LaFIn [50] relies on landmarks to fill missing regions. Therefore, we utilized the face alignment network FAN [156] to generate landmarks for the exemplar image. To avoid the misalignment between the guidance information (i.e., sketches and landmarks) and unmasked regions in the masked image, we first extracted the angles of roll, pitch, and yaw from the CelebAMask-HQ dataset, then selected 550 pairs with similar poses from the testing set. For each pair, we alternately took one facial image as the exemplar and the other one as the masked image to perform facial

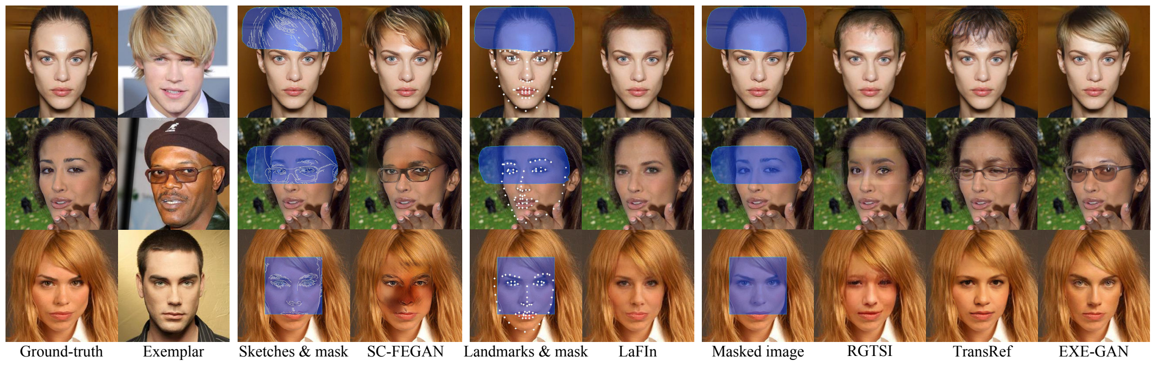


Figure 5.7: A qualitative comparison of our method and the SOTA guidance-based facial inpainting methods.

attribute inpainting. As a consequence, we obtained 1,100 inpainted images for each comparison method.

Fig. 5.6 shows the quantitative comparison of our EXE-GAN to SOTA guidance-based inpainting methods. FID scores with various masked ratios were compared. Experimental results show that our EXE-GAN is able to achieve the best FID scores for all kinds of masks. RGTSI and TransRef are primarily designed for inpainting images with natural scenes, with the assumption that each ground-truth image and its reference (exemplar) image are similar in terms of structure and texture. However, this assumption does not always hold for exemplar-guided facial image inpainting, particularly when the exemplars differ significantly in pose, expression, identity, or gender. In terms of the authenticity of inpainted images guided by exemplars, our method outperforms the compared methods.

As shown in Fig. 5.7, SC-FEGAN [16] effectively generates facial attributes with shapes guided by sketches but requires more information for high-quality facial attributes inpainting. Moreover, there may be visual artifacts in the inpainted images with SC-FEGAN. LaFIn [50] generates facial expressions similar to exemplars but may fail to inpaint the decorative attributes, such as glasses and hairstyles. RGTSI [9] and TransRef [10] can produce high-quality inpainted results, but closer inspection reveals some artifacts around the boundaries. In comparison, our method directly learns the style of facial attributes from the exemplar without extra input, and can produce more realistic facial inpainted results with exemplar-like facial attributes including decorative attributes.

User study. For the user study, we randomly selected 100 pairs of images from the 550 pairs of images with similar poses mentioned above. Then we randomly divided these selected 100 pairs into 5 groups, and each group consists of 20 pairs with different types of inpainting masks. For each pair, we set one image as the masked input and the other as the exemplar to generate inpainting results using the methods of SC-FEGAN [16], LaFIn [50], and our EXE-GAN, respectively. We then randomly ordered one of these 5 groups to a user. For each round, an exemplar image and its corresponding three inpainting images of SC-FEGAN [16], LaFIn [50], and our EXE-GAN were provided. The participants were asked to select the best image based on the visual quality of inpainting and the perceptual similarity to the exemplar. We recruited 63 volunteers to subjectively evaluate the effectiveness of our method comprehensively. The results show that our EXE-GAN obtains the majority of votes (59.67%) compared to SC-FEGAN [16] (12.87%) and LaFIn [50] (27.46%).

The user study validates that the output of our EXE-GAN is more realistic than the compared methods visually observed by subjects.

Comparison to facial attribute transfer. We compared our method on the Celeba-HQ dataset to the SOTA facial attribute transfer methods, including StarGANv2 [122], MaskGAN [123], SimSwap [119], RGTSI [9], TransRef [10], DiffFace [197], and DiffSwap [198], where the same exemplar image was used to guide the attribute transfer.

In this experiment, we used trained RGTSI [9] and TransRef [10] models, as described in Subsection 5.3.2. The pre-trained models of StarGANv2 [122], MaskGAN [123], SimSwap [119], DiffFace [197], and DiffSwap [198] provided in the official repository were used. For StarGANv2 [122], we set the input image as the "reference" image and the exemplar image as the "source" image. For MaskGAN [123], we extracted semantic masks of input images from the CelebAMask-HQ dataset and obtained the style-transferred results based on semantic masks and exemplars. SimSwap [119], DiffFace [197], and DiffSwap [198] directly perform the exemplar-guided face synthesis with the input and exemplar images. RGTSI [9], TransRef [10], and our EXE-GAN synthesize facial attributes for masked regions of input images guided by exemplar images.

As shown in Fig. 5.8, StarGANv2 [122] can transform an input image reflecting the identity of the exemplar. However, it leaves users little freedom to manipulate face images interactively. MaskGAN [122] transfers the style of exemplar to the input face

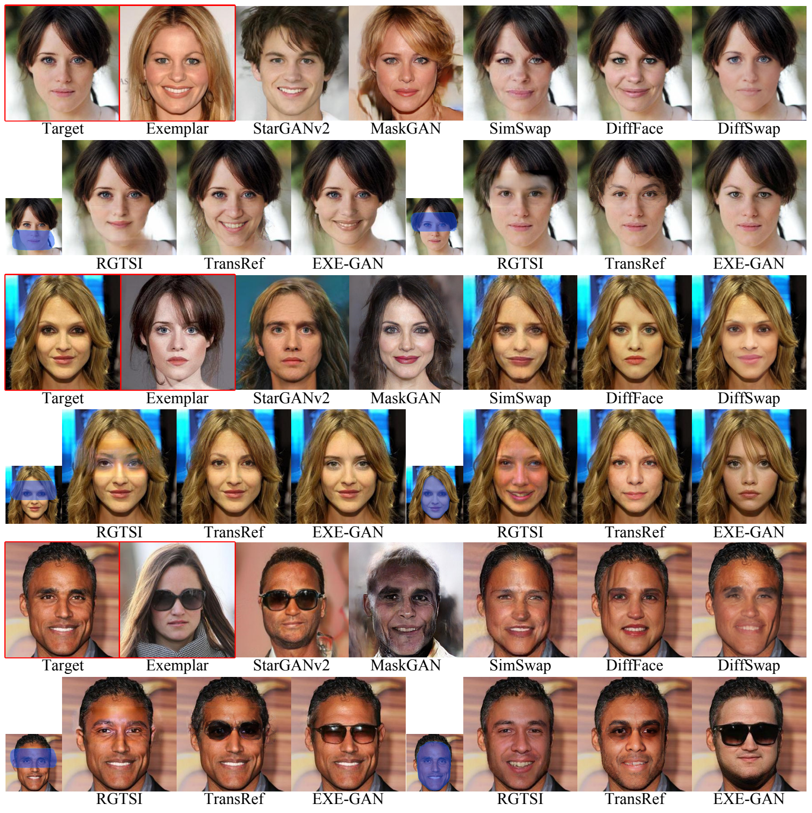


Figure 5.8: A qualitative comparison of our method and the SOTA facial attribute transfer methods.

image using the semantic mask. It requires projecting images into semantic masks and reconstructing images from the mask manifold. As a result, it may introduce irrelevant changes to fine details in the background. SimSwap [119], DiffFace [197], and DiffSwap [198] can transfer the identity of the exemplar face to the input face and preserve the facial attributes of the input. However, they do not allow users to flexibly select regions for face editing and cannot successfully transfer decorative attributes, like sunglasses. RGTSI [9], TransRef [10], and our method not only preserve the pixels of known regions but also allow more degrees of freedom to interactively perform

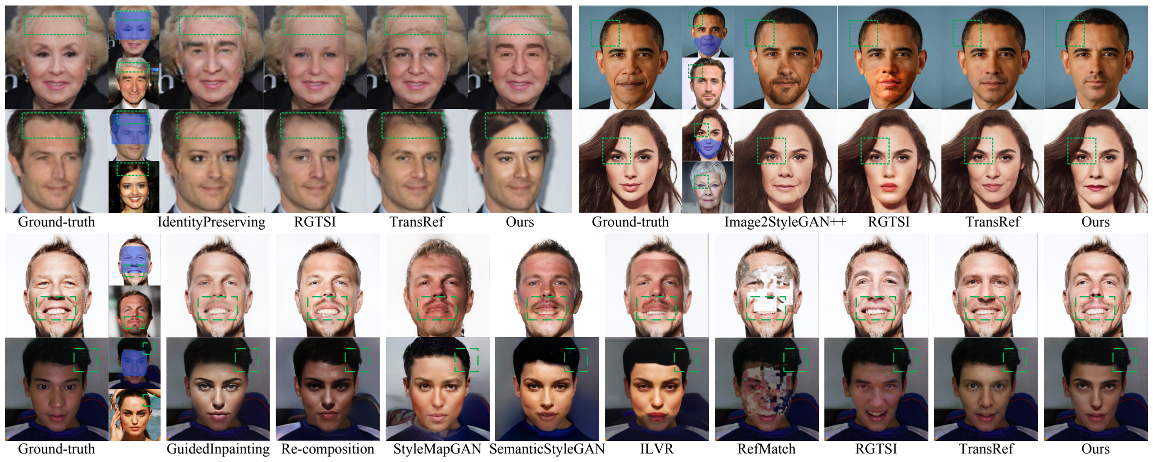


Figure 5.9: Visual comparisons between our EXE-GAN and closely related works (small images in each group: masked and exemplar images). The results of IdentityPreserving [1] and Image2StyleGAN++ [2] are taken from their respective papers. The results of GuidedInpainting [3] and Re-composition [4] are from paper [4]. The results of StyleMapGAN [5], SemanticStyleGAN [6], ILVR [7], and RefMatch [8] are generated using their publicly available trained models. The results of RGTSI [9] and TransRef [10] are generated from models re-trained from scratch.

facial attribute manipulation. However, because the exemplar facial features may conflict with those around mask boundaries, RGTSI and TransRef struggle to ensure a natural transition across the inpainted region boundaries. Thanks to our self-supervised attribute similarity metric for learning exemplar facial features and SVGL for enhancing natural transitions, EXE-GAN can produce high-quality results with facial attributes guided by exemplars, including gender, makeup style, hairstyle, and decorative style (e.g., glasses).

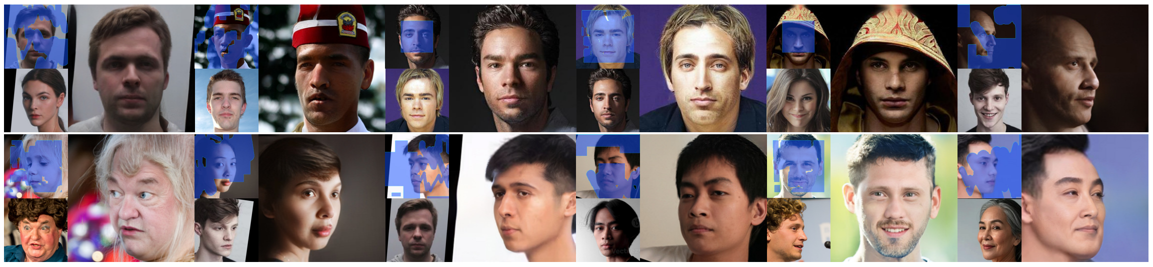


Figure 5.10: Visual results of EXE-GAN on images with low light and large pose variations (from top left to right in each group): masked image, exemplar image, inpainting result.

More comparisons to closely related works. More comparisons to closely related works are shown in Fig. 5.9. Although most methods produce plausible results, some visual artifacts can be seen in the details of IdentityPreserving [1] and GuidedInpainting [3], RefMatch [8], RGTSI [9], and TransRef [10]. Color inconsistencies may occur in SemanticStyleGAN [6], ILVR [7], and RefMatch [8]. In addition, Image2StyleGAN++ [2], Re-composition [4], StyleMapGAN [5], SemanticStyleGAN, and ILVR introduce unwanted changes in background or unedited regions. RefMatch [8] hypothesizes that the exemplar and targeted inpainting images have very similar contexts, such as the same landmark building. The inpainting quality deteriorates when the exemplar exhibits different facial features, such as poses, identity, or styles, that violate assumptions, as discussed in RefMatch's limitation section. On the other hand, our EXE-GAN can seamlessly fill in the masked pixels using exemplar-like attributes without changing unmasked areas, yielding high-quality inpainting results while avoiding the above artifacts.

Table 5.2: Network complexity and computational efficiency of compared methods. **Bold** indicates the best.

Method	EdgeConnect	GMCNN	CMOD	TFill	ZITS	RGTSI	TransRef	EXE-GAN
# Parameters (M)	21.54	**12.56**	79.17	109.45	67.90	175.55	41.97	79.18
FLOPs (G)	122.64	67.20	90.25	45.45	182.73	146.67	**7.55**	90.26
FPS	73.59	**121.58**	77.79	43.55	11.55	34.56	81.60	77.62
Training time (H)	331.84	**61.44**	192.28	287.04	299.53	123.92	88.41	269.49

5.3.3 Performance of EXE-GAN on challenging cases

Fig. 5.10 illustrates the exemplar-guided inpainting of EXE-GAN on challenging cases, including low-light conditions (first row) and large pose variations (second row), using various masks. In the first row, the inpainted images retain the low-light characteristics of the masked inputs, but faces remain recognizable. Additionally, using a low-light image as an exemplar (4-th case), the inpainted result clearly exhibits the exemplar's features while maintaining high fidelity. The second row shows that EXE-GAN effectively handles large pose variations. For instance, the second-last case demonstrates that EXE-GAN performs well even with extreme exemplar poses; the last case shows that despite using a completely different pose for inpainting, the resulting image maintains good quality.

5.3.4 Analysis on network complexity and computational efficiency

In Table 5.2, we show the model complexity and computational efficiency of our EXE-GAN and the SOTA methods in terms of the number of parameters, FLOPs, frames per second (FPS) in the inference phase, and training time (hours). All methods were evaluated on a machine with a single RTX 3090 GPU using images with the size of 256×256. To estimate the training time, we first recorded the time taken for 1,000 iterations on the CelebA-HQ dataset for each module of each method using their default hyperparameters. We then multiplied this time by the total number of 1000-iteration chunks required and summed the times for all modules to get the estimated training time for each method. All the models show good efficiency and practical viability, with no more than 200.0M learnable parameters. Most models, including GMCNN, TransRef, and EXE-GAN, can perform real-time inference and process more than 70 images per second. For example, TransRef only needs 7.55 GFLOPs for each image, showing excellent computational efficiency with robust inpainting performance. While our model does not have the fewest parameters, the moderate capacity of our network is critical in achieving competitive results when learning to use the exemplar facial features. Furthermore, our competitive image inpainting performance on the CelebA-HQ and FFHQ datasets shows that EXE-GAN achieves a good balance between image inpainting quality and computational resources.

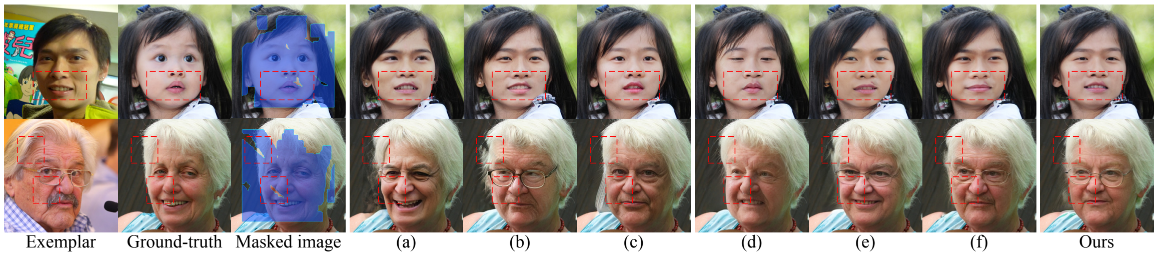


Figure 5.11: Qualitative examples of the ablation study for large-scale facial inpainting by exemplars with (a) EXE-GAN without any SVGLs in $\mathcal{L}_{attr}$ and $\mathcal{L}_{lpips}$, (b) EXE-GAN without SVGL in $\mathcal{L}_{attr}$, (c) EXE-GAN without SVGL in $\mathcal{L}_{lpips}$, (d) EXE-GAN without $\mathcal{L}_{attr}$, (e) EXE-GAN without $\mathcal{L}_{lpips}$, (f) EXE-GAN without $\mathcal{L}_{id}$, and (Ours) EXE-GAN.

Table 5.3: Ablation study for large-scale facial inpainting by exemplars with (a) EXE-GAN without any SVGLs in $\mathcal{L}_{attr}$ and $\mathcal{L}_{lpips}$, (b) EXE-GAN without SVGL in $\mathcal{L}_{attr}$, (c) EXE-GAN without SVGL in $\mathcal{L}_{lpips}$, and (Ours) EXE-GAN. Results are averaged over 5 runs. **Bold**: top-2 quantity.

Method	CelebA-HQ			FFHQ		
	FID$^{\downarrow}$	U-IDS$^{\uparrow}$	P-IDS$^{\uparrow}$	FID$^{\downarrow}$	U-IDS$^{\uparrow}$	P-IDS$^{\uparrow}$
(a)	12.853	4.024%	1.25%	7.298	17.29%	6.55%
(b)	10.433	**8.875%**	3.35%	4.909	22.46%	8.75%
(c)	**9.804**	**8.875%**	**4.25%**	**4.408**	**24.61%**	**10.04%**
Ours	**9.967**	**9.175%**	**3.85%**	**4.353**	**24.33%**	**9.92%**

5.3.5 Ablation study

Ablation study on SVGL. We further investigated the effectiveness of SVGL in our EXE-GAN via an ablation study. The free-form mask sampling strategy mentioned in Subsection 5.2.5 was adopted. We present qualitative examples in Fig. 5.11 (a-c) to express the visual effects of the ablation study. Table 5.3 shows the quantitative results. When SVGLs are removed from both the attribute and LPIPS losses, both quantitative measures and visual qualities drop dramatically, and completed images may not show clear manifestations of exemplar facial images in attributes, such as wearing glasses in Fig. 5.11 (a). When replacing the SVGL-based attribute loss with the standard attribute loss without SVGL, there may be visible boundary inconsistencies in the generated results (see Fig. 5.11 (b)), and the quantitative performance is also affected. When replacing the SVGL-based LPIPS loss with the standard LPIPS loss without SVGL, the visual similarities of facial attributes (e.g., facial expression, wearing glasses) between the generated result and the exemplar image decrease, while the quantitative scores are comparable to EXE-GAN for both testing datasets, as demonstrated in Fig. 5.11 (c). In this case, the standard LPIPS loss is applied to all pixels of the image to enforce the generator to reconstruct the contents of ground-truth instead of exemplar attributes. In comparison, our EXE-GAN is able to produce realistic facial images with facial attributes similar to exemplars while achieving competitive quantitative scores.

Ablation study on attribute loss, LPIPS loss, and identity loss. An-

Table 5.4: Ablation study for large-scale facial inpainting by exemplars with (d) EXE-GAN without $\mathcal{L}_{attr}$, (e) EXE-GAN without $\mathcal{L}_{lpips}$, and (Ours) EXE-GAN. Results are averaged over 5 runs. **Bold**: top-2 quantity.

Method	CelebA-HQ			FFHQ		
	FID$^{\downarrow}$	U-IDS$^{\uparrow}$	P-IDS$^{\uparrow}$	FID$^{\downarrow}$	U-IDS$^{\uparrow}$	P-IDS$^{\uparrow}$
(d)	**9.714**	**11.38%**	**4.65%**	**4.453**	**23.84%**	**9.86%**
(e)	10.207	7.37%	2.95%	4.726	23.50%	9.53%
Ours	**9.967**	**9.175%**	**3.85%**	**4.353**	**24.33%**	**9.92%**

other ablation study was carried out to investigate attribute loss and LPIPS loss in EXE-GAN. The free-form mask sampling strategy mentioned in Subsection 5.2.5 was employed. Fig. 5.11 (d-e) and Table 5.4 show visual comparisons as well as quantitative results. When the attribute loss in EXE-GAN is removed, the model produces comparable results but reduces the visual similarity of facial attributes between the generated result and the exemplar image, as shown in Fig. 5.11 (d). When the LPIPS loss is removed, both the quantitative and qualitative measures decrease rapidly, and visual artifacts can be seen around the mask boundaries, as illustrated in Fig. 5.11 (e). In contrast, EXE-GAN results demonstrate high fidelity with facial attributes similar to exemplars and competitive quantitative scores.

To further demonstrate the effectiveness of identity loss in preserving the identity information of exemplar images, we assessed the identity distance metric (IDD) in ArcFace's feature space. We trained another model without $\mathcal{L}_{id}$ and tested it alongside

Table 5.5: IDD scores for facial inpainting by exemplars with (f) EXE-GAN without $\mathcal{L}_{id}$, and (Ours) EXE-GAN. **Bold**: top-1 quantity.

Methods	Fixed	10-20%	20-30%	30-40%	40-50%	50-60%	60-70%	70-80%	80-90%	90-100%
(f)	1.277	1.449	1.430	1.407	1.376	1.314	1.292	1.251	1.211	1.193
Ours	**1.264**	**1.447**	**1.425**	**1.401**	**1.368**	**1.303**	**1.283**	**1.240**	**1.197**	**1.161**

our EXE-GAN on the CelebA-HQ test set with different masked ratios. Fig. 5.11 (f) shows that without $\mathcal{L}_{id}$, the trained model can still reflect the exemplar attributes in the inpainted images with similar visual quality. However, the IDD similarity results in Table 5.5 indicate that the inpainted images of Model (f) lose some identity information of the given exemplar image. By incorporating identity loss, our EXE-GAN performs better in preserving the identity information guided by exemplars.

Ablation study on exemplar styles modulation. We performed ablation experiments on the modulation of exemplar styles by re-training our model with different vector ϕ configurations, and ϕ is a binary vector that indicates which style for each style layer is modulated. According to Fig. 5.12, the more exemplar style codes that are modulated, the more exemplar facial attributes will appear in the inpainted images.

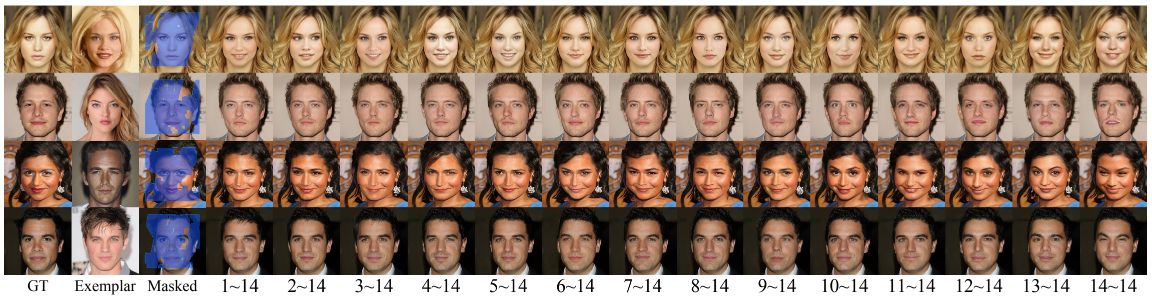


Figure 5.12: Examples of facial inpainting with various subsets of the style codes: ground-truth, exemplar, masked image, and various style effects. In each row, values of the i-th to j-th layers in the style code are from the exemplar, and values of the remaining layers are from the stochastic style code.

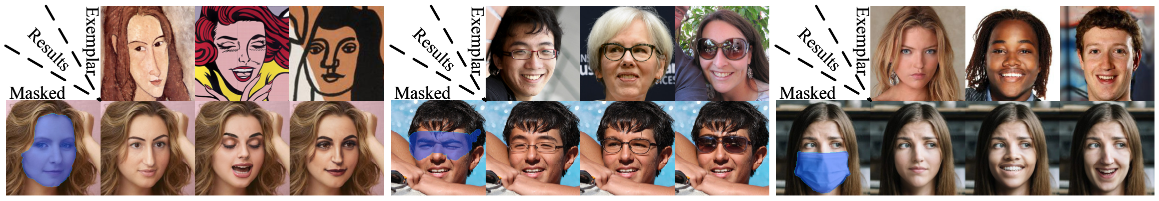


Figure 5.13: Examples of local facial attribute transfer.

5.4 Applications with our facial inpainting

5.4.1 Local facial attribute transfer

Since our EXE-GAN helps the generator learn the mapping between injected exemplar representations and corresponding facial attributes, our method can be used to produce vivid facial attribute transfer effects guided by various exemplars, such as real-world facial attributes and artistic expressions. As shown in Fig. 5.13 and Fig. 5.14, for a masked input, our EXE-GAN produces high-quality local facial at-

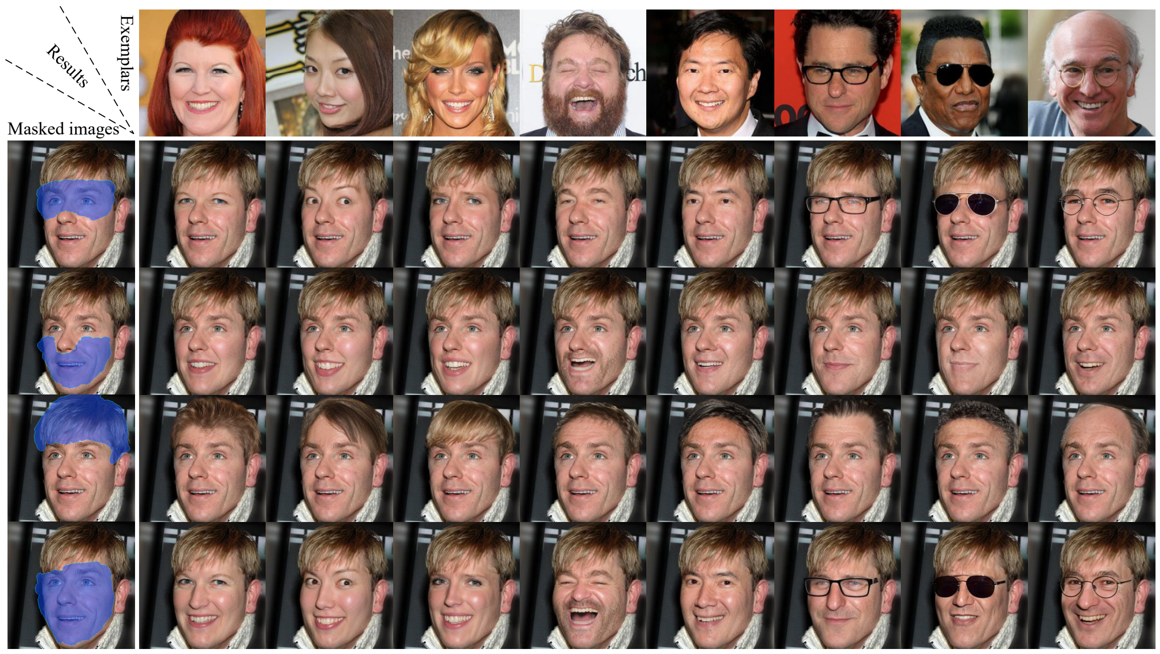


Figure 5.14: More examples of local facial attribute transfer guided by exemplars.

tribute transfer results by leveraging facial attributes of exemplars.

5.4.2 Guided facial style mixing

Our EXE-GAN can be used to generate facial inpainting effects by mixing two exemplar style latent codes. We first employ the style encoder E to obtain two exemplar style codes from two exemplars, respectively. Then, we apply the style mixing [42, 60] on the two latent codes with a crossover point. By simply changing the crossover point, we can obtain multiple mixed latent codes. Therefore, guided facial style mixing effects can be obtained by moving the crossover point over the vector ϕ in Eq. 5.1, as shown in Fig. 5.15.

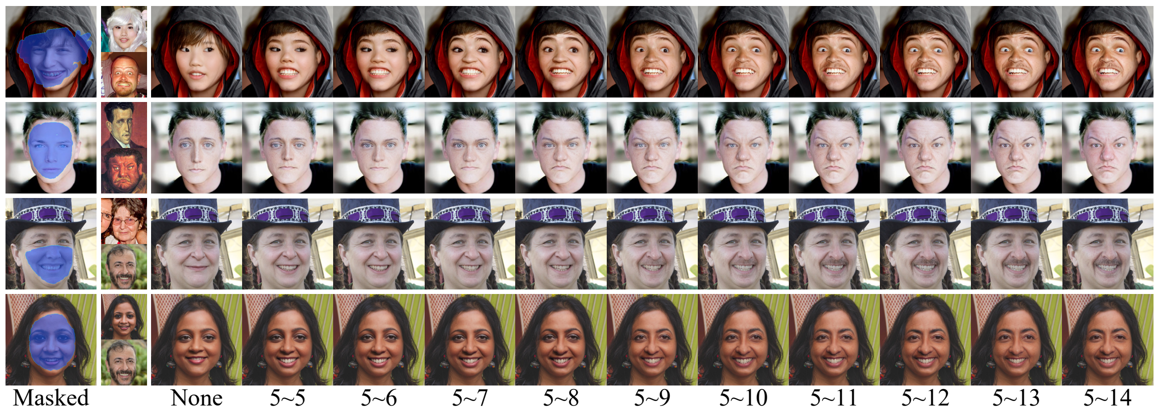


Figure 5.15: Examples of facial style mixing (from top left to right in each group): masked image, pairs of exemplars, and style-mixing effects. In each row, values of the style code of the first exemplar from i-th to j-th channels are replaced by those of the second exemplar.

5.4.3 Hairstyle editing

We further fine-tuned our trained EXE-GAN with extra hand-drawn-like sketches which were produced automatically with a pencil-sketch filter [18]. The application allows users to sketch in the mask to indicate roughly the hair styles. Given a masked image with sketches and an exemplar image, EXE-GAN produces a style-edited output. As shown in Fig. 5.16, various hairstyle editing results are obtained by changing the user-edited style sketches. It is easy even for a novice to obtain various styles by simple sketch editing.

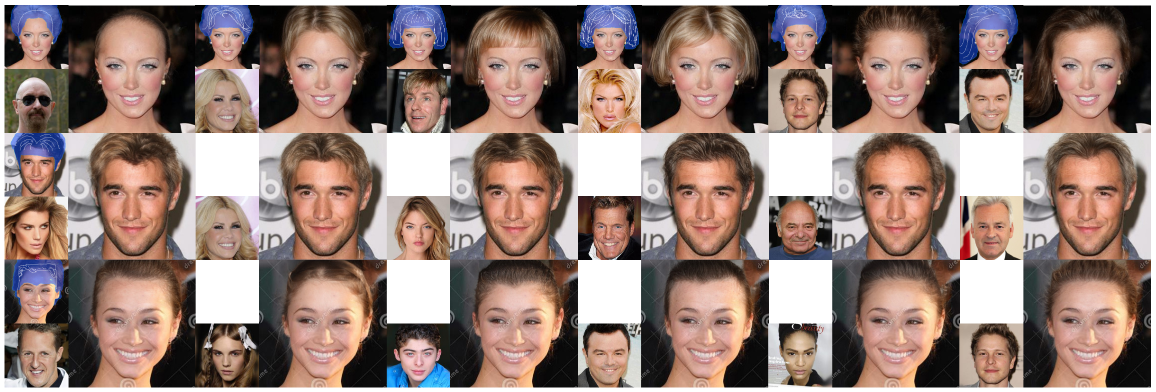


Figure 5.16: Examples of hairstyle editing: (the first row) edited results with different sketches guided by different exemplars and (last two rows) edited results with the same sketch guided by different exemplars.

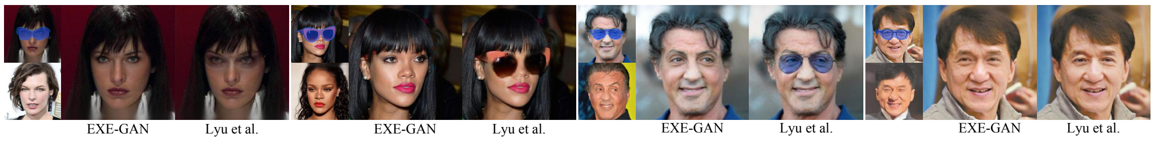


Figure 5.17: Comparison on portrait eyeglasses removal (from top left to right in each group): masked image, exemplar image, our recovered result, and Lyu et al.'s result [11].

5.4.4 Guided facial image recovery

The guided facial image recovery for occluded portrait eyeglasses and masks was achieved by masking them out and taking a different image from the same person as exemplar. As shown in Fig. 5.17, we compared our method to Lyu et al.'s eyeglasses removal method [11] on tinted eyeglasses (leftmost), sunglasses (mid-left), and myopia glasses (mid-right). The results show that our guided facial image recovery method

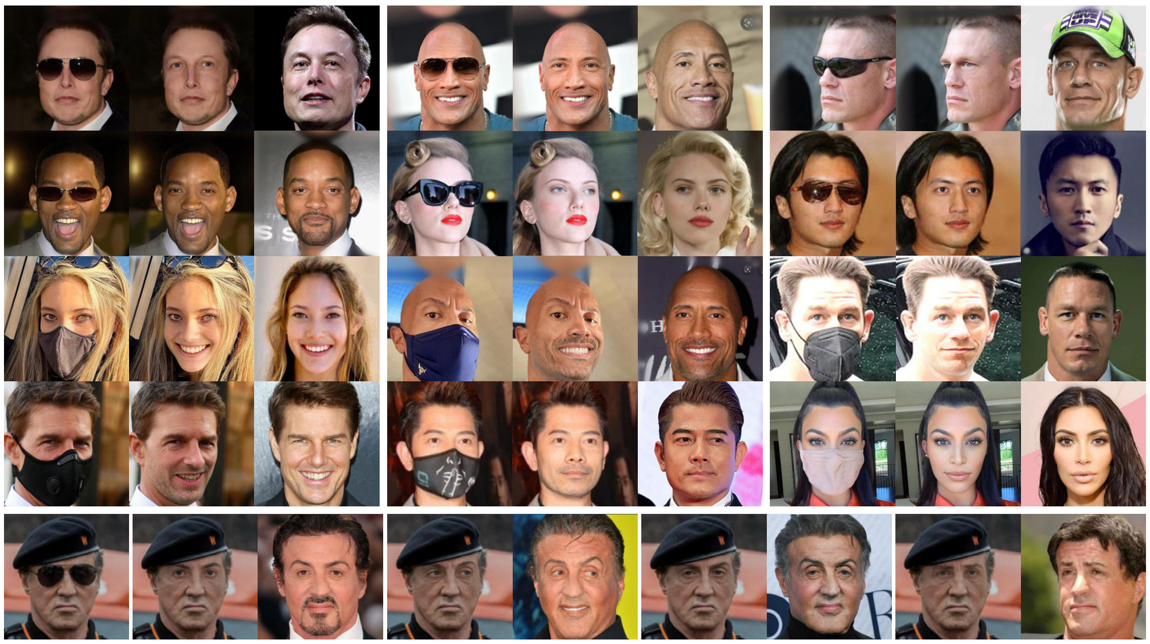


Figure 5.18: Examples of guided facial image recovery: (top four rows, from left to right in each group) occluded image, recovered face image, and exemplar; (bottom row) diverse recovered results guided by different exemplars.

performs well in removing tinted eyeglasses and sunglasses which may fail with Lyu et al.'s method. Fig. 5.18 shows that our method can also recover faces from occluded glasses or masks effectively.

5.4.5 Inherent stochasticity

EXE-GAN can produce multiple diverse facial inpainting results for an input masked facial image and an exemplar image by leveraging the inherent stochasticity. Users can easily select the preferred one among these results. The inherent stochasticity is achieved by adding per-pixel noise after each convolutional layer, leveraging the

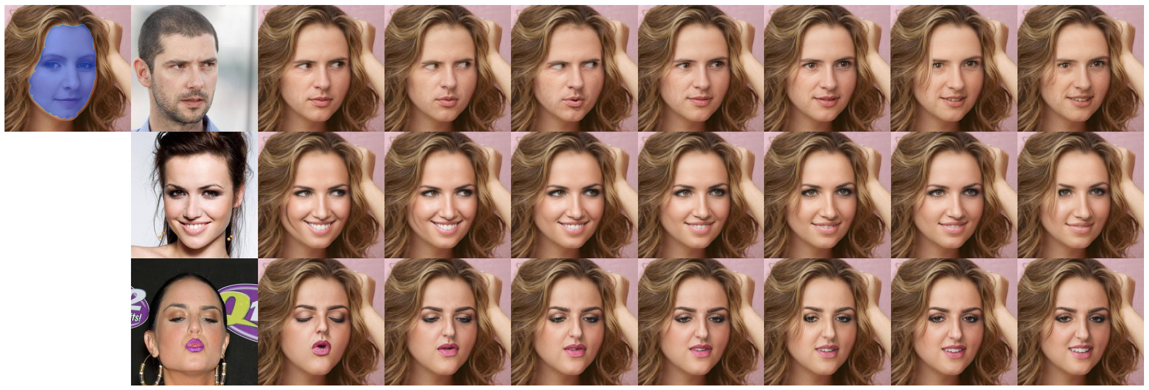


Figure 5.19: Examples of diverse facial inpainting with inherent stochasticity. Based on the same masked image, we use different exemplars to guide the generation of various results.

injected random latent code, and applying a truncation trick to tune the stochastic style representations [42, 60]. Fig. 5.19 shows a variety of facial inpainting results with various random latent codes.

5.5 Summary

In this chapter, we have presented a novel interactive framework for realistic facial inpainting by taking advantage of exemplar facial attributes. An attribute similarity metric was introduced to help the generative network learn the style of facial attributes from the exemplar. We further proposed a novel spatial variant gradient backpropagation technique to address the issue of visual inconsistency on the filling boundary. Extensive experiments and applications have demonstrated the effectiveness of our method.

Our method has some limitations. Using the embedded style codes, we successfully transfer the facial attribute styles from the exemplar image. The explicit mapping between the facial attribute and the embedded style codes, on the other hand, is still unknown [18]. Incorporating a more advanced embedding algorithm into our pipeline could be a good next step. The trained model works well for aligned images because the facial images in the experimented training datasets [154, 42] are highly aligned. It is necessary to align and crop the inputs before inpainting nonaligned images. It would be preferable to train models on unstructured datasets to create a more sophisticated algorithm.

Chapter 6

Multimodal Generative and Fusion Framework for Facial Editing

6.1 Introduction

The rapid development of digital imaging and mobile computing has fueled the demand for personalized content in social media and various applications [199, 200, 201], making facial image editing an essential research area in computer graphics and computer vision. Faces are universally acknowledged as the most representative and expressive aspect of human beings, which makes facial editing a challenging task [202].

Many image editing tools provide convenient guidance information to allow users to edit facial features interactively [203]. In order to provide convenient user interfaces, many face image editing tools make use of various input modalities to guide the editing of facial features. In recent years, multimodal facial image editing has attracted considerable interest. Since multimodal models excel at conveying various types of conditioning information, their combined synergy can offer clearer descriptions to aid in facial editing. For example, semantics can define a facial image's coarse layout; sketches can detail its structure and texture; and text or attribute labels can adjust facial attributes, to name just a few. The support of local editing capability is also important in a variety of image editing applications. Local editing enables users to edit local image regions in an incremental manner while keeping the contents of unedited background regions unchanged.

There are some limitations in existing multimodal local facial editing methods. The first limitation is the difficulty in maintaining visual contents in unedited background regions. Existing multimodal facial editing techniques [52, 14, 12, 13] can only edit the facial image as a whole and are prone to introduce unwanted changes to unedited background regions. When users are not satisfied with some local effects, these techniques will fail to edit the local regions in an incremental manner. As shown in Fig. 6.1 (bottom two rows), while state-of-the-art (SOTA) methods may perform high-quality edits, they are likely to include unwanted changes of other facial features in incremental editing scenarios, where an already edited image is subject to further modifications using different modalities. Some existing methods [52, 14] have the limitation in manual annotations of paired data. These methods train their models with labeled paired data across different modalities but manual annotations of train-

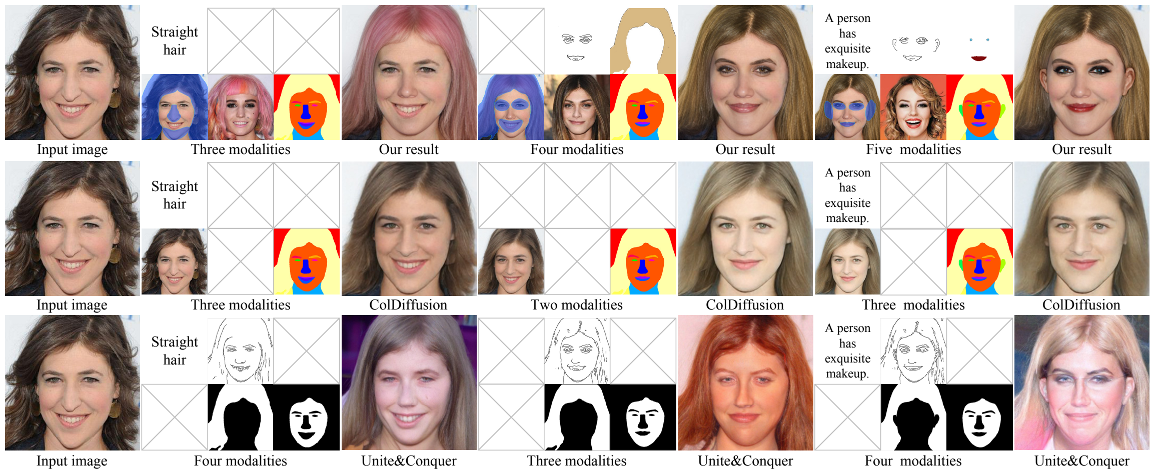


Figure 6.1: Examples demonstrating the superior performance of FACEMUG in high-quality globally consistent local facial editing, using subsets of the five modalities including semantic label, sketch, text, color, and exemplar image. Our method (top row) exhibits better visual quality and fidelity in incremental editing (the later editing taking the previous output image as input), compared to SOTA multimodal face editing methods: ColDiffusion [12] (middle row) and Unite&Conquer [13] (bottom row).

ing datasets are label-intensive. Recent methods based on diffusion models [12, 13] instead train all uni-modal models first and perform multimodal facial editing by integrating these pre-trained uni-modal models. However, when the number of modalities grows, more uni-modal models should be trained separately with these methods.

With this in mind, we explored ways to tackle these limitations. We investigated whether incorporating generative adversarial networks (GANs) [41] can improve global consistency for multimodal local facial editing. By learning the distribution of real facial images, adversarial training enforces the model to fill plausible contents for edited regions guided by multimodalities. To minimize the dependency on paired training data, we asked the question that if we can loosen the ties between the paired modalities data by aligning all modalities into a unified generative latent space to diminish the requirement for paired text, attribute label, and exemplar modalities. Instead of training a uni-modal model for each modality, we examined whether fusion and warping priors, along with multimodalities in both latent and feature space, could achieve seamless integration of multimodalities.

We thus introduce a **MU**ltimodal **G**enerative and fusion framework for local **FAC**ial **E**diting (FACEMUG), which can solve the above problems. First, since the StyleGAN latent space [60] is disentangled well, we design our framework by bridging all modalities to the StyleGAN latent space. Second, for the seamless integration of multimodalities, we design our multimodal generator with fusion and warping in latent and feature space. To support the heterogeneity and sparsity of the pixel-wise conditional inputs, we aggregate multimodal conditional inputs into a homogeneous feature space. Since a fully trained GAN model excels at capturing rich textures and structural priors [204], we utilize a StyleGAN generator as a facial feature bank to provide candidate facial features and introduce style fusion blocks to fuse facial features for improving the generation quality. Moreover, to rectify the pose misalignment between the edited image and the given latent codes, we present a self-supervised latent warping method to efficiently transfer the pose of the edited image to that of the given latent codes in the latent space. To simulate the latent editing process during training and boost facial editing capabilities, a diversity-enhanced attribute loss is proposed.

To the best of our knowledge, our FACEMUG is the first method that generates realistic facial features in response to multimodal inputs on the edited regions while maintaining visual coherence with the unedited background to achieve global consistency. We have conducted extensive comparisons of FACEMUG against the SOTA methods and comprehensive experiments to demonstrate the superiority of FACEMUG in terms of editing quality, flexibility, and semantic control, illustrating its potential to significantly enhance various applications within facial editing.

In summary, our paper makes the following contributions:

- A novel globally-consistent local facial editing framework that enables diverse facial attribute manipulation.
- A novel multimodal feature fusion mechanism that utilizes multimodal aggregation and style fusion blocks to fuse facial features in both latent and feature spaces.
- A novel latent warping algorithm automatically aligns facial poses between edited and exemplar images in latent space, without relying on annotated labels or pose detection models.
- Our novel framework would benefit numerous practical applications, supporting

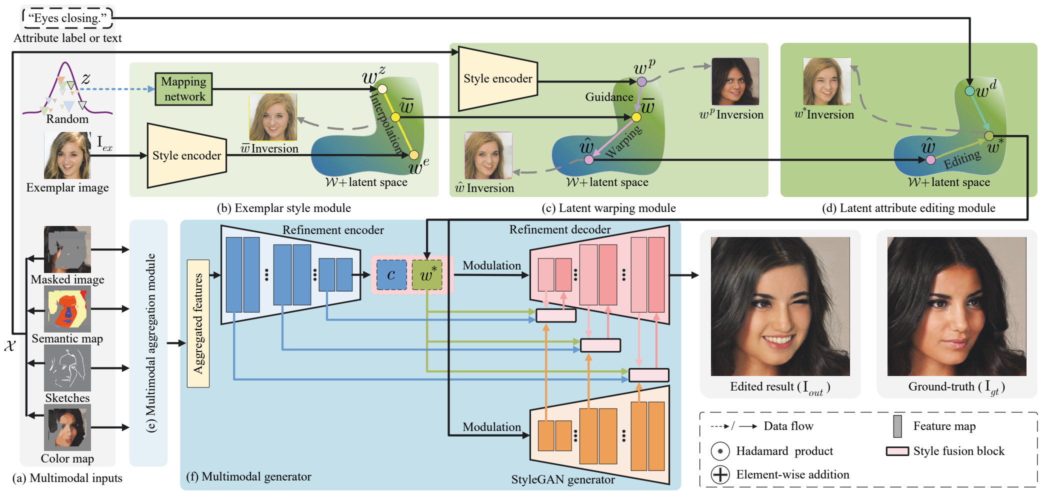


Figure 6.2: Overall pipeline of our FACEMUG globally-consistent local facial editing: the given attribute label (or text), random latent code z, and exemplar image $\mathbf{I}_{ex}$ are first processed through the exemplar style module, latent warping module, and the latent attribute editing module to get the edited latent codes. Simultaneously, the input pixel-wise multimodal inputs $\mathcal{X}$ and a binary mask $\mathbf{M}$ are fed into the multimodal aggregation module and the multimodal generator to get an edited realistic face image $\mathbf{I}_{out}$, where the manipulation of the masked regions in $\mathbf{M}$ is guided by multimodal inputs.

incremental editing scenarios guided by multimodities (sketches, semantic maps, color maps, exemplar images, text, and attribute labels).

6.2 Method

6.2.1 Overview

The overall editing pipeline of our FACEMUG framework is shown in Fig. 6.2. Given a collection of pixel-wise multimodal inputs $\mathcal{X} = \{\mathbf{I}_m, \mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_n\}$, an attribute label $w^d \in \mathcal{W}+$ (or text t_{tar}), an exemplar image $\mathbf{I}_{ex} \in \mathbb{R}^{h \times w \times 3}$, a ground-truth face image $\mathbf{I}_{gt} \in \mathbb{R}^{h \times w \times 3}$ (with $h \times w$ pixels and three color channels), and a binary mask $\mathbf{M} \in \mathbb{R}^{h \times w \times 1}$ (with 1 for editing and 0 for unedited pixels), the masked image $\mathbf{I}_m \in \mathbb{R}^{h \times w \times 3}$ is obtained by $\mathbf{I}_m = \mathbf{I}_{gt} \odot (\mathbf{1} - \mathbf{M})$, where each input pixel-wise modality $\mathbf{I}_k \in \mathbb{R}^{h \times w \times c_k}$ contains c_k channels and $\odot$ denotes the Hadamard product. Let $\mathcal{W}+$ denote the disentangled style latent space [60] and $\mathcal{Z}$ denote the random latent space. The goal of FACEMUG is to generate an edited realistic face image $\mathbf{I}_{out}$, where the manipulation of the masked regions in $\mathbf{M}$ is guided by multimodal inputs, while the unedited regions remain unchanged.

Exemplar style module. As shown in Fig. 6.2 (b), our exemplar style module is designed to support randomized and exemplar-guided facial attribute editing. First, a multi-layer fully-connected neural mapping network $F_{\hat{\theta}_f}$ with the network parameters $\hat{\theta}_f$ linearly maps a random latent code $z \in \mathbb{R}^{512 \times 1}$ ($z \in \mathcal{Z}$) to style latent codes $w^z = F_{\hat{\theta}_f}(z) = \{w_i^z \in \mathbb{R}^{512 \times 1} | i \in T\} \in \mathcal{W}+$, where $T = \{1, 2, ..., t\}$ and t is the number of the style latent code. Simultaneously, a style encoder $E_{\hat{\theta}_e}$ with the network parameters $\hat{\theta}_e$ maps multimodalities to $\mathcal{W}+$. Given an exemplar image $\mathbf{I}_{ex}$, the style encoder extracts exemplar latent codes $w^e = \{w_i^e \in \mathbb{R}^{512\times1} | i \in T\} = E_{\hat{\theta}_e}(\mathbf{I}_{ex}) \in \mathcal{W}+$. Then we perform style interpolation between w^z and w^e to get the interpolated latent codes $\overline{w} \in \mathcal{W}+$.

Latent warping module. As shown in Fig. 6.2 (c), to alleviate visual artifacts due to the misalignment between the facial poses of the exemplar image and the edited image, we introduce a latent warping module H_{θ_h} that learns to transfer the pose of a target image to a source image in the style latent space with the learnable network parameters θ_h. We first utilize the style encoder to project the pixel-wise multimodal inputs $\mathcal{X}$ to the projected latent codes $w^p = E_{\hat{\theta}_e}(\mathcal{X}) \in \mathcal{W}+$. Then we obtain the warped latent codes as $\hat{w} = H_{\theta_h}(w^p - \overline{w}, \overline{w}) + \overline{w}$, $\hat{w} \in \mathcal{W}+$, under the guidance of w^p. As a consequence, $\hat{w}$ aligns to the pose of projected latent codes w^p while preserving facial features of $\overline{w}$.

Latent attribute editing module. As shown in Fig. 6.2 (d), the latent attribute editing module is designed to edit the warped latent codes $\hat{w}$ to support attribute-conditional facial attribute editing and text-driven facial attribute editing in the style latent space. In this module, we obtain the edited latent codes w^*.

Multimodal aggregation module. As shown in Fig. 6.2 (e), for better controllability and visual quality in the image space, a multimodal aggregation module A_{θ_a} with the learnable network parameters θ_a is proposed to deal with multiple heterogeneous and sparse conditional inputs by merging them into a homogeneous feature space containing c_a feature channels. Given the pixel-wise multimodal inputs $\mathcal{X}$, we obtain the aggregated feature tensor $\hat{\mathbf{F}}^a = A_{\theta_a}(\mathcal{X}) \in \mathbb{R}^{h \times w \times c_a}$. $\hat{\mathbf{F}}^a$ is then input into our multimodal generator.

Multimodal generator. As shown in Fig. 6.2 (f), our multimodal generator consists of a facial feature bank and a refinement auto-encoder G_{θ_g} with the trainable network parameters θ_g. We implement the facial feature bank using the StyleGAN generator $S_{\hat{\theta}_s}$. $S_{\hat{\theta}_s}$ produces multi-scale coarse facial feature maps from edited latent codes w^* while G_{θ_g} refines the editing results by utilizing the aggregated feature tensor, the latent codes, and the generated coarse features. A set of facial feature maps $\mathcal{F}^s = \{\mathbf{F}_i^s \in \mathbb{R}^{\hat{h}_i \times \hat{w}_i \times \hat{c}_i} | i \in T\}$ and a reconstructed image $\mathbf{I}_p$ are obtained as $(\mathcal{F}^s, \mathbf{I}_p) = S_{\hat{\theta}_s}(w^*)$. We define $\hat{h}_i \times \hat{w}_i \times \hat{c}_i$ as the size of feature maps at i-th layer. Then, the refinement auto-encoder G_{θ_g} leverages $\hat{\mathbf{F}}^a$, w^*, and $\mathcal{F}^s$ to generate an edited image $\mathbf{I}_{out} \in \mathbb{R}^{h \times w \times 3}$:

$$\mathbf{I}_{out}(w^*) = \mathbf{I}_m \odot (\mathbf{1} - \mathbf{M}) + G_{\theta_g}(\hat{\mathbf{F}}^a, w^*, \mathcal{F}^s) \odot \mathbf{M}. \tag{6.1}$$

The refinement auto-encoder can be further divided into an encoder G^{en} and a decoder G^{de}, i. e., $G_{\theta_g} = \{G^{en}, G^{de}\}$.

Discriminator. A discriminative network D_{θ_d} with the learnable network parameters θ_d learns to judge whether an image is a real or fake image. The discriminator maps an image (e.g., $\mathbf{I}_{out}$ or $\mathbf{I}_{gt}$) to a scalar $D_{\theta_d}(\mathbf{I}) \in \mathbb{R}^{1 \times 1}$. Note that the discriminator is only applied during the training phase.

6.2.2 Latent warping module

To achieve realistic exemplar-guided or randomized facial attribute editing, the primary challenge lies in the potential differences in pose between the exemplar image and the edited image, which can readily result in noticeable misalignment between the two facial images. Several existing methods [25, 205] have been developed for facial pose alignment in the context of face reenactment. However, these methods often require additional pre-trained facial pose detection models or involve multi-stage training processes, which can be resource-intensive. We thus ask: Is it possible to directly conduct facial pose transfer in the latent space, thereby eliminating auxiliary steps such as preliminary GAN inversion, subsequent pose detection, and ultimately pose transfer?

To solve this, we propose a self-supervised latent warping method that eliminates the need for manual annotations and pre-trained facial pose detection models. Our method provides an intuitive and straightforward way of warping the pose of the exemplar image to match that of the edited image in the latent space while preserving the attributes of individual faces (e.g., identity or expressions). The key idea is that we use the pose of a target image to guide the pose of a source image by warping the latent codes in the style latent space.

To effectively predict the offsets in the latent space from given two latent codes for facial warping, we design our latent warping network H_{θ_h} by four stacked code-to-code modulation blocks. We construct our code-to-code modulation block by extending

FFCLIP's semantic modulation block [29] to code-to-code embeddings. Moreover, the sigmoid activation is incorporated in the semantic injection for gate activation.

Let $w^{ta} \in \mathbb{R}^{t \times 512 \times 1}$ and $w^{so} \in \mathbb{R}^{t \times 512 \times 1}$ be the target and source latent codes, respectively. The warped latent codes $w^{wa} \in \mathbb{R}^{t \times 512 \times 1}$ is obtained by warping w^{so} guided by w^{ta}:

$$w^{wa} = H_{\theta_h}(w^r, w^{so}) + w^{so}, \tag{6.2}$$

where $w^r = w^{ta} - w^{so}$ is the residual latent codes between w^{ta} and w^{so}. By leveraging the code-to-code modulation mechanism, our latent warping module effectively aligns the pose of the warped latent codes with the target latent codes.

6.2.3 Latent attribute editing module

Our method supports two types of latent attribute editing, including attribute-conditional facial attribute editing and text-driven facial attribute editing. It allows us to utilize conditional labels or text to manipulate latent codes for semantic-level editing with unedited portions unchanged, which is usually hard to achieve with GAN-inversion-based methods [17].

Attribute-conditional editing. Each attribute label corresponds to a semantic direction. For a user-specified target attribute label, we obtain the edited latent codes $w^* \in \mathcal{W}+$ by moving the warped latent codes $\hat{w}$ (w^{wa}) along the corresponding semantic direction w^d. The editing process can be expressed below [206]:

$$w^* = \hat{w} + \epsilon \cdot w^d, \tag{6.3}$$

where ϵ is a user-specified weight of the latent semantic direction (attribute label) $w^d \in \mathcal{W}+$ to control the degree of attribute adjustment.

Text-driven editing. For a user-specified target attribute text t_{tar}, we leverage CLIP [207] to find text-driven latent codes by solving the following latent codes optimization problem:

$$w^* = \underset{w \in \mathcal{W}+}{\arg\min} \left(\lambda_{clip} \cdot \mathcal{L}_{clip}(t_{tar}, t_{src}, w, \hat{w}) + \lambda_{reg} \cdot \|w - \hat{w}\|_2 \right), \tag{6.4}$$

where t_{src} = *"face"*; $\lambda_{clip} \in [0.1, 1.0]$ and λ_{reg} are used to balance the directional CLIP loss term and the regularization term. By default, we set $\lambda_{clip} = 0.05$ and $\lambda_{reg} = 0.08$. The directional CLIP loss $\mathcal{L}_{clip}$ [208] is employed to align directions between the text-image pairs of the original and edited images in the CLIP space:

$$\begin{aligned} \mathcal{L}_{clip}(t_{tar}, t_{src}, w, \hat{w}) = 1 - \cos\left(\Delta T, \Delta I\right), \\ \Delta T = E_T\left(t_{tar}\right) - E_T\left(t_{src}\right), \\ \Delta I = E_I\left(\mathbf{I}_{out}(w)\right) - E_I(\mathbf{I}_{out}(\hat{w})), \end{aligned} \tag{6.5}$$

where E_T and E_I are the text and image encoders of the CLIP model, $\mathbf{I}_{out}(w)$ and $\mathbf{I}_{out}(\hat{w})$ are obtained using Eq. 6.1.

6.2.4 Multimodal aggregation module

To integrate multimodalities within a unified framework, the varying density and value range among images, sketches, semantic maps, and color maps present challenges [209]. For pixel-wise multi-conditional image editing, the discrepancies in information content across modalities make it difficult to apply standard convolution layers to capture the diverse characteristics of each modality, leading to suboptimal generation quality. Moreover, the differing levels of details and densities in the inputs can impact the visual appearance and realism of the generated images. To mitigate this issue, we introduce a multimodal aggregation module that efficiently aggregates the multimodal inputs. We achieve this by using separate convolution layers for each modality and incorporating a normalized adaptive weighting mechanism to merge extracted features into a homogeneous feature space. As a result, the module provides more robust representations, making it well-suited for handling multi-conditional image editing tasks.

For the pixel-wise multimodal inputs $\mathcal{X}$, we first employ a residual block to extract feature maps for each modality, resulting in a feature set. Using this set, a shared residual block is utilized to compute the contribution scores for each spatial point across all pixel-wise modalities, producing a contribution score map for each modality. Each score map adaptively weights the importance of each modality in a pixel-wise fashion for the aggregation process. Thus, we can get the aggregated feature $\hat{\mathbf{F}}^a = A_{\theta_a}(\mathcal{X}) \in \mathbb{R}^{h \times w \times c_a}$. This adaptive weighting mechanism allows the model to assign higher importance to informative and detailed pixel-wise modalities while reducing the impact of less informative inputs.

6.2.5 Multimodal generator

To support multimodal conditional editing and generate high-quality editing results, we develop a multimodal generator that fully utilizes aggregated facial features and edited latent codes, while efficiently fusing feature maps from the refinement encoder, the facial feature bank, and the refinement decoder. To achieve this, we introduce a style fusion block to fuse features in both high-level and shallow-level feature spaces. These carefully designed modules enhance our approach's editing capabilities, enabling the generation of diverse and high-fidelity editing results.

Given the aggregated feature tensor $\hat{\mathbf{F}}^a$ and the edited latent codes w^*, the refinement encoder G_{en} extracts multi-scale feature maps $\mathcal{F}^{en} = \{\mathbf{F}_i^{en} \in \mathbb{R}^{\hat{h}_i \times \hat{w}_i \times \hat{c}_i} | i \in T\}$ and outputs a global latent vector $c \in \mathbb{R}^{512 \times 2}$ from the aggregated multimodal feature $\hat{\mathbf{F}}^a$, i.e., $(\mathcal{F}^{en}, c) = G_{en}(\hat{\mathbf{F}}^a)$. At the same time, the facial priors $\mathcal{F}^s = \{\mathbf{F}_i^s \in \mathbb{R}^{\hat{h}_i \times \hat{w}_i \times \hat{c}_i} | i \in T\}$ are extracted from the edited latent codes w^* in the facial feature bank with the StyleGAN generator. Finally, the feature maps $\mathcal{F}^{de} = \{\mathbf{F}_i^{de} \in \mathbb{R}^{\hat{h}_i \times \hat{w}_i \times \hat{c}_i} | i \in T\}$ in the refinement decoder G_{de} are calculated as follows:

$$\mathbf{F}_i^{de} = \begin{cases} \mathrm{UP}(\mathrm{SC}(\mathbf{F}_{i-1}^{de}, [w_i^*, c])), & \text{if } i \bmod 2 = 1; \\ \mathrm{SC}(\mathbf{F}_{i-1}^{g}, [w_i^*, c]), & \text{otherwise,} \end{cases} \tag{6.6}$$

where $[\cdot,\cdot]$ denotes concatenation; $\mathrm{UP}(\cdot)$ refers to up-sampling; $\mathrm{SC}(\cdot,\cdot)$ indicates the style layer [60], $\mathbf{F}_0^{de} = \mathrm{Conv}(\mathbf{F}_t^{en})$, $\mathrm{Conv}(\cdot)$ is a Convolution layer; $\mathbf{F}_j^{g}$ $(j = 1, 3, 5, \ldots, 2\lfloor \frac{t}{2} \rfloor + 1)$ is the fused feature map by the proposed style fusion block. We set $\mathbf{I}_{out} = \mathrm{Conv}(\mathbf{F}_t^{de}) \in \mathbb{R}^{h \times w \times 3}$ as the model output.

Style fusion block. The proposed style fusion block is as shown in Fig. 6.3. To fully leverage the guidance information extracted from the refinement auto-encoder and facial feature priors of the facial feature bank, we employ the gated fusion scheme to perform element-wise fusion between these features for enhancement. We first apply an adaptive gated fusion to activate features from the refinement encoder to obtain the intermediate generated feature $\hat{\mathbf{F}}_i^g$:

$$\hat{\mathbf{F}}_i^g = (\sigma(\mathrm{SC}(\mathbf{F}_i^{de}, w_{i+1}^*)) + \mathbf{1}) \odot \mathbf{F}_{t-i}^{en} + \phi(\mathrm{SC}(\mathbf{F}_i^{de}, w_{i+1}^*)), \tag{6.7}$$

where $\sigma(\cdot)$ denotes the sigmoid activation function and $\phi(\cdot)$ corresponds to the LeakyReLU activation function with the negative slope of 0.2. Then, the style layer computes the spatially-variant gate map $\mathbf{F}_i^m$ from the facial priors $\mathbf{F}_i^s$ and the modulated latent vector w_i^* for each i-th layer. The feature fusion is calculated as follows:

$$\begin{aligned} \mathbf{F}_i^m &= \sigma(\mathrm{SC}(\mathbf{F}_i^s, w_{i+1}^*)), \\ \mathbf{F}_i^g &= \mathbf{F}_i^m \odot \mathbf{F}_i^s + (\mathbf{1} - \mathbf{F}_i^m) \odot \hat{\mathbf{F}}_i^g, \end{aligned} \tag{6.8}$$

where the spatially-variant gating map $\mathbf{F}_i^m$ automatically selects the important features from generated feature maps $\hat{\mathbf{F}}_i^g$ and the facial priors $\mathbf{F}_i^s$.

6.2.6 Self-supervised module training

In our framework, we utilize the pre-trained StyleGAN generator $S_{\hat{\theta}_s}$ and mapping network $F_{\hat{\theta}_f}$ from StyleGAN-V2 [60]. We begin by the training of the style encoder $E_{\hat{\theta}_e}$. Next, we optimize the latent warping network H_{θ_h} with our proposed self-supervised warping learning. Finally, we detail the training process for the multimodal

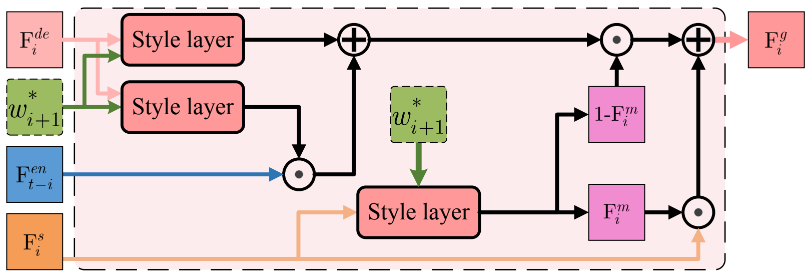


Figure 6.3: Illustration of our style fusion block. Conditioned by the modulated latent vector w^*_{i+1}, the block effectively integrates multi-scale facial features in both high-level and shallow-level feature spaces.

aggregation module A_{θ_a}, the refinement auto-encoder G_{θ_g}, and the discriminator D_{θ_d}. This training process can be in parallel with the optimization of the latent warping network.

Training of style encoder. The style encoder for exemplar images and that for multimodalities (sketch, color, semantic map, and mask) are the same encoder. The network of the style encoder $E_{\hat{\theta}_e}$ is borrowed from e4e [132]. In order to enable the encoder to project each modality individually into the latent space, we customized the first convolution layer of the e4e encoder to handle 26 channels from four modalities respectively: 3 channels for the exemplar image, 1 channel for sketches, 3 channels for colors, and 19 channels for semantic layouts. Then, we feed the concatenated randomly masked multimodal inputs into $E_{\hat{\theta}_e}$ for training. In addition, the loss functions defined in e4e [18, 132] are employed.

Training of latent warping module. As shown in Fig. 6.4, a triplet of initial, source, and target latent codes is utilized to learn the pose warping guided

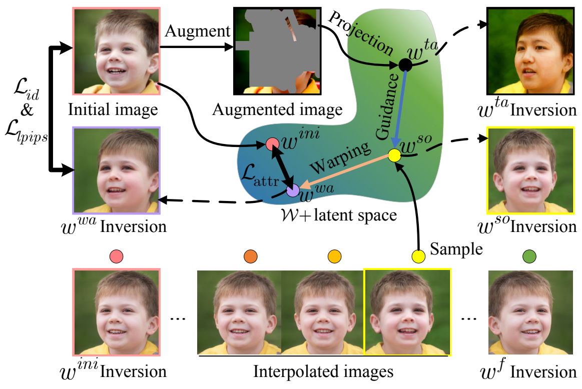


Figure 6.4: Illustration of the self-supervised training of our latent warping module. We employ the style encoder to project the augmented image to obtain the target latent codes w^{ta}. The source latent codes w^{so} are sampled using interpolation between the initial latent codes w^{ini} and the flipped latent codes w^{f}. The identity loss, the LPIPS loss, and the attribute loss are utilized as constraints to disentangle the identity and pose. This module effectively transfers the pose of w^{ta} to the warped latent codes w^{wa} while remaining other facial features unchanged. The inversion process is utilized for the visualization purpose.

by the target latent codes. We begin by projecting an initial image $\mathbf{I}_{ini}$ into the style latent space to get the initial latent codes $w^{ini} = E_{\hat{\theta}_e}(\mathbf{I}_{ini})$. Then, we obtain an augmented image $\mathbf{I}_{ta}$ by applying bilinear scaling, color jittering, and region masking operations [184] to $\mathbf{I}_{ini}$. In addition, we obtain a flipped image $\mathbf{I}_f$ with mirror flipping of $\mathbf{I}_{ini}$. Therefore, $\mathbf{I}_f$ shares the same identity as $\mathbf{I}_{ini}$ with a flipped pose. Then, the target latent codes w^{ta} and the flipped latent codes w^{f} can be obtained by $w^{ta} = E_{\hat{\theta}_e}(\mathbf{I}_{ta})$ and $w^{f} = E_{\hat{\theta}_e}(\mathbf{I}_f)$, respectively. Next, we obtain the source latent codes w^{so}

with a linear interpolation:

$$w^{so} = \beta \cdot w^{ini} + (1 - \beta) \cdot w^{f}, \tag{6.9}$$

where $\beta \in [0, 1]$ is a uniform random number. Since w^{ini} and w^{f} share the same facial features except for the pose, the interpolated source codes w^{so} maintain the same facial identity to w^{ini} but have a different pose. The warped latent codes w^{wa} are obtained by transferring the pose of w^{ta} to w^{so} while maintaining the identity of w^{so}, using our latent warping network H_{θ_h}. Finally, we obtain the warped image as $(\mathcal{F}^{wa}, \mathbf{I}_{wa}) = S_{\hat{\theta}_s}(w^{wa})$.

Total loss. In order to disentangle the identity and pose during warping, we train H_{θ_h} by utilizing the identity loss, the LPIPS loss, and the attribute loss to constrain the identity and attribute similarities between w^{wa} and w^{ini}. The total training loss of H_{θ_h} is defined as:

$$O(\theta_h) = \lambda_{latent} \cdot (\mathcal{L}_{id}(\mathbf{I}_{wa}, \mathbf{I}_{ini}) + \mathcal{L}_{lpips}(\mathbf{I}_{wa}, \mathbf{I}_{ini}) + \mathcal{L}_{attr}(w^{wa}, w^{ini})), \tag{6.10}$$

where λ_{latent} is empirically set to 0.1 in this work; $\mathcal{L}_{id}$, $\mathcal{L}_{lpips}$, and $\mathcal{L}_{attr}$ are the identity loss, the LPIPS loss, and the attribute loss, respectively, and are defined below.

Identity loss. The identity loss [27, 29] is incorporated to constrain the identity similarity:

$$\mathcal{L}_{id}(\mathbf{I}_x, \mathbf{I}_y) = 1 - \cos(R(\mathbf{I}_x), R(\mathbf{I}_y)), \tag{6.11}$$

where $R(\cdot)$ is a pre-trained ArcFace network [195].

LPIPS loss. The Learned Perceptual Image Patch Similarity (LPIPS) [196] is applied to constrain the perceptual similarity:

$$\mathcal{L}_{lpips}(\mathbf{I}_x, \mathbf{I}_y) = \|P(\mathbf{I}_x) - P(\mathbf{I}_y)\|_2, \tag{6.12}$$

where $P(\cdot)$ is a pre-trained VGG feature extractor [149].

Attribute loss. The attribute loss [27, 210] is used to constrain the learning in the style latent space:

$$\mathcal{L}_{attr}(w^x, w^y) = \|w^x - w^y\|_2. \tag{6.13}$$

Consequently, we can obtain the optimized parameters θ_h^* via the minimization of $O(\theta_h)$. By taking advantage of the code-to-code modulation mechanism of H_{θ_h} with the loss constraints, w^{wa} effectively learns the pose from w^{ta} while remaining other facial features (e.g., identity) of w^{so} unchanged.

Training of multimodal aggregation module, refinement auto-encoder and discriminator. The identity loss, the LPIPS loss, the diversity-enhanced attribute loss, and the adversarial loss are combined to optimize the multimodal aggregation module, the refinement auto-encoder, and the discriminator.

To learn the mapping between style latent codes and corresponding facial attributes, and to support attribute-conditional editing in the style latent space, a diversity-enhanced attribute loss $\mathcal{L}_{attr}(w^o, \overline{w})$ is employed to constrain the consistency between facial attributes of the edited image $\mathbf{I}_{out}$ and the interpolated latent codes $\overline{w}$, where $w^o = E_{\hat{\theta}_e}(\mathbf{I}_{out})$.

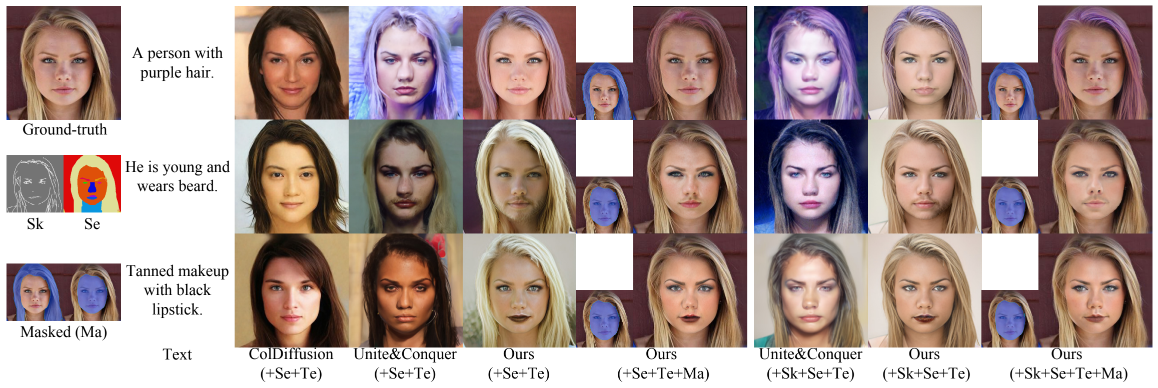


Figure 6.5: Visual comparison to ColDiffusion [12] and Unite&Conquer [13] for text-driven multimodal facial editing. Our method produces visually appealing and globally consistent images with good responses to the corresponding multimodal inputs, and remains unmasked parts unchanged.

Total loss. The total training loss is defined as:

$$\begin{aligned} O(\theta_a, \theta_g, \theta_d) = \lambda_{id}\mathcal{L}_{id}(\mathbf{I}_{out}, \mathbf{I}_{ex}) + \lambda_{attr}\mathcal{L}_{attr}(w^o, \overline{w}) \\ + \lambda_{lpips}\mathcal{L}_{lpips}(\mathbf{I}_{out}, \mathbf{I}_{gt}) + \mathcal{L}_{adv}(\mathbf{I}_{out}, \mathbf{I}_{gt}), \end{aligned} \tag{6.14}$$

where we empirically set $\lambda_{id} = 0.1$, $\lambda_{lpips} = 0.5$, and $\lambda_{attr} = 0.1$ in this work; $\mathcal{L}_{adv}$ is the adversarial non-saturating logistic loss [41] with R_1 regularization [151].

The refinement network G_{θ_g} is trained to generate a realistic edited image $\mathbf{I}_{out}$ while the discrinimator D_{θ_g} tries to differentiate between $\mathbf{I}_{gt}$ and $\mathbf{I}_{out}$. In an alternating fashion, A_{θ_a} and G_{θ_g} are trained in a phase while D_{θ_d} is trained in the other. For each iteration, we obtain the optimized parameters θ_a^*, θ_g^* and θ_d^* via the minimax game iteratively.

Implementation without manual annotation. Our framework was implemented using Python and PyTorch. We trained FACEMUG and our latent warping module independently. For more implementation details of the latent warping module, the multimodal aggregation module, and the training procedures of our networks, please refer to the supplementary document.

Manual annotations of labeled paired data across different modalities are required in existing multimodal editing methods [52, 14]. On the contrary, the training modalities (i.e., editing mask, exemplar, semantics, sketch, and color) in this paper were generated without any manual annotation using well-built methods. The trained masks were generated randomly with the mask generation algorithm from CMOD [49]. The Face-parsing model [211] was used to extract semantic maps. Hand-drawn-like sketches were generated using a pencil-sketch filter [18]. Color images were processed using a mean color of each semantic region. Exemplar images were sampled randomly from the ground-truth images. By aligning all modalities into a unified generative latent space, FACEMUG effectively loosens the ties between the paired modalities and enables model training without any human annotation.

The latent attribute editing module was implemented as follows. For the attribute-conditional editing, attribute labels employed in InterfaceGAN [134], GANSpace [137], StyleCLIP [138], and CLIP2StyleGAN [135] were integrated in the module for semantic direction. For the text-driven editing, the edited latent codes were obtained through 100 ~ 300 iterations of gradient descent [138] with the learning rate of 0.1.

Following the settings of StyleGANv2 [60], we employed the Adam optimizer with the first momentum coefficient of 0.5, the second momentum coefficient of 0.99, and the learning rate of 0.002. We trained the networks for 800,000 iterations with a batch size of 8.

6.3 Experimental results and comparisons

6.3.1 Settings

We conducted experimental evaluations on two publicly available and commonly used benchmark face image datasets: CelebA-HQ [154] and FFHQ [42]. Our FACEMUG was trained on the training set of FFHQ and evaluated on the testing set of CelebA-HQ and FFHQ, respectively. In the CelebA-HQ dataset, $2,000$ images were randomly selected for testing. For the FFHQ dataset, $60,000$ images were randomly chosen for training, and the remaining $10,000$ images were used for testing. All images were resized to the resolution of 256×256. To ensure a fair comparison, the same training and testing splits were used for all experiments.

All experiments were conducted on the NVIDIA Tesla V100 GPU. The training time of our FACEMUG was around one month. We also evaluated FACEMUG on a PC equipped with an NVIDIA GeForce RTX 4090 GPU. It takes 29 ms (34 FPS) for each inference.

We quantitatively evaluated the performance by using the Fréchet inception distance (FID) [155], the unpaired inception discriminative score (U-IDS) [49], and

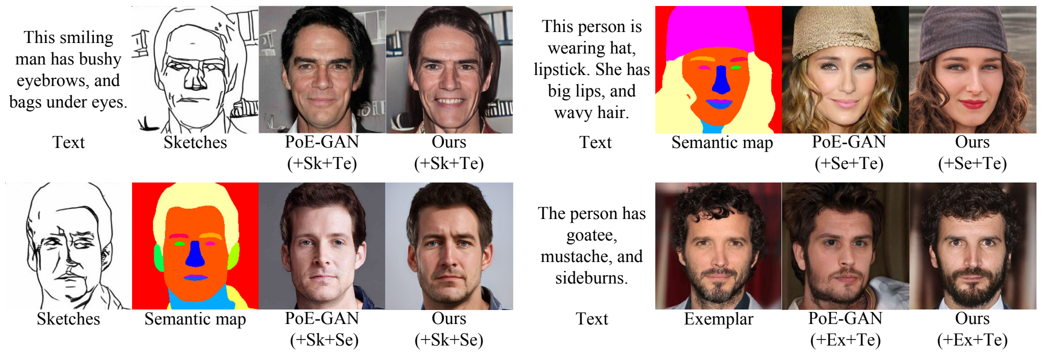


Figure 6.6: Visual comparison to PoE-GAN [14]. We used the input modalities and results published in their paper. The consistency between generated images and multimodal inputs of FACEMUG is better than that of PoE-GAN.

LPIPS [196] metrics which are robust assessment measures and correlate well with human perception for the image quality.

We demonstrated the performance of FACEMUG by utilizing various multimodal inputs. For convenience, we used the following abbreviations for these multimodalities: "+Sk" for adding sketches, "+Se" for adding semantic maps, "+Co" for adding color information, "+Ex" for adding an exemplar image, "+Te" for adding text. We used the notation "+Ma" to represent the inclusion of a masked image for local editing. Different masks may affect the quantitative results because of the variation in position, size, and shape. We included various types of masks for comprehensive quantitative evaluation. Each mask was selected randomly from one of the following types of masks: hair, face, foreground subject, irregular region (50 – 60% mask ratio), and a fixed center (128 × 128) rectangle. The center mask was included because it effectively covers most of the facial region in a facial image. For quantitative comparisons, masks were obtained automatically. Semantics-based masks (hair, face, etc.) were created with the Face-parsing model [211]. Irregular masks were obtained from the irregular mask templates [101]. For qualitative comparisons, masks were manually defined. A consistent set of inputs was used in each comparison to ensure fairness, both quantitatively and qualitatively. For more experimental results, please refer to the supplementary document.

6.3.2 Comparison on multimodal facial editing

To evaluate the quality of generated images and the responsiveness of multimodal inputs, we performed comparisons against SOTA multimodal facial image editing techniques, including Unite&Conquer [13], Collaborative-Diffusion (ColDiffusion) [12], and PoE-GAN [14] using multimodal conditional inputs. Unless specified, we employed officially released pre-trained models of compared methods.

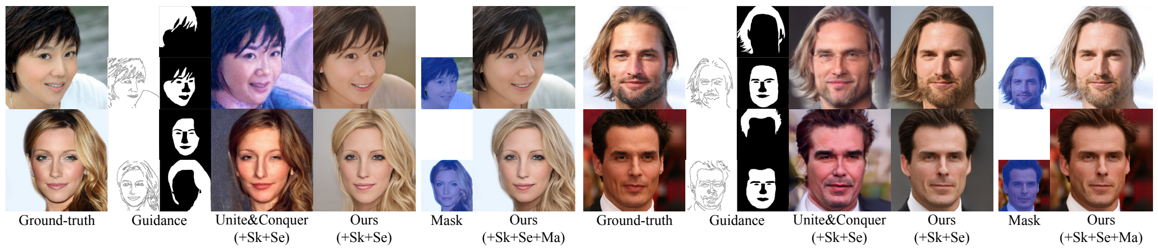


Figure 6.7: Visual comparison to Unite&Conquer [13]. We show more FACEMUG results by adding extra masks. Our method shows better visual quality and preserves background information when using masks.

Table 6.1: Quantitative comparison of our method to SOTA multimodal facial editing methods with sketches, semantic maps, and text on the CelebA-HQ dataset. **Bold**: top-1 quantity.

Method	FID$\downarrow$	U-IDS$\uparrow$	LPIPS$\downarrow$
ColDiffusion [12] (+Se+Te)	26.87	0	0.5283
Unite&Conquer [13] (+Se+Te)	44.52	0	0.5809
Ours (+Se+Te)	38.41	0	0.4699
Ours (+Se+Te+Ma)	**11.85**	**0.20%**	**0.1645**
Unite&Conquer [13] (+Sk+Se+Te)	45.29	0	0.5493
Ours (+Sk+Se+Te)	32.65	0	0.4046
Ours (+Sk+Se+Te+Ma)	**11.24**	**0.28%**	**0.1448**

Fig. 6.5 shows visual comparisons of our FACEMUG to Unite&Conquer and ColDiffusion. It shows that all the compared methods can use semantics or sketches to control the facial layout while adjusting appearance using given text. However, Unite&Conquer and ColDiffusion show low consistency between the output image and the text caption. In contrast, our method is capable of combining the three modalities to perform high-quality multimodal editing. Fig. 6.6 shows visual comparisons of our FACEMUG to PoE-GAN using text with sketches, semantics, and exemplar, respectively. The results of PoE-GAN show high-quality and good responses to the corresponding input modalities. However, for the first case, our method shows a clear

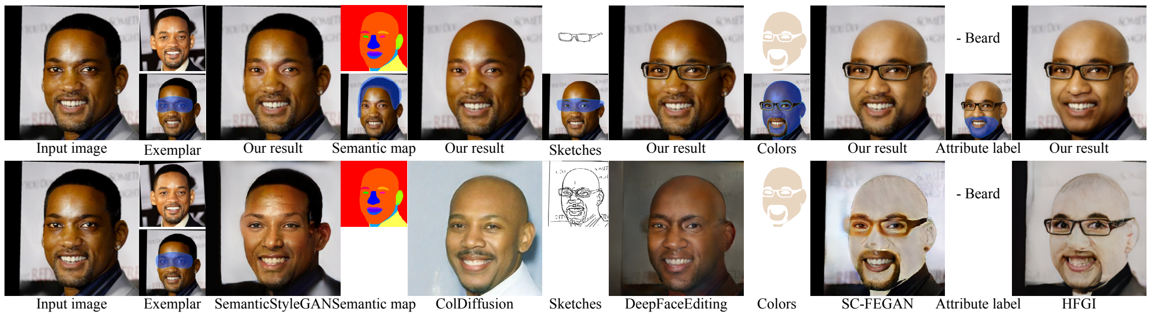


Figure 6.8: Qualitative comparison of incremental editing (the later editing taking the previous output image as input), compared to SOTA methods (SemanticStyle-GAN [6], ColDiffusion [12], DeepFaceEditing [15], SC-FEGAN [16], HFGI [17]) in row 2, using uni-modality. Our model shows better visual quality.

smile expression, and for the last case, the edited result exhibits more correlation with the exemplar. For FACEMUG, the semantic maps and sketches provide geometry information, while text controls the appearance of the generated content. Moreover, by leveraging the additional masked image, the generated content exhibits high consistency to the input modalities and is coherent to the editing boundary, preserving the unedited part unchanged. We also show the quantitative comparison of multimodal editing with ColDiffusion [12] and Unite&Conquer [13] on CelebA-HQ dataset, as shown in Table 6.1. The semantic maps and text are from CelebAMask-HQ [123] and CelebA-Dialog [212] datasets, respectively. Guided by semantic maps and text, both ColDiffusion and Unite&Conquer produce high-quality images yet exhibit lower consistency with ground-truth images, as reflected by their LPIPS scores. In contrast, FACEMUG achieves lower LPIPS and competitive FID scores, indicating good fidelity

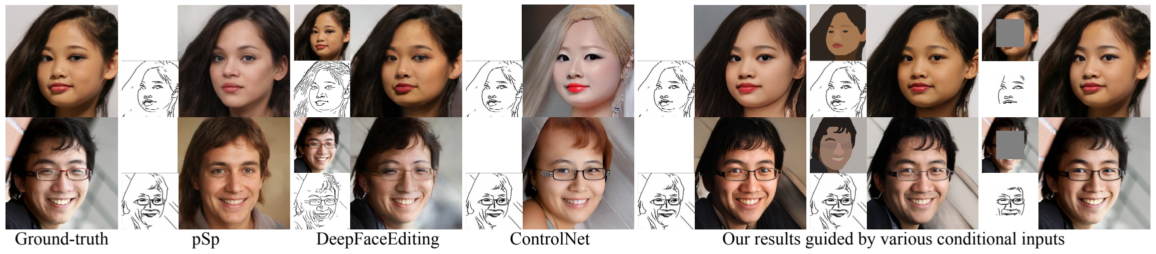


Figure 6.9: Visual comparison between our FACEMUG and the SOTA sketch-guided editing methods (pSp [18], DeepFaceEditing [15], and ControlNet [19]). The sub-images in each group represent the guidance information for the editing process. FACEMUG produces images with superior quality and finer details by using more modalities, and shows global consistency when adding masks for local facial editing.

of our results. When applying a mask to indicate the editing regions, FACEMUG not only demonstrates improved performance of FID and LPIPS scores but also illustrates superior editing quality. Our method achieves the lowest FID scores because it can modify specific facial attributes within the masked area while maintaining the unmasked regions unchanged and enhancing the overall image consistency.

We further conducted comparisons between FACEMUG and Unite&Conquer [13], focusing on facial editing using sketches and semantics. For a fair comparison, given that Unite&Conquer utilizes only the “skin” and “hair” semantic maps, we accordingly adjusted FACEMUG by removing other semantic labels. As shown in Fig. 6.7, Unite&Conquer unites multiple diffusion models trained on multiple sub-tasks to perform editing with the guidance of sketch and partial semantic labels. However, the presence of visual artifacts in the edited images indicates that employing disparate off-

Table 6.2: Quantitative comparison to SOTA multimodal facial image editing methods with sketches and semantic maps on CelebA-HQ and FFHQ datasets. **Bold**: top-1 quantity.

Method	CelebA-HQ			FFHQ		
	FID↓	U-IDS↑	LPIPS↓	FID↓	U-IDS↑	LPIPS↓
Unite&Conquer [13] (+Sk+Se)	44.76	0	0.5350	52.39	0	0.5707
Ours (+Sk+Se)	29.96	0	0.3999	23.16	0.03%	0.4036
Ours (+Sk+Se+Ma)	**10.36**	**0.63%**	**0.1371**	**2.26**	**30.38%**	**0.1162**

the-shelf diffusion models, trained on different datasets, remains a challenging task. In comparison, FACEMUG adeptly leverages sketches and partial semantic labels to facilitate geometry-guided facial generation (+Sk+Se) and editing (+Sk+Se+Ma). Moreover, the generated contents maintains consistency with the unedited portions. Table 6.2 displays the FID, U-IDS, and LPIPS scores of each method on CelebA-HQ and FFHQ datasets. Unite&Conquer introduces a novel reliability parameter to facilitate the multimodal mixing of contents generated from various uni-modal diffusion networks. Nevertheless, our method surpasses Unite&Conquer, exhibiting superior FID, U-IDS, and LPIPS scores.

Fig. 6.1 and Fig. 6.8 show examples of incremental multimodal local facial editing. Our FACEMUG incrementally edits the input facial images to achieve high-quality manipulation by taking advantage of multimodal inputs, including masks, exemplars,

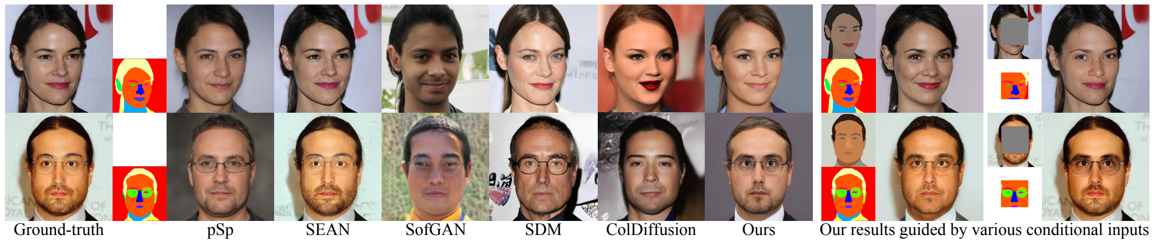


Figure 6.10: Visual comparison between our FACEMUG and the SOTA semantic-guided editing methods (pSp [18], SEAN [20], SofGAN [21], SDM [22], and ColDiffusion [12]). The sub-images in each group represent the guidance information for the editing process. FACEMUG produces more visually appealing results using more conditional modalities, and achieves high-quality local facial editing by incorporating masks.

semantics, sketches, colors, and attribute labels. It is worth noting that incremental editing is achieved effectively with our unified FACEMUG model, while existing methods have to use multiple uni-modal models (see Fig. 6.8) and introduce unwanted modifications on unedited facial features (see Fig. 6.1). By leveraging provided multimodalities, our FACEMUG can edit various realistic facial attributes (e.g., facial geometry, hairstyle, and decorative goods) while preserving unedited parts unchanged.

User study. We conducted a user study for multimodal facial editing comparing ColDiffusion and Unite&Conquer with "Se+Te" and "+Sk+Se+Te" configurations. We sampled 100 images randomly from the CelebA-HQ test set and obtained corresponding edited images. 20 pairs of edited images were chosen randomly for each method-to-method comparison. For FACEMUG, ten edited images were produced

Table 6.3: The user study results on the CelebA-HQ dataset. We present the percentages (%) of cases where our results were preferred over those of the compared methods.

Configuration	Method-to-method comparison	Percentage
+Se+Te	Ours vs. ColDiffusion [12]	54.90%
+Se+Te	Ours vs. Unite&Conquer [13]	76.83%
+Se+Sk+Te	Ours vs. Unite&Conquer [13]	77.88%

with additional masks (+Ma), which were selected randomly from hair, face, and foreground subject masks; the other ten images were produced without masks. The participants were asked to perform two-alternative forced choices (2AFCs) based on visual realism and consistency with input modalities. Finally, we recruited 52 participants, resulting in 1040 votes per comparison.

Table 6.3 shows the user study results. With the "+Se+Te" configuration, our FACEMUG received 54.90% and 76.83% of the preference votes compared to ColDiffusion (45.10%) and Unite&Conquer (23.17%), respectively. Our FACEMUG also surpassed Unite&Conquer with 77.88% of majority votes with the "+Sk+Se+Te" configuration. The user study validated that our FACEMUG effectively produces realistic facial images by taking advantage of multimodal local facial editing.

Table 6.4: Quantitative comparison to SOTA sketch-guided facial image editing methods (+Sk) on CelebA-HQ and FFHQ datasets. **Bold**: top-1 quantity.

Method	CelebA-HQ			FFHQ		
	FID↓	U-IDS↑	LPIPS↓	FID↓	U-IDS↑	LPIPS↓
TediGAN [52]	42.83	0	0.4909	85.17	0	0.6031
pSp [18]	42.70	0	0.4911	85.16	0	0.6031
DeepFaceEditing [15]	19.78	0	0.2796	11.59	5.13%	0.2850
ControlNet [19]	64.59	0	0.5530	62.31	0	0.5475
Ours (+Sk)	36.58	0	0.4071	24.29	0	0.4096
Ours (+Sk+Co)	17.43	0	0.2936	9.12	9.59%	0.2815
Ours (+Sk+Ma)	**11.43**	**0.52%**	**0.1444**	**2.49**	**29.51%**	**0.1259**

6.3.3 Comparison on sketch-guided facial editing

We compared FACEMUG to the SOTA sketch-guided facial editing methods, including pSp [18], DeepFaceEditing [15] and ControlNet [19]. The pre-trained models of pSp and DeepFaceEditing provided in the official online repository were used in this experiment. For ControlNet, we fine-tuned the officially provided pre-trained weights on the FFHQ training set with corresponding sketches. To ensure optimal performance, we used default forms of sketches that were used during training for each method.

As depicted in Fig. 6.9, all the compared methods are capable of generating high-quality images guided by sketches. However, pSp may neglect some facial attributes, like glasses from input sketches. Even conditioned on facial appearance and sketches, DeepFaceEditing exhibits some visual artifacts in edited outputs. While ControlNet can generate high-quality results, it needs several denoising steps and depends on high-quality prompts. Without colors and unmasked background information (third last column), the color richness generated by FACEMUG may be affected. By incorporating both colors and sketches as guidance (second last column), FACEMUG can produce facial images with high fidelity. Our method (last column) demonstrates sketch-guided local facial editing, showcasing our method's ability to achieve high-quality facial image editing by leveraging unmasked pixels and input sketches while preserving the unedited regions unchanged.

Table 6.4 presents a quantitative comparison between FACEMUG and the existing SOTA approaches. All methods demonstrate good FID scores on both datasets. DeepFaceEditing achieves competitive FID and U-IDS scores by utilizing sketch and appearance information extracted from ground-truth images. Since sketches lack color and appearance information, the performance of FACEMUG (+Sk) may be limited. FACEMUG (+Sk+Co), which incorporates color as guidance information, further enhances the visual quality. FACEMUG (+Sk+Ma) surpasses all the compared methods by leveraging background information, demonstrating superior performance.

Table 6.5: Quantitative comparison to SOTA semantic-guided facial image editing methods (+Se) on CelebA-HQ and FFHQ datasets. **Bold**: top-1 quantity.

Method	CelebA-HQ			FFHQ		
	FID$^{\downarrow}$	U-IDS$^{\uparrow}$	LPIPS$^{\downarrow}$	FID$^{\downarrow}$	U-IDS$^{\uparrow}$	LPIPS$^{\downarrow}$
TediGAN [52]	45.19	0	0.5208	85.50	0	0.6109
pSp [18]	45.19	0	0.5208	85.50	0	0.6109
SEAN [20]	28.74	0	0.2595	26.55	1.5%	0.3801
SofGAN [21]	38.88	0	0.6369	15.43	8.14%	0.6297
SDM [22]	27.61	0	0.4995	-	-	-
ColDiffusion [12]	30.22	0	0.5280	76.73	0	0.5937
Ours (+Se)	41.18	0	0.4462	43.78	0	0.4536
Ours (+Se+Co)	31.02	0	0.3691	14.08	8.09%	0.3018
Ours (+Se+Ma)	**10.65**	**0.53%**	**0.1570**	**2.41**	**28.99%**	**0.1356**

6.3.4 Comparison on semantic-guided facial editing

We compared FACEMUG to the SOTA semantic-guided facial editing methods, including pSp [18], SEAN [20], SofGAN [21], SDM [22], and ColDiffusion [12]. The pre-trained models of the compared methods provided in the official online repository were used in this experiment.

As shown in Fig. 6.10, all methods can generate high-quality images guided by semantic maps. However, pSp and ColDiffusion may not accurately capture certain attributes (e.g., the shape of the glasses). SofGAN may produce facial images with less

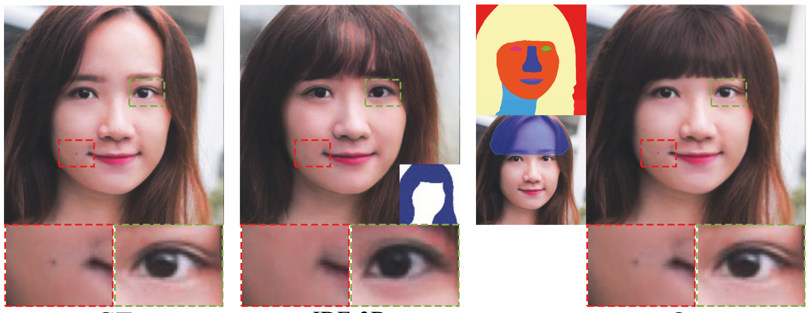
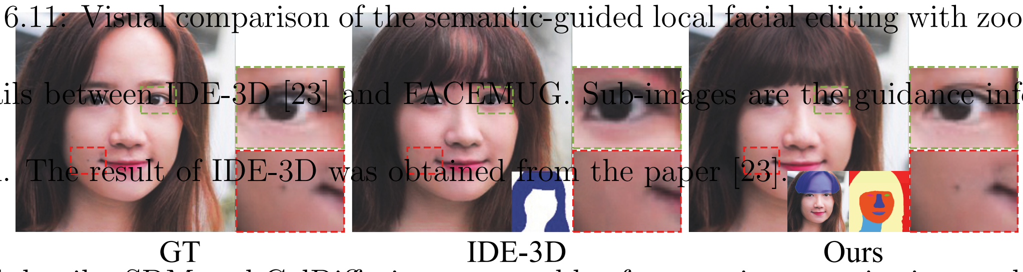


Figure 6.11: Visual comparison of the semantic-guided local facial editing with zoom-in details between IDE-3D [23] and FACEMUG. Sub-images are the guidance information. The result of IDE-3D was obtained from the paper [23].



defined details. SDM and ColDiffusion are capable of generating superior images but at the cost of longer computation time. Additionally, above compared methods fail to preserve the unedited regions surrounding the editing area. In contrast, FACEMUG (third last column) conditioned by semantic maps exhibits clear manifestations of semantic maps. When guided by color information as well (second last column), our method demonstrates superior visual performance. Furthermore, FACEMUG (last column) allows for local facial editing guided by semantic maps, preserving known regions while providing more flexibility for interactive facial attribute manipulation. FACEMUG consistently produces high-quality results with facial attributes guided by semantic maps.

Table 6.5 presents the quantitative performance of the compared methods. SDM and ColDiffusion achieve good FID scores when conditioned by semantic maps. Without colors and unmasked pixels, the performance of FACEMUG (+Se) is limited. In-

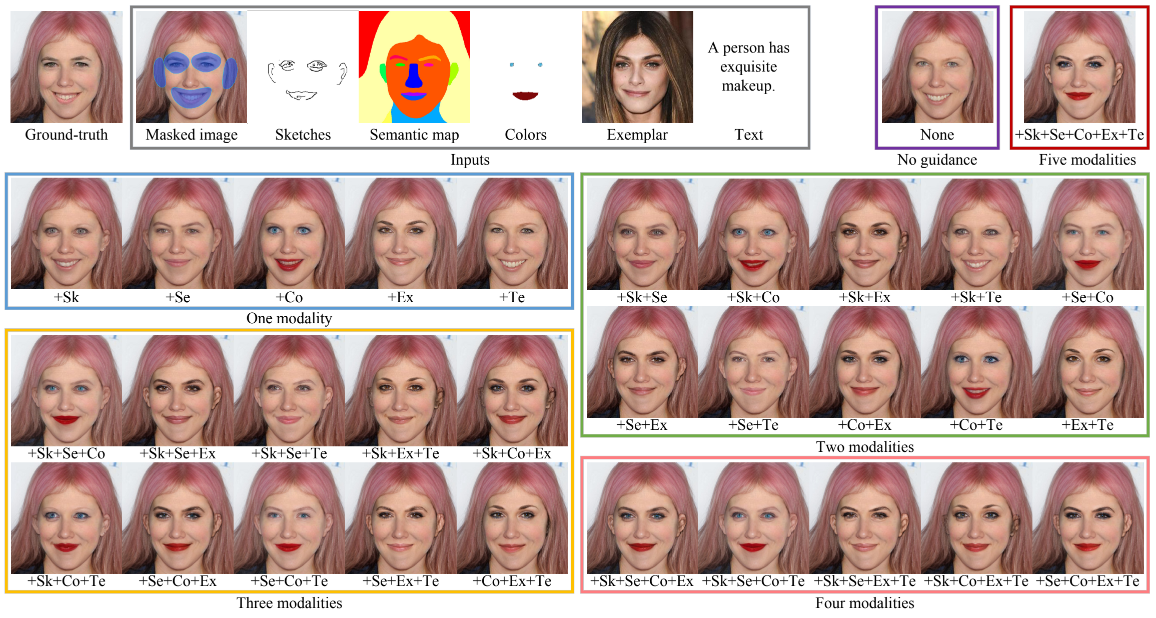


Figure 6.12: Visual performance of our FACEMUG with various modal inputs. The masked image was utilized for all outputs. There are a total of 32 combinations (subsets) of five modalities. Our FACEMUG generates visually appealing results and shows high global consistency to the unedited regions.

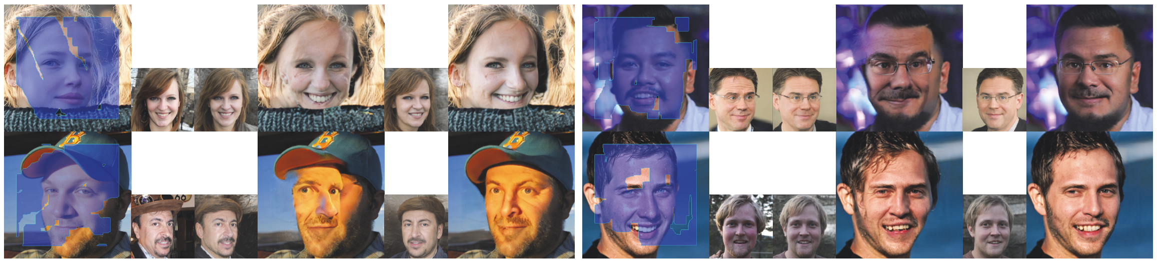


Figure 6.13: Visual comparison of the ablation study on the latent warping module (from left to right in each group): the masked image, the exemplar image, inversion of latent codes of the exemplar image, the editing result without the latent warping module (+Ex w/o warping), inversion of warped latent codes of the exemplar image, and the editing result with the latent warping module (+Ex). Our warping module improves the visual quality when the pose misalignment happens between the edited image and the exemplar.

corporating colors significantly improves the performance of FACEMUG (+Se+Co). With the ability to edit masked regions while preserving unmasked pixels, FACEMUG (+Se+Ma) achieves the best FID, U-IDS, and LPIPS scores for both datasets. FACEMUG outperforms the compared models in terms of the authenticity of locally edited facial images guided by semantic maps.

Fig. 6.11 shows the visual comparison between FACEMUG and the SOTA 3D-aware facial editing method, IDE-3D [23]. Both methods achieve high-quality editing. IDE-3D supports 3D face synthesis but may result in some losses in facial details during GAN inversion, such as moles and eyelids. In comparison, FACEMUG excels at preserving unedited facial attributes but faces challenges in reconstructing 3D

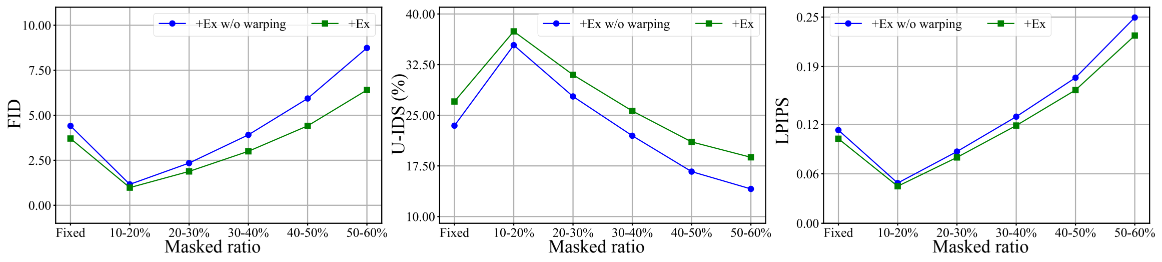


Figure 6.14: Quantitative comparison of the ablation study on the latent warping module on the FFHQ dataset. "+Ex w/o warping": the exemplar image without the latent warping module used; "+Ex": the exemplar image with the latent warping module used. Other modalities were not used.

6.3.5 Ablation study

Ablation study on multimodal inputs. We also conducted experiments to evaluate the effects of multimodal inputs on our FACEMUG framework. Fig. 6.12 showcases editing examples of FACEMUG using a total of 32 input configurations of five modalities. Generally, when multimodalities are consistent with each other, incorporating more modalities achieves much better visual quality. "None" means that only the masked image is utilized for image inpainting. When using sketches or semantic maps, FACEMUG clearly exhibits the structure of inputs. Incorporating colors enhances texture details and produces faithful edited results. Example images further help transfer the styles and facial identity to the edited regions. With the guidance of the text, more detailed facial attributes can be manipulated. Utilizing all modalities (five modalities) information allows FACEMUG to achieve the best overall performance. It also demonstrates that our method can work well on all the subsets of five modalities.

Ablation study on latent warping module. We conducted a study to assess the effectiveness of our latent warping module. Various irregular masks with different mask ratios [101] and a fixed center 25% (128×128) rectangular mask were used to simulate different editing situations. We show visual examples in Fig. 6.13. We can find that our latent warping module effectively aligns the exemplar pose to the edited image while preserving the facial attributes and identity of the exemplar images. As shown in Fig. 6.14, the quantitative performance results demonstrate that our latent warping module achieves better quantitative performance (+Ex) compared to not using it (+Ex w/o warping). Since the edited and exemplar images contain different poses, directly transferring features to target regions may cause obvious artifacts. Our latent warping module mitigates this issue by adapting the exemplar pose to align with the edited image in the style latent space, thereby avoiding the boundary issue during editing.

Ablation study on other main components. Here, we explored the image generation performance of other main components by comparing FACEMUG to its five variants on the FFHQ dataset with $50 - 60\%$ mask ratios, as shown in Table 6.6.

Table 6.6: Ablation study of the image generation performance of other main components on the FFHQ dataset with 50 – 60% mask ratios. (A) represents our full model (FACEMUG). (B) replaces our style fusion blocks with element-wise addition operations. (C) replaces our style fusion blocks with gated convolution blocks [26]. (D) replaces our diversity-enhanced attribute loss with the attribute loss [27]. (E) removes the facial feature bank. **Bold**: top-1 quantity.

Method	FID↓	U-IDS↑	LPIPS↓
A	**3.274**	**28.14%**	**0.1917**
B	3.829	24.05%	0.2010
C	3.478	26.70%	0.1954
D	3.599	26.44%	0.1991
E	3.725	25.26%	0.1971

We tested to replace our fusion blocks with element-wise addition operations (B) and gated convolution blocks [26] (C). The quantitative performance of models (B) and (C) exhibited a certain degree of decline, compared to our full model (A). When replacing our diversity-enhanced attribute loss with the attribute loss [27] (D), the quantitative measures also show a decrease. We also show FACEMUG's effectiveness by removing the facial feature bank (E). The quantitative scores dropped significantly, underscoring the importance of the facial feature bank for high-quality results.

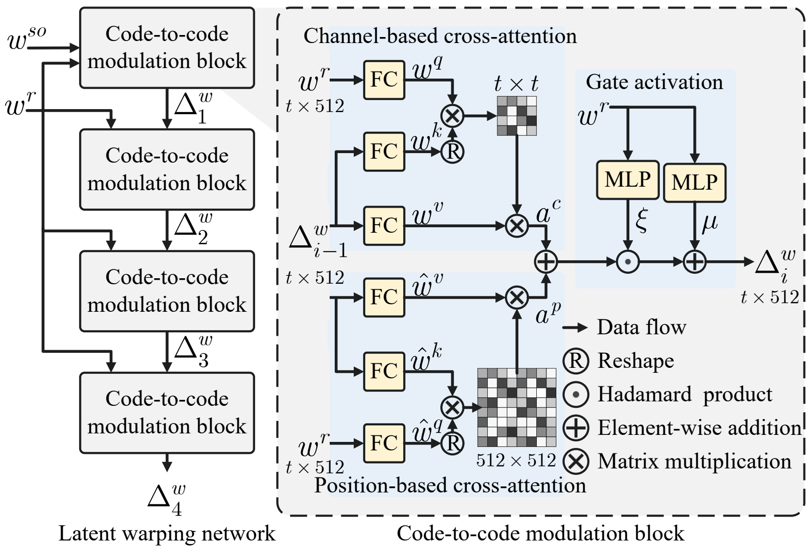


Figure 6.15: The architecture of the latent warping network.

6.4 More implementation details

6.4.1 Latent warping module

Let $w^{ta} \in \mathbb{R}^{t\times 512\times 1}$ and $w^{so} \in \mathbb{R}^{t\times 512\times 1}$ be the target and source latent codes, and $w^r = w^{ta} - w^{so}$ be the residual latent codes between w^{ta} and w^{so}. As illustrated in Fig. 6.15 (right), each code-to-code modulation block computes channel-based attention [147], position-based attention [147], and gated maps [26] between w^r and

Δ^w_{i-1}, and outputs the latent codes Δ^w_i $(i = 1, 2, 3, 4)$:

$$\begin{aligned}
&w^q = \mathrm{FC}(w^r), w^k = \mathrm{FC}(\Delta^w_{i-1}), w^v = \mathrm{FC}(\Delta^w_{i-1}),\\
&a^c = \mathrm{Softmax}(w^q \cdot w^{k^\top}/\tau_1) \cdot w^v,\\
&\hat{w}^q = \mathrm{FC}(w^r), \hat{w}^k = \mathrm{FC}(\Delta^w_{i-1}), \hat{w}^v = \mathrm{FC}(\Delta^w_{i-1}),\\
&a^p = \hat{w}^v \cdot \mathrm{Softmax}(\hat{w}^{q^\top} \cdot \hat{w}^k/\tau_2),\\
&\xi = \sigma(\mathrm{MLP}(w^r)), \mu = \phi(\mathrm{MLP}(w^r)),\\
&\Delta^w_i = \mathrm{LayerNorm}(a^p + a^c) \odot (\xi + 1) + \mu,
\end{aligned} \tag{6.15}$$

where w^q, w^k, w^v, a^c, $\hat{w}^q$, $\hat{w}^k$, $\hat{w}^v$, a^p, ξ, μ, and Δ^w_i have the same dimension as w^{so}, and $\Delta^w_0 = w^{so}$. We set $\tau_1 = \sqrt{t}$ and $\tau_2 = \sqrt{512}$. Our code-to-code modulation block calculates query projections (w^q and $\hat{w}^q$), key projections (w^k and $\hat{w}^k$), and value projections (w^v and $\hat{w}^v$) to obtain channel-based and position-based cross-attention maps, respectively. This allows us to obtain the reorganized latent codes a^c and a^p. The gated maps ξ and the bias μ are utilized to assign importance weights and offsets to each element. $\mathrm{FC}(\cdot)$ is a fully connected layer; $\mathrm{Softmax}(\cdot)$ is the softmax activation; $\mathrm{MLP}(\cdot)$ is a stack of two fully connected layers; $\mathrm{LayerNorm}(\cdot)$ is the LayerNorm layer.

The latent warping network $H_{\theta_h}(\cdot)$ outputs the latent codes Δ^w_4 and we can obtain the warped latent codes $w^{wa} = \Delta^w_4 + w^{so}$. By leveraging the code-to-code modulation mechanisms, our latent warping module can effectively align the pose of the source latent representations with the target latent codes.

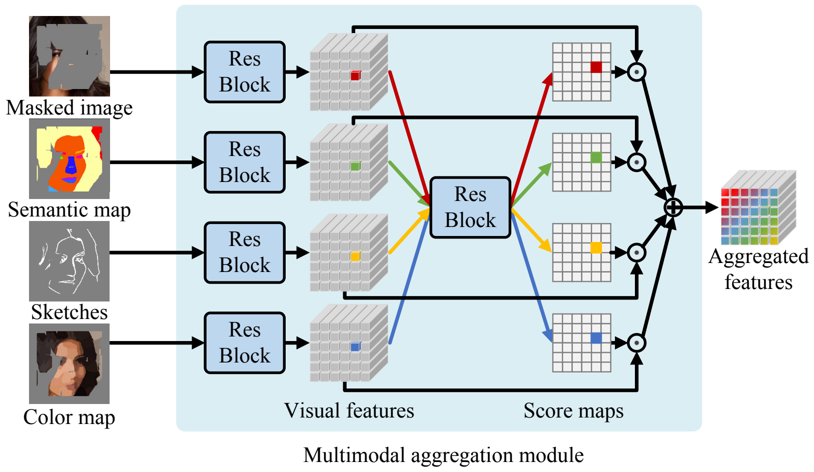


Figure 6.16: Illustration of the multimodal aggregation module.

6.4.2 Multimodal aggregation module

As shown in Fig. 6.16, for the given pixel-wise multimodal inputs $\mathcal{X} = \{\mathbf{I}_m, \mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_n\}$, we first employ a residual block to extract feature maps for each modality, resulting in a feature set $\{\mathbf{F}_0^a, \mathbf{F}_1^a, \mathbf{F}_2^a, \ldots, \mathbf{F}_n^a\}$, where $\mathbf{F}_j^a \in \mathbb{R}^{h \times w \times c_a}$ and $j = 0, 1, \ldots, n$. Subsequently, a shared residual block is utilized to compute the contribution scores for each spatial point across all pixel-wise modalities, producing $n+1$ score maps $\{\mathbf{B}_0, \mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_n\}$, where $\mathbf{B}_j \in \mathbb{R}^{h \times w}$ and $j = 0, 1, \ldots, n$. To obtain the normalized contribution score for each modality, the softmax activation is applied to normalize scores along the channel dimension. Specifically, the normalized score map $\hat{\mathbf{B}}_k \in \mathbb{R}^{h \times w}$ for the k-th modality is computed as follows:

$$\hat{\mathbf{B}}(u, v)_k = \frac{\exp\left(\mathbf{B}(u, v)_k\right)}{\sum_{j=0}^{n} \exp\left(\mathbf{B}(u, v)_j\right)}, \tag{6.16}$$

where (u, v) denotes the spatial point. The contribution score map adaptively weights the importance of each modality in a pixel-wise fashion for the aggregation process.

This adaptive weighting mechanism allows the model to assign higher importance to the more informative and detailed pixel-wise modalities while reducing the impact of less informative inputs. The final aggregated multimodal feature is obtained with the broadcasting technique:

$$\hat{\mathbf{F}}^a = \sum_{j=0}^{n} \mathbf{F}_j^a \odot \hat{\mathbf{B}}_j. \tag{6.17}$$

6.4.3 Training of multimodal aggregation module, refinement auto-encoder and discriminator

The identity loss, the LPIPS loss, the diversity-enhanced attribute loss, and the adversarial loss are combined to optimize the multimodal aggregation module, refinement auto-encoder, and discriminator.

Identity loss. The identity loss $\mathcal{L}_{id}(\mathbf{I}_{out}, \mathbf{I}_{ex})$ is employed to constrain the identity similarity between the edited image $\mathbf{I}_{out}$ and the exemplar image $\mathbf{I}_{ex}$.

The identity loss [27, 29] is defined as:

$$\mathcal{L}_{id}(\mathbf{I}_{out}, \mathbf{I}_{ex}) = 1 - \cos(R(\mathbf{I}_{out}), R(\mathbf{I}_{ex})), \tag{6.18}$$

where $R(\cdot)$ is a pre-trained ArcFace network [195].

LPIPS loss. The LPIPS loss $\mathcal{L}_{lpips}(\mathbf{I}_{out}, \mathbf{I}_{gt})$ is applied to enforce the perceptual similarity between $\mathbf{I}_{out}$ and $\mathbf{I}_{gt}$. When $\mathbf{I}_{gt}$ and $\mathbf{I}_{ex}$ are not from the same image, we set $\mathcal{L}_{lpips}(\mathbf{I}_{out}, \mathbf{I}_{gt}) = 0$.

The LPIPS loss [196] is defined as:

$$\mathcal{L}_{lpips}(\mathbf{I}_{out}, \mathbf{I}_{gt}) = \|P(\mathbf{I}_{out}) - P(\mathbf{I}_{gt})\|_2, \tag{6.19}$$

where $P(\cdot)$ corresponds the pre-trained perceptual feature extractor VGG [149].

Diversity-enhanced attribute loss. In order to learn the mapping between style latent codes and corresponding facial attributes, and support attribute-conditional editing in the style latent space, we propose a diversity-enhanced attribute loss to constrain the consistency between facial attributes of the edited image $\mathbf{I}_{out}$ and the interpolated latent codes $\overline{w}$. To emulate the latent editing process during the training phase, our approach involves style mixing and interpolation operations. We first apply the style mixing [60] to get mixed latent codes $\hat{w}^z$ from two random latent codes. Then we generate exemplar latent codes $w^e = E_{\hat{\theta}_e}(\mathbf{I}_{ex})$ and interpolate between $\hat{w}^z$ and w^e to obtain the interpolated latent codes $\overline{w}$:

$$\begin{aligned} \hat{w}^z =& \text{Mixing}(F_{\hat{\theta}_f}(z_1), F_{\hat{\theta}_f}(z_2)), \\ \overline{w} =& \alpha \cdot w^e + (1-\alpha) \cdot \hat{w}^z, \end{aligned} \tag{6.20}$$

where $z_1 \in \mathcal{Z}$ and $z_2 \in \mathcal{Z}$ are two random latent codes, $\alpha \in [0,1]$ is the uniformly sampled random number, and we set $\alpha = 1.0$ when $\mathbf{I}_{gt} = \mathbf{I}_{ex}$. Then, we modulate the interpolated latent codes $\overline{w}$ into the multimodal generator to obtain the edited image $\mathbf{I}_{out}$. We then apply the attribute loss to constrain the training in the style latent space, i.e., $\mathcal{L}_{attr}(w^o, \overline{w})$, where $w^o = E_{\hat{\theta}_e}(\mathbf{I}_{out})$.

The attribute loss [27, 210] is defined as:

$$\mathcal{L}_{attr}(w^o, \overline{w}) = \|w^o - \overline{w}\|_2. \tag{6.21}$$

The diversity-enhanced attribute loss is designed to mimic the latent attribute editing process, promoting a diverse range of embedded latent codes throughout training. By exposing the multimodal generator to a wide variety of mapping cases, the model becomes more effective in handling potential latent codes, thereby improving the latent attribute editing capability.

Adversarial loss. We use the adversarial non-saturating logistic loss [41] with R_1 regularization [151]:

$$\begin{aligned}\mathcal{L}_{adv}(\mathbf{I}_{out}, \mathbf{I}_{gt}) &= \mathbb{E}_{\mathbf{I}_{out}}[\log(1 - D(\mathbf{I}_{out})] \\ &+ \mathbb{E}_{\mathbf{I}_{gt}}[\log(D(\mathbf{I}_{gt}))] - \frac{\gamma}{2}\mathbb{E}_{\mathbf{I}_{gt}}[\|\nabla_{\mathbf{I}_{gt}} D(\mathbf{I}_{gt})\|_2^2],\end{aligned} \tag{6.22}$$

where $\gamma = 10$ is used to balance the R_1 regularization term.

Total loss. The total training loss is defined as:

$$\begin{aligned}O(\theta_a, \theta_g, \theta_d) &= \lambda_{id}\mathcal{L}_{id}(\mathbf{I}_{out}, \mathbf{I}_{ex}) + \lambda_{attr}\mathcal{L}_{attr}(w^o, \overline{w}) \\ &+ \lambda_{lpips}\mathcal{L}_{lpips}(\mathbf{I}_{out}, \mathbf{I}_{gt}) + \mathcal{L}_{adv}(\mathbf{I}_{out}, \mathbf{I}_{gt}),\end{aligned} \tag{6.23}$$

where $\lambda_{id} = 0.1$, $\lambda_{lpips} = 0.5$, and $\lambda_{attr} = 0.1$ in this work.

For each iteration, we obtain the optimized parameters θ_a^*, θ_g^* and θ_d^* via the minimax game iteratively:

$$\begin{aligned}(\theta_a^*, \theta_g^*) &= \arg\min_{\theta_a, \theta_g} O(\theta_a, \theta_g, \theta_d), \\ \theta_d^* &= \arg\max_{\theta_d} O(\theta_a, \theta_g, \theta_d).\end{aligned} \tag{6.24}$$

The refinement network G_{θ_g} is trained to generate a realistic edited image $\mathbf{I}_{out}$ while the discrinimator D_{θ_g} tries to differentiate between $\mathbf{I}_{gt}$ and $\mathbf{I}_{out}$. In an alternating fashion, A_{θ_a} and G_{θ_g} are trained in a phase while D_{θ_d} is trained in the other. Note that $\hat{\theta}_f$, $\hat{\theta}_e$, and $\hat{\theta}_s$ are keeping unchanged. During training, we attempted to embed all pixel-wise modalities in each iteration, but the model tended to overfit with the combined modalities and struggled with missing ones. As a solution, we randomly removed pixels and dropped out some input modalities for each training iteration, enhancing the robustness of missing modalities during inference.

6.4.4 Training peseudo-codes

The pseudo-codes of our training procedures are provided in Algorithm 3, Algorithm 4 and Algorithm 5. The threshold $\rho \in [0, 1]$ is used to control the probability that the sampled ground-truth image and exemplar image are the same. The threshold $\omega \in [0, 1]$ and the random masks $\hat{\mathcal{M}}$ are used to control the sparsity for multimodal inputs. We set $\rho = 0.5$ and $\omega = 0.8$ in this paper.

6.5 More experimental results and comparisons

6.5.1 Comparison on color-guided facial editing

We compared FACEMUG to SC-FEGAN [16] on color-guided facial editing using sketches and colors. We masked out the edited pixels and utilized the corresponding color information to guide makeup generation. To preserve the source facial structures and other facial attributes, we extracted sketches using a Canny edge detector to extract sketches for guiding the generation of facial geometry. Fig. 6.17 presents

Algorithm 3 Training procedure of the style encoder

```
1: while $E_{\theta_e}$ have not converged do
2:     Sample batch images $\mathcal{I}_{gt}$ from training data
3:     Set exemplars from ground-truth $\mathcal{I}_0 \leftarrow \mathcal{I}_{gt}$
4:     Sample corresponding multimodal inputs $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_n$
5:     Create random masks $\mathcal{M}$
6:     Sample a random number $r \in [0, n]$
7:     for $i = 0$ to $n$ do
8:         if $r = i$ then
9:             Get masked modality $\mathcal{I}_i \leftarrow \mathcal{M} \odot \mathcal{I}_i$
10:        else
11:            Set modality $\mathcal{I}_i$ to be zero tensor
12:    Set inputs $\mathcal{X} \leftarrow \{\mathcal{I}_0, \mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_n\}$
13:    Get projected latent codes $w^p \leftarrow E_{\theta_e}(\mathcal{X})$
14:    Get a reconstructed image $\mathcal{I}_p \leftarrow S_{\hat{\theta}_s}(w^p)$
15:    Update $\theta_e$ using the total loss defined in e4e
```

visual illustrations of editing results. Results from SC-FEGAN show clear responses to the input sketches and colors. However, some visual artifacts are noticeable in the masked regions. In contrast, FACEMUG utilizes sketches for facial geometry and colors for color editing to enable customized makeup editing within editing regions. Our style fusion blocks efficiently utilize extracted features to produce more visually appealing results.

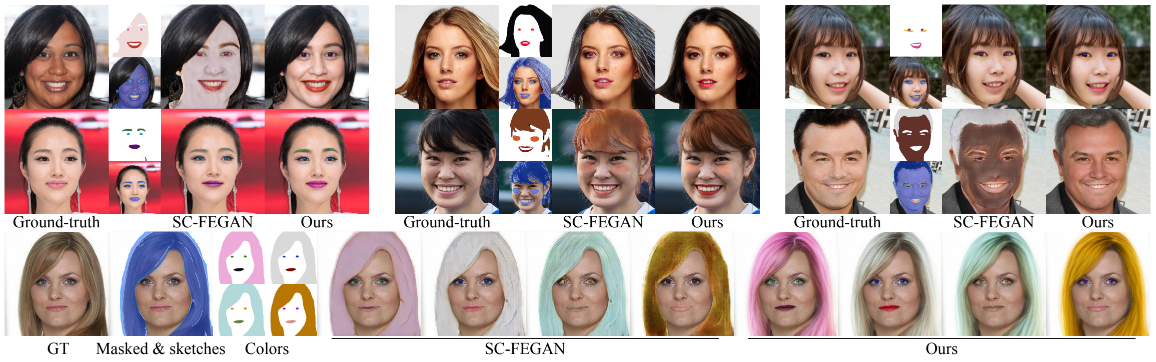


Figure 6.17: Visual comparison of color-guided facial editing between SC-FEGAN [16] and ours: (first two rows) sub-images represent the colors and masked images with sketches; (bottom row) diverse editing results guided by different color maps. Our method produces images with higher quality.

Algorithm 4 Training procedure of our latent warping module

```
1:  while $H_{\theta_h}$ has not converged do
2:      Sample batch images $\mathcal{I}_{gt}$ from training data
3:      Calculate augmented images $\mathcal{I}_{ta}$ from $\mathcal{I}_{gt}$
4:      Calculate mirror flipped images $\mathcal{I}_f$ from $\mathcal{I}_{gt}$
5:      Get ground-truth latent codes $w^{gt} \leftarrow E_{\hat{\theta}_e}(\mathcal{I}_{gt})$
6:      Get target latent codes $w^{ta} \leftarrow E_{\hat{\theta}_e}(\mathcal{I}_{ta})$
7:      Get flipped latent codes $w^{f} \leftarrow E_{\hat{\theta}_e}(\mathcal{I}_{f})$
8:      Sample a random number $\beta \in [0, 1]$
9:      Get source latent codes $w^{so} \leftarrow \beta \cdot w^{gt} + (1 - \beta) \cdot w^{f}$
10:     Get results $w^{wa} \leftarrow H_{\theta_h}(w^{ta} - w^{so}, w^{so}) + w^{so}$
11:     Update $\theta_h$ with $\mathcal{L}_{attr}$, $\mathcal{L}_{id}$, and $\mathcal{L}_{lpips}$
```

Algorithm 5 Training procedure of the multimodal aggregation module, the refinement auto-encoder, and the discriminator

```
1: while $A_{\theta_a}$, $G_{\theta_g}$, and $D_{\theta_d}$ have not converged do
2:     Sample batch images $\mathcal{I}_{gt}$ from training data
3:     Sample corresponding multimodal inputs $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_n$
4:     Create random masks $\mathcal{M}$
5:     Sample a random number $r \in [0, 1]$
6:     if $r >$ threshold $\rho$ then
7:         Sample exemplars $\mathcal{I}_{ex}$ from training data
8:         Set $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_n$ to be zero tensors
9:     else
10:        Set exemplars from ground-truth $\mathcal{I}_{ex} \leftarrow \mathcal{I}_{gt}$
11:        Sample random numbers $q_1, q_2, \ldots, q_n \in [0, 1]$
12:        for $i = 1$ to $n$ do
13:            if $q_i >$ threshold $\omega$ then
14:                Set modality $\mathcal{I}_i$ to be zero tensor
15:            else
16:                Create random masks $\hat{\mathcal{M}}$
17:                Get masked modality $\mathcal{I}_i \leftarrow \hat{\mathcal{M}} \odot \mathcal{M} \odot \mathcal{I}_i$
18:    Get masked images $\mathcal{I}_m \leftarrow \mathcal{I}_{gt} \odot (1 - \mathcal{M})$
19:    Set inputs $\mathcal{X} \leftarrow \{\mathcal{I}_m, \mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_n\}$
20:    Get exemplar latent codes $w^e \leftarrow E_{\hat{\theta}_e}(\mathcal{I}_{ex})$
21:    if $r >$ threshold $\rho$ then
22:        Sample random latent vectors $\mathcal{Z}_1$ and $\mathcal{Z}_2$
23:        Get mixed codes $\hat{w}^z \leftarrow \text{Mixing}(F_{\theta_f}(\mathcal{Z}_1), F_{\theta_f}(\mathcal{Z}_2))$
24:        Sample a random number $\alpha \in [0, 1]$
25:        Get interpolated codes $\overline{w} \leftarrow \alpha \cdot w^z + (1 - \alpha) \cdot \hat{w}^e$
26:    else
27:        Get codes from exemplar latent codes $\overline{w} \leftarrow w^e$
28:    Get projected latent codes $w^p \leftarrow E_{\hat{\theta}_e}(\mathcal{X})$
29:    Extract facial features from the StyleGAN generator $\mathcal{F}^s \leftarrow S_{\hat{\theta}_s}(w^p)$
30:    Extract aggregated features $\mathbf{F}^a \leftarrow A_{\theta_a}(\mathcal{X})$
31:    Get $\mathcal{I}_{out} \leftarrow \mathcal{I}_m \odot (\mathbf{1} - \mathcal{M}) + G_{\theta_g}(\mathbf{F}^a, \overline{w}, \mathcal{F}^s) \odot \mathcal{M}$
32:    Update $\theta_g$ with $\mathcal{L}_{adv}$, $\mathcal{L}_{id}$, $\mathcal{L}_{lpips}$, and $\mathcal{L}_{attr}$
33:    Update $\theta_d$ with $\mathcal{L}_{adv}$
```

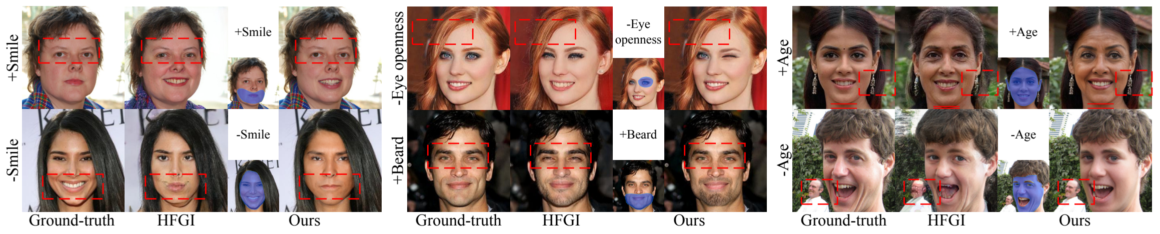


Figure 6.18: Visual comparison of attribute-conditional editing between HFGI [17] and our FACEMUG. For FACEMUG, the left sub-images in each group represent the attribute label (top) and editing regions (bottom). FACEMUG shows more flexible attribute-conditional editing by only manipulating selected regions (e.g., first row, middle, wink expression), and keeps unrelated features unchanged to show better visual quality and global consistency.

6.5.2 Comparison on local attribute-conditional facial editing

Through aligning our latent space with the $\mathcal{W}+$ style latent space [60], we demonstrate that our approach is not only capable of performing attribute-conditional facial edits using off-the-shelf latent-based semantic editing techniques [134, 24, 137, 135], but also retains the integrity of unedited regions.

Fig. 6.18 shows how our method can carry out attribute-conditioned semantic modifications on masked relevant semantic areas. The compared method HFGI [17] alters unwanted facial attributes or background information, while our approach ensures the background information (unmasked area) remains unchanged. Additionally, FACEMUG exhibits the capability to execute more complex edits, such as producing

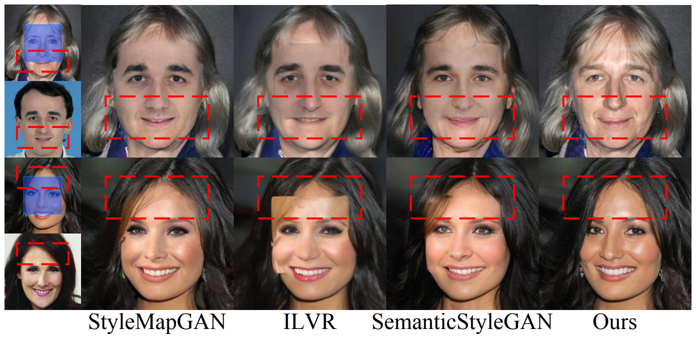


Figure 6.19: Visual comparison of our FACEMUG to the SOTA exemplar-guided editing methods (StyleMapGAN [5], ILVR [7], and SemanticStyleGAN [6]): top-left (masked image), bottom-left (exemplar image), right (results). FACEMUG shows seamless incorporation of exemplar facial attributes while keeping unmasked regions unchanged and achieving global consistency.

a winking expression, attributable to its flexibility in choosing editing regions.

6.5.3 Comparison on exemplar-guided facial editing

We also show the comparison with StyleMapGAN [5], ILVR [7], and SemanticStyleGAN [6] for exemplar-guided editing. To ensure proper facial alignment between the exemplar image and the input image, we extracted the roll, pitch, and yaw angles from the CelebAMask-HQ [123] dataset. Subsequently, we selected 550 pairs with similar poses from the testing set of CelebA-HQ. For each pair, we alternately used one facial image as the exemplar and the another as the masked input image to perform editing. The masked regions were replaced with corresponding exemplar facial features. Publicly available trained models were utilized to generate the editing results.

Table 6.7: Quantitative comparison of our method to existing exemplar-guided editing. **Bold**: top-1 quantity.

Metric	StyleMapGAN [5]	ILVR [7]	SemanticStyleGAN [6]	Ours
FID↓	24.59	62.27	32.54	**13.28**
LPIPS↓	0.5833	0.3495	0.5801	**0.0851**

As shown in Fig. 6.19, although most methods produce plausible results, some visible boundary inconsistencies can be found in the details of SemanticStyleGAN, ILVR, and StyleMapGAN. Moreover, these compared methods may introduce unwanted changes in the background or unedited regions. Our FACEMUG can seamlessly fill in the masked pixels using exemplar-like attributes without changing unmasked areas, yielding high-quality editing results while avoiding the above artifacts.

Table 6.7 shows the quantitative performance of compared methods. It shows that exemplar-guided editing is still a challenging task. For SemanticStyleGAN and ILVR, directly filling the corresponding exemplar's facial features into the masked regions may cause obvious artifacts. The style maps in StyleMapGAN can help achieve more harmonious editing results, but it still shows limits on exemplar-guided editing. Our FACEMUG achieves the best FID and LPIPS scores, indicating that the edited images from FACEMUG obtain the highest visual quality.

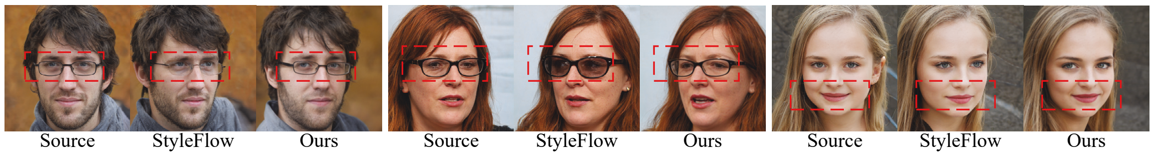


Figure 6.20: Visual comparison to StyleFlow [24] on pose editing. Our warping module shows better visual quality and visual consistency for facial attributes.

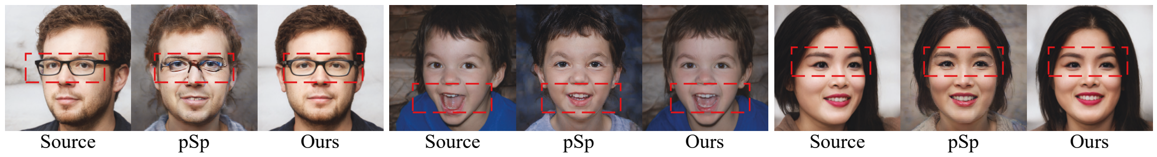


Figure 6.21: Visual comparison to pSp [18] on face frontalization. Our warping module provides superior visual quality and maintains the consistency of attributes.

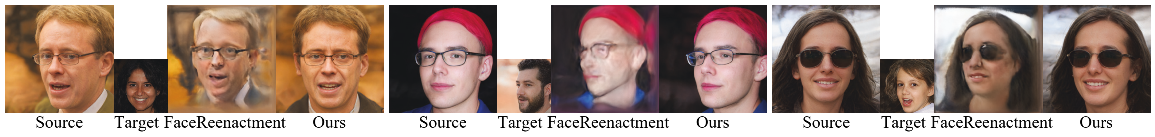


Figure 6.22: Visual comparison to FaceReenactment [25] on pose transfer. Our warping module ensures enhanced visual quality and consistency of facial features.

6.5.4 Comparison on guided facial pose editing

To further demonstrate the effectiveness of our latent warping module, we conducted comparisons with SOTA latent-based facial pose editing methods, including StyleFlow [24], pSp [18], and FaceReenactment [25] on pose editing, face frontalization, and pose transfer, respectively. We utilized their pre-trained models and codes obtained from their official websites. Fig. 6.20 shows visual examples of editing cases

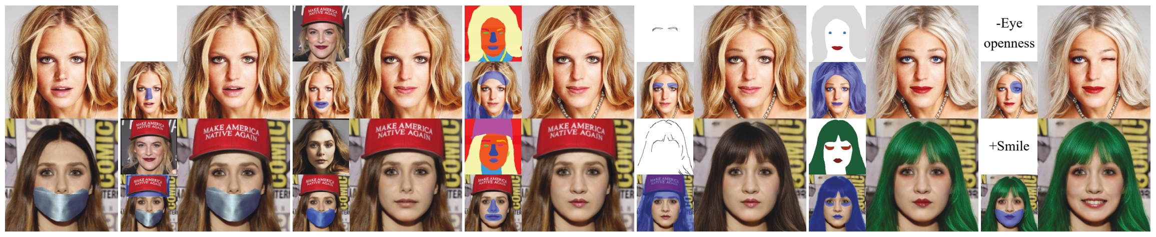


Figure 6.23: Incremental local facial editing examples with our FACEMUG. Each row: given an input image (first row), FACEMUG incrementally edits the facial image with blemish removal, exemplar-guided facial style transfer, semantic-guided attribute edits, sketch-guided hairstyle edits, color-guided makeup, and attribute-conditioned semantic edits (e.g., gender, age, and expression). For each group, FACEMUG only edits the masked area (bottom-left) guided by the guidance information (top-left) to produce the edited image (right). In the last row's first edit, we copy the hat to the input image and regenerate boundaries seamlessly.

from StyleFlow and ours. StyleFlow excels in editing facial orientation while preserving identity and other facial attributes. However, we observed minor changes in decorative attributes and expressions. In contrast, our latent warping module achieves more natural editing, ensuring consistency in facial attributes through our self-supervised training. Fig. 6.21 shows that pSp demonstrates face frontalization capabilities but exhibits minor visual artifacts in the generated results. On the contrary, our latent warping module produces more pleasing outcomes. Moreover, we computed the cosine similarity (CSIM [25]) of ArcFace between frontalized images and the ground-truth images for each method on prepared image pairs from Subsec-

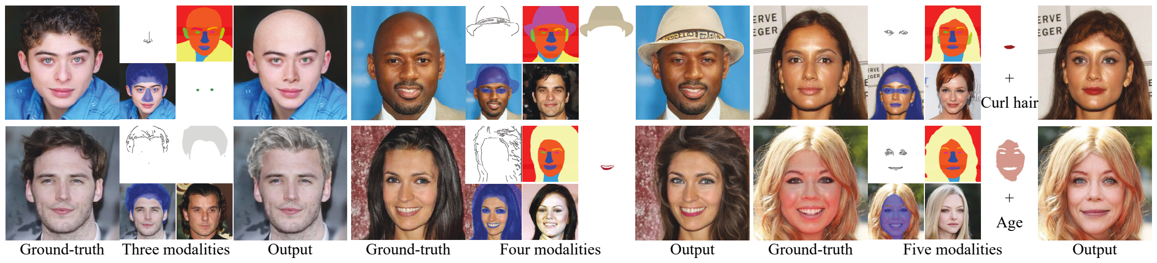


Figure 6.24: More multimodal local facial editing examples with our FACEMUG. Sketches, semantic maps, colors, exemplars, and attribute labels were utilized for these editing results.

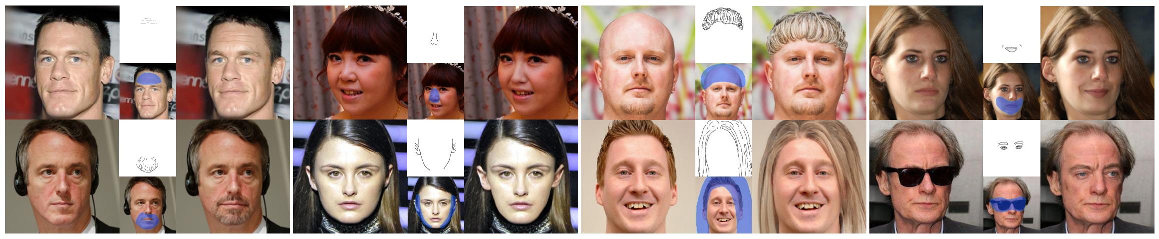


Figure 6.25: Examples of sketch-guided facial attribute editing with FACEMUG: texture editing, structure editing, hairstyle editing, and expression editing. For each group: (left) ground-truth, (top-middle) free-hand sketches, (bottom-middle) masked image, and (right) editing result.

tion 6.5.3. Compared to pSp with the CSIM score of 0.035, our FACEMUG achieves the CSIM score of 0.835, demonstrating the superiority of our method in preserving facial identity. Fig. 6.22 shows that FaceReenactment focuses on transferring the facial pose from a target image to a source face. Compared to FaceReenactment, our warping module effectively transfers the facial pose of source images to the target faces with high fidelity.

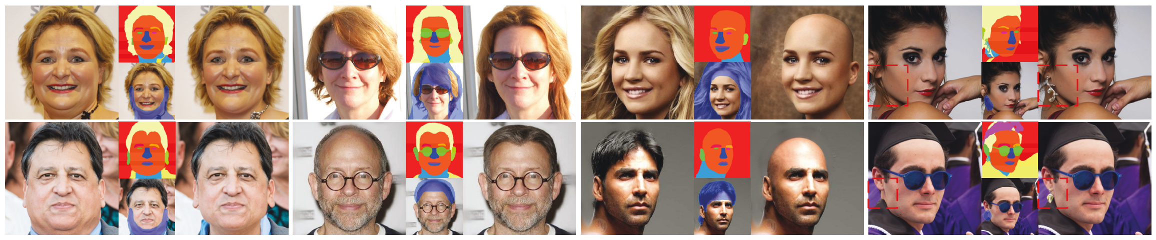


Figure 6.26: Examples of semantic-guided facial attribute editing with FACEMUG: chin editing, hairstyle editing, hair removal, and accessory addition. For each group: (left) ground-truth, (top-middle) hand-edited semantic maps, (bottom-middle) masked image, and (right) editing result.

6.5.5 More results on multimodal facial editing

Here, we show more results guided by multimodalities, including incremental local facial editing in Fig. 6.23 and multimodal local facial editing in Fig. 6.24. We also show more facial structure editing of FACEMUG by utilizing free-hand sketches and hand-edited semantic maps. As illustrated in Fig. 6.25, our FACEMUG framework allows for facial attribute editing using sketches. When provided with masked images and free-hand facial sketches, our method is capable of performing editing on various facial features such as texture (wrinkles and beards), structure (nose and chin), hairstyles, and expressions (mouth and eyes) while preserving the unedited regions unchanged. As displayed in Fig. 6.26, FACEMUG endows users with the capability to edit chins, remove hair, modify hairstyles, and add accessories, all utilizing the provided masked images and semantic maps. Our approach generates visually appealing and globally consistent images, effectively responding to multimodal inputs while preserving the unmasked areas.

6.6 Summary

In this chapter, we have explored a novel multimodal generative and fusion framework (FACEMUG) for globally-consistent local facial editing, which generates realistic facial features in response to multimodal inputs on the edited regions while maintaining visual coherence with unedited parts. FACEMUG takes advantage of diverse input modalities (e.g., sketches, semantic maps, color maps, exemplar images, text, and attribute labels) to perform fine-grained and semantic facial editing on geometry, color, expressions, attributes, and identity within edited regions, and allows for incremental edits. Extensive experiments have demonstrated the effectiveness of the proposed method.

Limitations and future work. Our approach comes with certain limitations. First, although the inference of FACEMUG is fast, the training of FACEMUG is time-intensive and requires approximately one month to complete on a V100 GPU. We plan to develop a more lightweight model to expedite FACEMUG's training process in the future. Second, because of the limited training data, the pre-trained StyleGAN struggles to generate relatively extreme expressions, poses, and appearances, potentially leading to FACEMUG failures in these domains. A more expressive and powerful pre-trained StyleGAN will improve our model. Third, like most multimodal editing algorithms [14], FACEMUG may not generate satisfying results when different modalities contain contradictory guiding information. Designing a more sophisticated technique to reweight the contribution of each modality would be a promising step forward. Finally, an interesting future direction is to incorporate more modalities, such as text and audio, into our multimodal aggregation module to achieve more diverse editing. A possible solution is to employ facial landmarks as an intermediate motion representation [213].

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis introduces four innovative methods for manipulating face images using advanced generative models. We started with a diffusion-based style prompt framework for blind face restoration; then, we designed the GRIG for data-efficient image inpainting. To explore conditional image inpainting and editing, we proposed the EXE-GAN for interactive facial inpainting using exemplar images. We finally designed the FACEMUG to investigate multimodal facial manipulation.

First, the diffusion-based style prompt framework uses a SMART layer and a style prompt module to greatly improve the quality of blind face restoration, demonstrating its utility in practical applications. Second, GRIG employs iterative residual reasoning with image-level and patch-level discriminators, enhancing data efficiency in image inpainting and setting new performance benchmarks across both small and large datasets. Third, EXE-GAN uses exemplar facial attributes to produce realistic and visually consistent outcomes, highlighting the effectiveness of attribute-driven generative models. Fourth, FACEMUG utilizes various input modalities to ensure high visual consistency between edited and unedited regions, proving the strength of integrating multiple modalities in generative networks. These methods improve the visual quality of the results and streamline processes to more accurately mimic human visual perception. The extensive experiments and applications have demonstrated the effectiveness of the proposed methods.

7.2 Limitations and discussions

While our proposed methods demonstrated considerable advancements, they come with certain limitations. First, like most blind facial restoration algorithms, our style prompt framework may not work well on highly complex backgrounds. That might be caused by the lack of sufficiently diverse background data in the training set. Designing a more sophisticated technique to decouple the foreground and background restoration would be a promising direction. Second, GRIG is not specialized in diverse image inpainting. Using a mapping network to embed random style codes into the generator could be a good solution for the diversity of data-efficient image inpainting. Third, for the EXE-GAN, the explicit mapping between the facial attribute and the embedded style codes, on the other hand, is still unknown [18]. Incorporating a more advanced embedding algorithm into our pipeline could be a good next step. The trained model works well for aligned images because the facial images in the experimented training datasets [154, 42] are highly aligned. It is necessary to align and crop the inputs before inpainting nonaligned images. It would be preferable to train models on unstructured datasets to create a more sophisticated algorithm. Fourth, although the inference of FACEMUG is fast, the training of FACEMUG is time-intensive and requires approximately one month to complete on a V100 GPU. We plan to develop a more lightweight model to expedite FACEMUG's training process in the future. In addition, because of the limited training data, the pre-trained StyleGAN struggles to generate relatively extreme expressions, poses, and appearances, potentially leading to FACEMUG failures in these domains. A more expressive and powerful pre-trained StyleGAN will improve our model. Moreover, like most multimodal editing algorithms [14], FACEMUG may not generate satisfying results when different modalities contain contradictory guiding information. Designing a more sophisticated technique to reweight the contribution of each modality would be a promising step forward.

7.3 Future work

Current models excel at capturing major facial features like expressions and poses but often overlook finer details such as decorative goods or facial textures. Future research could aim to enhance the models' ability to recognize and preserve these smaller details. Incorporating advanced features like the Transformer [46] and Mamba [214] into existing models could improve their accuracy in maintaining both overall and detailed aspects of images.

As technologies for creating fake images and videos advance, the potential for misuse increases. Future efforts should concentrate on developing robust and transparent methods to detect and counteract these threats. This is crucial for safeguarding privacy and maintaining trust in digital media.

Bibliography

[1] Yajie Zhao, Weikai Chen, Jun Xing, Xiaoming Li, Zach Bessinger, Fuchang Liu, Wangmeng Zuo, and Ruigang Yang. Identity preserving face completion for large ocular region occlusion. In *BMVC*, page Article 109, 2018.

[2] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *CVPR*, pages 8293–8302, 2020.

[3] Yinan Zhao, Brian Price, Scott Cohen, and Danna Gurari. Guided image inpainting: Replacing an image region by pulling content from another image. In *WACV*, pages 1514–1523, 2019.

[4] Rong Zhang, Wei Li, Yiqun Zhang, Hong Zhang, Jinhui Yu, Ruigang Yang, and Weiwei Xu. Image re-composition via regional content-style decoupling. In *ACM MM*, pages 3–11, 2021.

[5] Hyunsu Kim, Yunjey Choi, Junho Kim, Sungjoo Yoo, and Youngjung Uh. Exploiting spatial dimensions of latent in gan for real-time image editing. In *CVPR*, pages 852–861, 2021.

[6] Yichun Shi, Xiao Yang, Yangyue Wan, and Xiaohui Shen. Semanticstylegan: Learning compositional generative priors for controllable image synthesis and editing. In *CVPR*, 2022.

[7] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. In *ICCV*, pages 14347–14356, 2021.

[8] Jiacheng Li, Zhiwei Xiong, and Dong Liu. Reference-guided landmark image inpainting with deep feature matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(12):8422–8435, 2022.

[9] Taorong Liu, Liang Liao, Zheng Wang, and Shin'Ichi Satoh. Reference-guided texture and structure inference for image inpainting. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 1996–2000. IEEE, 2022.

[10] Liang Liao, Taorong Liu, Delin Chen, Jing Xiao, Zheng Wang, Chia-Wen Lin, and Shin'Ichi Satoh. Transref: Multi-scale reference embedding transformer for reference-guided image inpainting. *arXiv preprint arXiv:2306.11528*, 2023.

[11] Junfeng Lyu, Zhibo Wang, and Feng Xu. Portrait eyeglasses and shadow removal by leveraging 3d synthetic data. In *CVPR*, pages 3429–3439, 2022.

[12] Ziqi Huang, Kelvin C.K. Chan, Yuming Jiang, and Ziwei Liu. Collaborative diffusion for multi-modal face generation and editing. In *CVPR*, 2023.

[13] Nithin Gopalakrishnan Nair, Wele Gedara Chaminda Bandara, and Vishal M Patel. Unite and conquer: Plug & play multi-modal synthesis using diffusion models. In *CVPR*, pages 6070–6079, 2023.

[14] Xun Huang, Arun Mallya, Ting-Chun Wang, and Ming-Yu Liu. Multimodal conditional image synthesis with product-of-experts gans. In *ECCV*, page 91–109, 2022.

[15] Shu-Yu Chen, Feng-Lin Liu, Yu-Kun Lai, Paul L. Rosin, Chunpeng Li, Hongbo Fu, and Lin Gao. DeepFaceEditing: Deep face generation and editing with disentangled geometry and appearance control. *TOG*, 40(4):90:1–90:15, 2021.

[16] Youngjoo Jo and Jongyoul Park. Sc-fegan: Face editing generative adversarial network with user's sketch and color. In *ICCV*, pages 1745–1753, 2019.

[17] Tengfei Wang, Yong Zhang, Yanbo Fan, Jue Wang, and Qifeng Chen. High-fidelity gan inversion for image attribute editing. In *CVPR*, 2022.

[18] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *CVPR*, pages 2287–2296, 2021.

[19] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, pages 3836–3847, 2023.

[20] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. Sean: Image synthesis with semantic region-adaptive normalization. In *CVPR*, 2020.

[21] Anpei Chen, Ruiyang Liu, Ling Xie, Zhang Chen, Hao Su, and Jingyi Yu. Sofgan: A portrait image generator with dynamic styling. *TOG*, 41(1):Article 1, 2022.

[22] Weilun Wang, Jianmin Bao, Wengang Zhou, Dongdong Chen, Dong Chen, Lu Yuan, and Houqiang Li. Semantic image synthesis via diffusion models. *arXiv preprint arXiv:2207.00050*, 2022.

[23] Jingxiang Sun, Xuan Wang, Yichun Shi, Lizhen Wang, Jue Wang, and Yebin Liu. Ide-3d: Interactive disentangled editing for high-resolution 3d-aware portrait synthesis. *TOG*, 41(6), 2022.

[24] Rameen Abdal, Peihao Zhu, Niloy J. Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *TOG*, 40(3), 2021.

[25] Stella Bounareli, Vasileios Argyriou, and Georgios Tzimiropoulos. Finding directions in gan's latent space for neural face reenactment. *BMVC*, 2022.

[26] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas Huang. Free-form image inpainting with gated convolution. In *ICCV*, pages 4470–4479, 2019.

[27] Wanglong Lu, Hanli Zhao, Xianta Jiang, Xiaogang Jin, Yong-Liang Yang, and Kaijie Shi. Do inpainting yourself: Generative facial inpainting guided by exemplars. *Neurocomputing*, 617:128996, 2025.

[28] Shuai Yang, Liming Jiang, Ziwei Liu, and Chen Change Loy. Pastiche master: Exemplar-based high-resolution portrait style transfer. In *CVPR*, pages 7693–7702, 2022.

[29] Yiming Zhu, Hongyu Liu, Yibing Song, Ziyang Yuan, Xintong Han, Chun Yuan, Qifeng Chen, and Jue Wang. One model to edit them all: Free-form text-driven image manipulation with semantic modulations. In *NeurIPS*, 2022.

[30] Chao Xu, Jiangning Zhang, Miao Hua, Qian He, Zili Yi, and Yong Liu. Region-aware face swapping. In *CVPR*, pages 7632–7641, 2022.

[31] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Only a matter of style: Age transformation using a style-based regression model. *TOG*, 40(4), 2021.

[32] Yiqian Wu, Yong-Liang Yang, and Xiaogang Jin. Hairmapper: Removing hair from portraits using gans. In *CVPR*, pages 4227–4236, 2022.

[33] Yiqian Wu, Yong-Liang Yang, Qinjie Xiao, and Xiaogang Jin. Coarse-to-fine: facial structure editing of portrait images via latent space classifications. *TOG*, 40(4):Article 46, 2021.

[34] Chaofeng Chen, Xiaoming Li, Lingbo Yang, Xianhui Lin, Lei Zhang, and Kwan-Yee K. Wong. Progressive semantic-aware style transformation for blind face restoration. In *CVPR*, pages 11896–11905, 2021.

[35] Xin Yu, Basura Fernando, Bernard Ghanem, Fatih Porikli, and Richard Hartley. Face super-resolution guided by facial component heatmaps. In *ECCV*, pages 217–233, 2018.

[36] Xiaoming Li, Wenyu Li, Dongwei Ren, Hongzhi Zhang, Meng Wang, and Wangmeng Zuo. Enhanced blind face restoration with multi-exemplar images and adaptive spatial feature fusion. In *CVPR*, pages 2706–2715, 2020.

[37] Zhouxia Wang, Jiawei Zhang, Runjian Chen, Wenping Wang, and Ping Luo. Restoreformer: High-quality blind face restoration from undegraded key-value pairs. In *CVPR*, pages 17512–17521, 2022.

[38] Xintao Wang, Yu Li, Honglun Zhang, and Ying Shan. Towards real-world blind face restoration with generative facial prior. In *CVPR*, pages 9168–9178, 2021.

[39] Sachit Menon, Alexandru Damian, Shijia Hu, Nikhil Ravi, and Cynthia Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *CVPR*, pages 2437–2445, 2020.

[40] Zhouxia Wang, Jiawei Zhang, Tianshui Chen, Wenping Wang, and Ping Luo. Restoreformer++: Towards real-world blind face restoration from undegraded key-value paris. *TPAMI*, 45(12):15462–15476, 2023.

[41] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, volume 27, pages 2672–2680, 2014.

[42] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *TPAMI*, 43(12):4217–4228, 2021.

[43] Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural discrete representation learning. In *NeurIPS*, volume 30, page 6309–6318, 2017.

[44] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. In *WACV*, pages 3172–3182, 2022.

[45] Wenbo Li, Zhe Lin, Kun Zhou, Lu Qi, Yi Wang, and Jiaya Jia. Mat: Mask-aware transformer for large hole image inpainting. In *CVPR*, pages 10748–10758, 2022.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, volume 30, 2017.

[47] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In *ICLR*, 2017.

[48] Lei Zhao, Qihang Mo, Sihuan Lin, Zhizhong Wang, Zhiwen Zuo, Haibo Chen, Wei Xing, and Dongming Lu. Uctgan: Diverse image inpainting based on unsupervised cross-space translation. In *CVPR*, pages 5740–5749, 2020.

[49] Shengyu Zhao, Jonathan Cui, Yilun Sheng, Yue Dong, Xiao Liang, Eric I Chang, and Yan Xu. Large scale image completion via co-modulated generative adversarial networks. In *ICLR*, 2021.

[50] Yang Yang and Xiaojie Guo. Generative landmark guided face inpainting. In *PRCV*, pages 14–26, 2020.

[51] Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Qureshi, and Mehran Ebrahimi. Edgeconnect: Structure guided image inpainting using edge prediction. In *ICCVW*, pages 3265–3274, 2019.

[52] Weihao Xia, Yujiu Yang, Jing-Hao Xue, and Baoyuan Wu. Tedigan: Text-guided diverse face image generation and manipulation. In *CVPR*, pages 2256–2265, 2021.

[53] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[54] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, volume 33, pages 6840–6851, 2020.

[55] Yilun Du, Shuang Li, B. Joshua Tenenbaum, and Igor Mordatch. Learning iterative reasoning through energy minimization. In *ICML*, 2022.

[56] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[57] Wanglong Lu, Jikai Wang, Tao Wang, Kaihao Zhang, Xianta Jiang, and Hanli Zhao. Visual style prompt learning using diffusion models for blind face restoration. *Pattern Recognition*, 161:111312, 2025.

[58] Wanglong Lu, Xianta Jiang, Xiaogang Jin, Yong-Liang Yang, Minglun Gong, Tao Wang, Kaijie Shi, and Hanli Zhao. Grig: Few-shot generative residual image inpainting, 2023.

[59] Wanglong Lu, Jikai Wang, Xiaogang Jin, Xianta Jiang, and Hanli Zhao. Facemug: A multimodal generative and fusion framework for local facial editing. *TVCG*, pages 1–15, 2024.

[60] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, pages 8107–8116, 2020.

[61] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *ICCV*, pages 2849–2857, 2017.

[62] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, pages 2223–2232, 2017.

[63] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *TOG*, 36(4):Article 107, 2017.

[64] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, pages 4681–4690, 2017.

[65] Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic DENOYER, and Marc' Aurelio Ranzato. Fader networks:manipulating images by sliding attributes. In *NeurIPS*, volume 30, 2017.

[66] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *NeurIPS*, 2021.

[67] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, volume 37, pages 2256–2265, 2015.

[68] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In *NeurIPS*, 2021.

[69] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022.

[70] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[71] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. In *NeurIPS*, 2021.

[72] Tao Wang, Guangpin Tao, Wanglong Lu, Kaihao Zhang, Wenhan Luo, Xiaoqin Zhang, and Tong Lu. Restoring vision in hazy weather with hierarchical contrastive learning. *Pattern Recognition*, 145:109956, 2024.

[73] Tao Wang, Wanglong Lu, Kaihao Zhang, Wenhan Luo, Tae-Kyun Kim, Tong Lu, Hongdong Li, and Ming-Hsuan Yang. Promptrr: Diffusion models as prompt generators for single image reflection removal. *arXiv preprint arXiv:2402.02374*, 2024.

[74] Qingxing Cao, Liang Lin, Yukai Shi, Xiaodan Liang, and Guanbin Li. Attention-aware face hallucination via deep reinforcement learning. In *CVPR*, pages 690–698, 2017.

[75] Xiaoming Li, Chaofeng Chen, Shangchen Zhou, Xianhui Lin, Wangmeng Zuo, and Lei Zhang. Blind face restoration via deep multi-scale component dictionaries. In *ECCV*, 2020.

[76] Ziyu Wan, Bo Zhang, Dongdong Chen, Pan Zhang, Dong Chen, Jing Liao, and Fang Wen. Bringing old photos back to life. In *CVPR*, pages 2747–2757, 2020.

[77] Tao Yang, Peiran Ren, Xuansong Xie, and Lei Zhang. Gan prior embedded network for blind face restoration in the wild. In *CVPR*, pages 672–681, 2021.

[78] Feida Zhu, Junwei Zhu, Wenqing Chu, Xinyi Zhang, Xiaozhong Ji, Chengjie Wang, and Ying Tai. Blind face restoration via integrating face shape and generative priors. In *CVPR*, pages 7662–7671, 2022.

[79] Yu Chen, Ying Tai, Xiaoming Liu, Chunhua Shen, and Jian Yang. Fsrnet: End-to-end learning face super-resolution with facial priors. In *CVPR*, pages 2492–2501, 2018.

[80] Xin Qiu, Congying Han, Zicheng Zhang, Bonan Li, Tiande Guo, and Xuecheng Nie. Diffbfr: Bootstrapping diffusion model for blind face restoration. *ACM MM*, page 7785–7795, 2023.

[81] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera. Filling-in by joint interpolation of vector fields and gray levels. *TIP*, 10(8):1200–1211, 2001.

[82] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *SIGGRAPH*, pages 417–424, 2000.

[83] Levin, Zomet, and Weiss. Learning how to inpaint from global image statistics. In *ICCV*, volume 1, pages 305–312, 2003.

[84] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of Graphics Tools*, 9(1):23–34, 2004.

[85] Vivek Kwatra, Irfan Essa, Aaron Bobick, and Nipun Kwatra. Texture optimization for example-based synthesis. *TOG*, 24(3):795–802, 2005.

[86] Hanli Zhao, Heyang Guo, Xiaogang Jin, Jianbing Shen, Xiaoyang Mao, and Junru Liu. Parallel and efficient approximate nearest patch matching for image editing applications. *Neurocomputing*, 305:39–50, 2018.

[87] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *TOG*, 28(3):Article 24, 2009.

[88] Denis Simakov, Yaron Caspi, Eli Shechtman, and Michal Irani. Summarizing visual data using bidirectional similarity. In *CVPR*, pages 1–8, 2008.

[89] Ding Ding, Sundaresh Ram, and Jeffrey J. Rodríguez. Image inpainting using nonlocal texture matching and nonlinear filtering. *TIP*, 28(4):1705–1719, 2019.

[90] Jingjing Zheng, Wanglong Lu, Wenzhe Wang, Yankai Cao, Xiaoqin Zhang, and Xianta Jiang. Handling the non-smooth challenge in tensor svd: A multi-objective tensor recovery framework. In *ECCV*, pages 449–464, 2025.

[91] Jimmy SJ Ren, Li Xu, Qiong Yan, and Wenxiu Sun. Shepard convolutional neural networks. In *NeurIPS*, volume 28, 2015.

[92] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

[93] Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Image inpainting via generative multi-column convolutional neural networks. In *NeurIPS*, pages 331–340, 2018.

[94] Zhaoyi Yan, Xiaoming Li, Mu Li, Wangmeng Zuo, and Shiguang Shan. Shift-net: Image inpainting via deep feature rearrangement. In *ECCV*, 2018.

[95] Yanhong Zeng, Jianlong Fu, Hongyang Chao, and Baining Guo. Learning pyramid-context encoder network for high-quality image inpainting. In *CVPR*, pages 1486–1494, 2019.

[96] Hanli Zhao, Ying Liu, Wanglong Lu, Xiaogang Jin, Hui Huang, and Kaijie Shi. Efficient generative face completion with perceptual deblurring. *Journal of Computer-Aided Design & Computer Graphics*, 34(9):1420–1431, 2022.

[97] Hongyu Liu, Bin Jiang, Yi Xiao, and Chao Yang. Coherent semantic attention for image inpainting. In *ICCV*, pages 4169–4178, 2019.

[98] Chaohao Xie, Shaohui Liu, Chao Li, Ming-Ming Cheng, Wangmeng Zuo, Xiao Liu, Shilei Wen, and Errui Ding. Image inpainting with learnable bidirectional attention maps. In *ICCV*, pages 8857–8866, 2019.

[99] Zili Yi, Qiang Tang, Shekoofeh Azizi, Daesik Jang, and Zhan Xu. Contextual residual aggregation for ultra high-resolution image inpainting. In *CVPR*, pages 7508–7517, 2020.

[100] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Generative image inpainting with contextual attention. In *CVPR*, pages 5505–5514, 2018.

[101] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *ECCV*, pages 89–105, 2018.

[102] Wei Xiong, Jiahui Yu, Zhe Lin, Jimei Yang, Xin Lu, Connelly Barnes, and Jiebo Luo. Foreground-aware image inpainting. In *CVPR*, pages 5833–5841, 2019.

[103] Yurui Ren, Xiaoming Yu, Ruonan Zhang, Thomas H. Li, Shan Liu, and Ge Li. Structureflow: Image inpainting via structure-aware appearance flow. In *ICCV*, pages 181–190, 2019.

[104] Yanhong Zeng, Jianlong Fu, Hongyang Chao, and Baining Guo. Aggregated contextual transformations for high-resolution image inpainting. *TVCG*, pages 1–1, 2022.

[105] Chuanxia Zheng, Tat-Jen Cham, Jianfei Cai, and Dinh Phung. Bridging global context interactions for high-fidelity image completion. In *CVPR*, pages 11512–11522, 2022.

[106] Yu Zeng, Zhe Lin, Jimei Yang, Jianming Zhang, Eli Shechtman, and Huchuan Lu. High-resolution image inpainting with iterative confidence feedback and guided upsampling. In *ECCV*, pages 1–17, 2020.

[107] Ziyu Wan, Jingbo Zhang, Dongdong Chen, and Jing Liao. High-fidelity pluralistic image completion with transformers. In *ICCV*, pages 4672–4681, 2021.

[108] Qiankun Liu, Zhentao Tan, Dongdong Chen, Qi Chu, Xiyang Dai, Yinpeng Chen, Mengchen Liu, Lu Yuan, and Nenghai Yu. Reduce information loss in transformers for pluralistic image inpainting. In *CVPR*, pages 11347–11357, 2022.

[109] Shangchen Zhou, Kelvin C.K. Chan, Chongyi Li, and Chen Change Loy. Towards robust blind face restoration with codebook lookup transformer. In *NeurIPS*, 2022.

[110] Min Wang, Wanglong Lu, Jiankai Lyu, Kaijie Shi, and Hanli Zhao. Generative image inpainting with enhanced gated convolution and transformers. *Displays*, 75:102321, 2022.

[111] Ziyu Wan, Jingbo Zhang, Dongdong Chen, and Jing Liao. High-fidelity pluralistic image completion with transformers. In *ICCV*, pages 4672–4681, 2021.

[112] Qiankun Liu, Zhentao Tan, Dongdong Chen, Qi Chu, Xiyang Dai, Yinpeng Chen, Mengchen Liu, Lu Yuan, and Nenghai Yu. Reduce information loss in

transformers for pluralistic image inpainting. In *CVPR*, pages 11347–11357, 2022.

[113] Haoran Zhang, Zhenzhen Hu, Changzhi Luo, Wangmeng Zuo, and Meng Wang. Semantic image inpainting with progressive generative networks. In *ACM MM*, page 1939–1947, 2018.

[114] Jingyuan Li, Fengxiang He, Lefei Zhang, Bo Du, and Dacheng Tao. Progressive reconstruction of visual structure for image inpainting. In *ICCV*, pages 5961–5970, 2019.

[115] Jingyuan Li, Ning Wang, Lefei Zhang, Bo Du, and Dacheng Tao. Recurrent feature reasoning for image inpainting. In *CVPR*, 2020.

[116] Zongyu Guo, Zhibo Chen, Tao Yu, Jiale Chen, and Sen Liu. Progressive image inpainting with full-resolution residual network. In *ACM MM*, pages 2496–2504, 2019.

[117] Tiancheng Sun, Jonathan T. Barron, Yun-Ta Tsai, Zexiang Xu, Xueming Yu, Graham Fyffe, Christoph Rhemann, Jay Busch, Paul Debevec, and Ravi Ramamoorthi. Single image portrait relighting. *TOG*, 38(4), 2019.

[118] Hao Zhou, Sunil Hadap, Kalyan Sunkavalli, and David W. Jacobs. Deep single-image portrait relighting. In *ICCV*, 2019.

[119] Renwang Chen, Xuanhong Chen, Bingbing Ni, and Yanhao Ge. Simswap: An efficient framework for high fidelity face swapping. In *ACM MM*, pages 2003–2011, 2020.

[120] Tiziano Portenier, Qiyang Hu, Attila Szabó, Siavash Arjomand Bigdeli, Paolo Favaro, and Matthias Zwicker. Faceshop: Deep sketch-based face image editing. *TOG*, 37(4), 2018.

[121] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, pages 8789–8797, 2018.

[122] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *CVPR*, pages 8185–8194, 2020.

[123] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *CVPR*, pages 5548–5557, 2020.

[124] Taihong Xiao, Jiapeng Hong, and Jinwen Ma. Elegant: Exchanging latent encodings with gan for transferring multiple face attributes. In *ECCV*, pages 172–187, 2018.

[125] Jingtao Guo, Zhenzhen Qian, Zuowei Zhou, and Yi Liu. Mulgan: Facial attribute editing by exemplar. *arXiv preprint*, arXiv:1912.12396, 2019.

[126] Lingzhi Li, Jianmin Bao, Hao Yang, Dong Chen, and Fang Wen. Advancing high fidelity identity swapping for forgery detection. In *CVPR*, pages 5073–5082, 2020.

[127] Xiao-Chang Liu, Yong-Liang Yang, and Peter Hall. Learning to warp for style transfer. In *CVPR*, pages 3701–3710, 2021.

[128] Xinyang Li, Shengchuan Zhang, Jie Hu, Liujuan Cao, Xiaopeng Hong, Xudong Mao, Feiyue Huang, Yongjian Wu, and Rongrong Ji. Image-to-image translation via hierarchical style disentanglement. In *CVPR*, pages 8639–8648, 2021.

[129] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *ICCV*, pages 4431–4440, 2019.

[130] Daniel Roich, Ron Mokady, Amit H. Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *TOG*, 42(1), 2022.

[131] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *ECCV*, pages 592–608, 2020.

[132] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *TOG*, 40(4):Article 133, 2021.

[133] Bingchuan Li, Tianxiang Ma, Peng Zhang, Miao Hua, Wei Liu, Qian He, and Zili Yi. Reganie: Rectifying gan inversion errors for accurate real image editing. *AAAI*, 37(1):1269–1277, 2023.

[134] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. Interfacegan: Interpreting the disentangled face representation learned by gans. *TPAMI*, 2020.

[135] Rameen Abdal, Peihao Zhu, John Femiani, Niloy Mitra, and Peter Wonka. Clip2stylegan: Unsupervised extraction of stylegan edit directions. In *SIGGRAPH*, 2022.

[136] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. *CVPR*, pages 1532–1540, 2020.

[137] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. In *NeurIPS*, 2020.

[138] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *ICCV*, pages 2085–2094, 2021.

[139] Po-Wen Hsieh and Pei-Chiang Shao. Blind image deblurring based on the sparsity of patch minimum information. *Pattern Recognition*, 109:107597, 2021.

[140] Hao Shen, Zhong-Qiu Zhao, Wenrui Liao, Weidong Tian, and De-Shuang Huang. Joint operation and attention block search for lightweight image restoration. *Pattern Recognition*, 132:108909, 2022.

[141] Yadong Wang and Xiangzhi Bai. Versatile recurrent neural network for wide types of video restoration. *Pattern Recognition*, 138:109360, 2023.

[142] Shengmin Zhao, Sung-Kwun Oh, Jin-Yul Kim, Zunwei Fu, and Witold Pedrycz. Motion-blurred image restoration framework based on parameter estimation and fuzzy radial basis function neural networks. *Pattern Recognition*, 132:108983, 2022.

[143] Salma Gonzalez-Sabbagh, Antonio Robles-Kelly, and Shang Gao. Dgd-cgan: A dual generator for image dewatering and restoration. *Pattern Recognition*, 148:110159, 2024.

[144] Vaishnav Potlapalli, Syed Waqas Zamir, Salman Khan, and Fahad Khan. PromptIR: Prompting for all-in-one image restoration. In *NeurIPS*, page 19, 2023.

[145] Bin Xia, Yulun Zhang, Shiyin Wang, Yitong Wang, Xinglong Wu, Yapeng Tian, Wenming Yang, and Luc Van Gool. Diffir: Efficient diffusion model for image restoration. In *ICCV*, pages 13095–13105, 2023.

[146] Yuchao Gu, Xintao Wang, Liangbin Xie, Chao Dong, Gen Li, Ying Shan, and Ming-Ming Cheng. Vqfr: Blind face restoration with vector-quantized dictionary and parallel decoder. In *ECCV*, pages 126–143, 2022.

[147] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *CVPR*, pages 3146–3154, 2019.

[148] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018.

[149] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[150] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *CVPR*, pages 4685–4694, 2019.

[151] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Which training methods for gans do actually converge? In *ICML*, volume 80, pages 3481–3490, 2018.

[152] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, pages 3730–3738, 2015.

[153] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, 2007.

[154] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018.

[155] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, volume 30, pages 6629–6640, 2017.

[156] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230,000 3d facial landmarks). In *ICCV*, pages 1021–1030, 2017.

[157] Andrey Savchenko. Facial expression recognition with adaptive frame rate based on multiple testing correction. In *ICML*, volume 202, pages 30119–30129, 2023.

[158] Huiyuan Tian, Li Zhang, Shijian Li, Min Yao, and Gang Pan. Pyramid-vae-gan: Transferring hierarchical latent variables for image inpainting. *Computational Visual Media*, 9(4):827–841, 2023.

[159] Xiaoxing Zeng, Zhelun Wu, Xiaojiang Peng, and Yu Qiao. Joint 3d facial shape reconstruction and texture completion from a single image. *Computational Visual Media*, 8(2):239–256, 2022.

[160] Heng Wu, Kui Fu, Yifan Zhao, Haokun Song, and Jia Li. Joint self-supervised and reference-guided learning for depth inpainting. *Computational Visual Media*, 8(4):597–612, 2022.

[161] Zhongqi Wu, Jianwei Guo, Chuanqing Zhuang, Jun Xiao, Dong-Ming Yan, and Xiaopeng Zhang. Joint specular highlight detection and removal in single images via unet-transformer. *Computational Visual Media*, 9(1):141–154, 2023.

[162] Xuewei Bian, Chaoqun Wang, Weize Quan, Juntao Ye, Xiaopeng Zhang, and Dong-Ming Yan. Scene text removal via cascaded text stroke detection and erasing. *Computational Visual Media*, 8(2):273–287, 2022.

[163] Tao Wang, Kaihao Zhang, Xuanxi Chen, Wenhan Luo, Jiankang Deng, Tong Lu, Xiaochun Cao, Wei Liu, Hongdong Li, and Stefanos Zafeiriou. A survey of deep face restoration: Denoise, super-resolution, deblur, artifact removal. *arXiv preprint arXiv:2211.02831*, 2022.

[164] Utkarsh Ojha, Yijun Li, Cynthia Lu, Alexei A. Efros, Yong Jae Lee, Eli Shechtman, and Richard Zhang. Few-shot image generation via cross-domain correspondence. In *CVPR*, pages 10738–10747, 2021.

[165] Bingchen Liu, Yizhe Zhu, Kunpeng Song, and Ahmed Elgammal. Towards faster and stabilized GAN training for high-fidelity few-shot image synthesis. In *ICLR*, 2021.

[166] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected gans converge faster. In *NeurIPS*, volume 34, pages 17480–17492, 2021.

[167] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *CVPR*, pages 5747–5756, 2019.

[168] Xiangyu Chen, Xintao Wang, Jiantao Zhou, Yu Qiao, and Chao Dong. Activating more pixels in image super-resolution transformer. In *CVPR*, pages 22367–22377, 2023.

[169] Dafeng Zhang, Feiyu Huang, Shizhuo Liu, Xiaobing Wang, and Zhezhu Jin. Swinfir: Revisiting the swinir with fast fourier convolution and improved training for image super-resolution. *arXiv preprint arXiv:2208.11247*, 2022.

[170] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *CVPR*, 2022.

[171] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

[172] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*, volume 97, pages 6105–6114, 2019.

[173] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[174] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, pages 5967–5976, 2017.

[175] Ondřej Texler, David Futschik, Michal Kučera, Ondřej Jamriška, Šárka Sochorová, Menglei Chai, Sergey Tulyakov, and Daniel Sýkora. Interactive video stylization using few-shot patch-based training. *TOG*, 39(4):73, 2020.

[176] Weilun Wang, Jianmin Bao, Wengang Zhou, Dongdong Chen, Dong Chen, Lu Yuan, and Houqiang Li. Sindiffusion: Learning a diffusion model from a single natural image. *arXiv preprint arXiv:2211.12445*, 2022.

[177] Jitesh Jain, Yuqian Zhou, Ning Yu, and Humphrey Shi. Keys to better image inpainting: Structure and texture go hand in hand. In *WACV*, pages 208–217, 2023.

[178] Qiaole Dong, Chenjie Cao, and Yanwei Fu. Incremental transformer structure enhanced image inpainting with masking positional encoding. In *CVPR*, pages 11358–11368, 2022.

[179] MMEditing Contributors. MMEditing: OpenMMLab image and video editing toolbox. `https://github.com/open-mmlab/mmediting`, Sep. 2022. [Online; accessed 14-Feb-2022].

[180] Muhammad Moazam Fraz, Paolo Remagnino, Andreas Hoppe, Bunyarit Uyyanonvara, Alicja R. Rudnicka, Christopher G. Owen, and Sarah A. Bar-

man. An ensemble classification-based approach applied to retinal blood vessel segmentation. *IEEE Transactions on Biomedical Engineering*, 59(9):2538–2548, 2012.

[181] Zhangzhang Si and Song-Chun Zhu. Learning hybrid image templates (hit) by information projection. *TPAMI*, 34(7):1354–1367, 2012.

[182] Carl Doersch, Saurabh Singh, Abhinav Gupta, Josef Sivic, and Alexei A. Efros. What makes paris look like paris? *SIGGRAPH*, 31(4):101:1–101:9, 2012.

[183] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *TPAMI*, 40(6):1452–1464, 2018.

[184] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. In *NeurIPS*, volume 33, pages 7559–7570, 2020.

[185] Lu Chi, Borui Jiang, and Yadong Mu. Fast fourier convolution. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *NeurIPS*, volume 33, pages 4479–4488, 2020.

[186] Minyu Chen, Zhi Liu, Linwei Ye, and Yang Wang. Attentional coarse-and-fine generative adversarial networks for image inpainting. *Neurocomputing*, 405:259–269, 2020.

[187] Libin Jiao, Hao Wu, Haodi Wang, and Rongfang Bie. Multi-scale semantic image inpainting with residual learning and gan. *Neurocomputing*, 331:199–212, 2019.

[188] Long Chen, Changan Yuan, Xiao Qin, Wei Sun, and Xiaofeng Zhu. Contrastive structure and texture fusion for image inpainting. *Neurocomputing*, 536:1–12, 2023.

[189] Anpei Chen, Ruiyang Liu, Ling Xie, Zhang Chen, Hao Su, and Jingyi Yu. Sofgan: A portrait image generator with dynamic styling. *ACM Transactions on Graphics*, 41(1):Article 1, 2022.

[190] Jie Liu and Cheolkon Jung. Facial image inpainting using attention-based multi-level generative network. *Neurocomputing*, 437:95–106, 2021.

[191] Yiqian Wu, Yong-Liang Yang, Qinjie Xiao, and Xiaogang Jin. Coarse-to-fine: facial structure editing of portrait images via latent space classifications. *ACM Transactions on Graphics*, 40(4):Article 46, 2021.

[192] Junyan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. In *ECCV*, pages 597–613, 2016.

[193] Edo Collins, Raja Bala, Bob Price, and Sabine Süsstrunk. Editing in style: Uncovering the local semantics of gans. In *CVPR*, pages 5770–5779, 2020.

[194] Omer Kafri, Or Patashnik, Yuval Alaluf, and Daniel Cohen-Or. Stylefusion: Disentangling spatial segments in stylegan-generated images. *TOG*, 41(5), 2022.

[195] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4685–4694, 2019.

[196] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[197] Kihong Kim, Yunho Kim, Seokju Cho, Junyoung Seo, Jisu Nam, Kychul Lee, Seungryong Kim, and KwangHee Lee. Diffface: Diffusion-based face swapping with facial guidance. *arXiv preprint arXiv:2212.13344*, 2022.

[198] Wenliang Zhao, Yongming Rao, Weikang Shi, Zuyan Liu, Jie Zhou, and Jiwen Lu. Diffswap: High-fidelity and controllable face swapping via 3d-aware masked diffusion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8568–8577, June 2023.

[199] Yue-Ren Jiang, Shu-Yu Chen, Hongbo Fu, and Lin Gao. Identity-aware and shape-aware propagation of face editing in videos. *TVCG*, pages 1–12, 2023.

[200] Linzi Qu, Jiaxiang Shang, Xiaoguang Han, and Hongbo Fu. Reenactartface: Artistic face image reenactment. *TVCG*, pages 1–13, 2023.

[201] Xiaoguang Han, Kangcheng Hou, Dong Du, Yuda Qiu, Shuguang Cui, Kun Zhou, and Yizhou Yu. Caricatureshop: Personalized and photorealistic caricature sketching. *TVCG*, 26(7):2349–2361, 2020.

[202] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Cross-domain and disentangled face manipulation with 3d guidance. *TVCG*, 29(4):2053–2066, 2023.

[203] Wanchao Su, Hui Ye, Shu-Yu Chen, Lin Gao, and Hongbo Fu. Drawingin-styles: Portrait image generation and editing with spatially conditioned stylegan. *TVCG*, 29(10):4074–4088, 2023.

[204] Kelvin CK Chan, Xintao Wang, Xiangyu Xu, Jinwei Gu, and Chen Change Loy. Glean: Generative latent bank for large-factor image super-resolution. In *CVPR*, 2021.

[205] Ayush Kumar Tewari, Mohamed A. Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhöfer, and Christian Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. *CVPR*, pages 6141–6150, 2020.

[206] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *CVPR*, 2020.

[207] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark,

et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763, 2021.

[208] Fahad Shamshad, Muzammal Naseer, and Karthik Nandakumar. Clip2protect: Protecting facial privacy using text-guided makeup via adversarial latent search. In *CVPR*, pages 20595–20605, 2023.

[209] Nikola Popović, Ritika Chakraborty, Danda Pani Paudel, Thomas Probst, and Luc Van Gool. Spatially multi-conditional image generation. In *WACV*, pages 734–743, 2023.

[210] Xianxu Hou, Linlin Shen, Or Patashnik, Daniel Cohen-Or, and Hui Huang. Feat: Face editing with attention. *arXiv preprint*, arXiv:2202.02713, 2022.

[211] zllrunning. Face parsing in pytorch, 2023. [Online; accessed 14-Feb-2023].

[212] Yuming Jiang, Ziqi Huang, Xingang Pan, Chen Change Loy, and Ziwei Liu. Talk-to-edit: Fine-grained facial editing via dialog. In *ICCV*, pages 13799–13808, 2021.

[213] Hao Hu, Xuan Wang, Jingxiang Sun, Yanbo Fan, Yu Guo, and Caigui Jiang. Vectortalker: Svg talking face generation with progressive vectorisation. *arXiv preprint*, arXiv:2312.11568, 2023.

[214] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.