UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Image inpainting in latent space

Author:
Ihor BABIN

Supervisor:
Roman RIAZANTSEV
James PRITTS

A thesis submitted in fulfillment of the requirements
for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2024

Declaration of Authorship

I, Ihor BABIN, declare that this thesis titled, "Image inpainting in latent space" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Master of Science

Image inpainting in latent space

by Ihor BABIN

Abstract

This thesis introduces a framework in which training image autoencoders by applying losses in latent space improves the quality of decodings and can significantly decrease training time. Furthermore, within this framework, we propose a mask guidance mechanism that combines mask features and image features in the early phases of image encoding, which supports the encoder in reconstructing the image embeddings. These methods are demonstrated in the context of the ill-posed image inpainting problem, which seeks to reconstruct regions of the image that have been occluded by a mask. We show that latent loss application results in more naturally inpainted textures when used in a state-of-the-art inpainting architecture. We also show that a mask-controlled embedding gives superior results across every common inpainting metric when compared to a state-of-the-art approach, which provides mask conditioning only in the image space. The final component of our study involves the visualization of latent space to highlight damaged areas of features that need refinement.

Acknowledgements

I am deeply grateful to my supervisors, *Roman Riazantsev* and *James Pritts*, for their significant contributions in developing numerous valuable hypotheses, ideas, and suggestions throughout the thesis and for their consistent support.

In addition, we extend our gratitude to *ADVA Soft* for their outstanding support. Supplying high-performance GPUs for our research significantly speed up our projects and improved our computational efficiency.

Contents

List of Figures

List of Tables

List of Abbreviations

LaMa	Resolution-robust **La**rge **Ma**sk Inpainting with Fourier Convolution
RGB	**R**ed **G**reen **B**lue
SD	**S**table **D**iffusion
CNN	**C**onvolutional **N**eural **n**etwork
GAN	**G**enerative **A**dversarial **n**etworks
FID	**F**réchet **I**nception **D**istance
LPIPS	**L**earned **P**ercuptual **I**mage **P**atch **S**imilarity
SSIM	**S**tructure **Sim**ilarity
FFC	**F**ast **F**ourier **C**onvolution
FcF	**F**ourier **C**oarse-to-**F**ine

Dedicated to the Armed Forces of Ukraine

Chapter 1

Introduction

1.1 Motivation

Image inpainting [1.1] is the task of completing missing data in a designated region, called the mask, of the input image. Although the task is simple to state, it is an open problem in computer vision because of its difficulty, which has been developed over the past two decades.

Inpainting techniques have faced the challenge of filling missing or damaged parts of images. Although early methods such as PatchMatch [Barnes et al., 2009] were effective in duplicating patterns, they frequently lacked sufficient semantic details for intricate image textures, motivating the need for learned representations that can richly model semantics, texture, and scene structure.

FIGURE 1.1: Illustraion of the inpainting task. Mask highlighted in violet for visualization purposes. Inpainting outcome is displayed on the right. The first row shows challenges with repetitive textures, while the second row demonstrates the necessity for semantic understanding. [Suvorov et al., 2022]

1.2 Task definition

Several significant advances have marked the evolution of image inpainting, reflecting the field's movement from early classical methods to sophisticated deep learning

techniques and diffusion-based techniques. Notably, inpainting is an ill-posed problem, meaning that there is a distribution of plausible outputs for masked regions and damaged regions. To contend with this inherent challenge, most methods treat the inpainting task with a model-based approach, which we also adopt.

By model-based approach, we mean that we create a latent space such that a masked image is embedded with inpainting that is highly likely given the unmasked input image content and similar image content from the training data set that was used to build the model weights. In other words, by feasible inpainting, we want the inpainted region of the image to reflect the statistics of the input image and the corpus of images that were used to train the model. Ideally, the unmasked region should be invariant to the inpainting process. While the details of how to measure plausible inpainting are method-specific, several necessary conditions for plausibility can be defined: the inpainted image should look like the undamaged image (if available); the inpainting should preserve the semantics and geometric structure of the unmasked region; the inpainting should incorporate both local and global image statistics from the input image; and one inpainting result must be selected from many feasible inpainting outcomes (as opposed to blurring or averaging them). State-of-the-art inpainting methods have losses that consider all of these factors.

In practice, we do not always have an exact ground truth image, so instead of a strict reconstruction loss, losses such as FID [Heusel et al., 2017], LPIPS [Zhang et al., 2018] and SSIM [Brunet, 2012] are used by state-of-the-art methods that address the conditions for feasible inpainting. Furthermore, the use of reconstruction loss alone is not sufficient to disambiguate a mode from the distribution of feasible inpaintings.

The introduction of deep learning, especially CNN [LeCun, Bengio, et al., 1995] and GAN [Goodfellow et al., 2014], makes a big step in quality for image inpainting. CNNs enhanced the ability to generate contextually coherent images as a result of the learned semantically meaningful information, while GANs with their adversarial loss allow one to account for both the content and the probabilistic nature of inpainting to restore missing information. However, these methods concentrate on optimizing inside output's pixel space - using distance metrics, adversarial, and perceptual losses, making them harder to train and resource intensive.

1.3 Overview of Proposed Approach

The proposed approach focuses on restoring areas of the latent space associated with the image's damaged regions, eliminating the necessity to decode the entire image from the latent space. Within this space, the refinement process receives a better signal from damaged areas in latent space throughout the entire image, thereby enhancing both efficiency and accuracy. The primary element of this method is the latent loss, which acts as a measure to evaluate the distance between the features of the original and refined images. Furthermore, our framework proposes a mask guidance mechanism that combines mask features and image features in the early phases of image encoding,

1.4 Research questions and contribution

This thesis explores two main research questions. The first research question explores the architecture of the latent space to determine if the autoencoder configuration can reveal details about the damaged area. This was examined by analyzing the

difference between the features of the ground truth and those of the masked image within the latent space.

The second question examines whether the speed of training can be increased and the quality of inpainting models preserved by adding additional guidance within the latent space before features are decoded into pixel space. The initial guidance comes from mask guidance, adapting the control mechanism of ControlNet [Zhang, Rao, and Agrawala, 2023]. This method is implemented on top of the frozen encoder to maintain the fidelity of the unmasked areas, as the embedding is conditioned on the mask and these areas are expected to remain invariant to the inpainting task. The second guidance is derived from latent loss, which pushes the refined features closer to the ground-truth features, anticipating that they will be more accurate after being decoded in the feature space, thus providing a stronger signal than using the pixel space loss. Further experiments confirm that applying these enhancements to a recognized baseline model results in better performance and reduced training duration.

1.5 Structure of thesis

Chapter 2 of this thesis presents related work and forms the foundation of the study. Subsequently, Chapter 3 outlines the configuration of inpainting datasets and evaluation metrics. Chapter 4 details our method and motivation for this. Chapter 5 discusses our experimental procedures, culminating in the conclusion of the thesis in Chapter 6.

Chapter 2

Related Work

This section reviews seminal inpainting methods. The methods include both discriminative and generative models. Notably, both classes of methods must explicitly model the fact that there is a distribution of plausible inpaintings, which is encoded in a variety of losses introduced in the literature. Failure to do so results in blurred inpaintings. We review these losses and discuss their relevance to learning an interpretable latent space.

2.1 Classical Methods

2.1.1 PatchMatch

In its initial stages, PatchMatch [Barnes et al., 2009] laid the foundation and achieved notable quality in this task. PatchMatch [2.1] uses a stochastic very large neighborhood search to match patches within an image, which proved effective in texture-rich scenarios, but less so in complex contexts. It works by initially assigning random correspondences between patches in the missing and intact regions. Then, an iterative process refines these matches to align the textures and patterns more closely. The strength of PatchMatch lies in its ability to quickly and effectively copy and paste these patches to fill in missing regions, making it highly effective for images with repetitive textures.

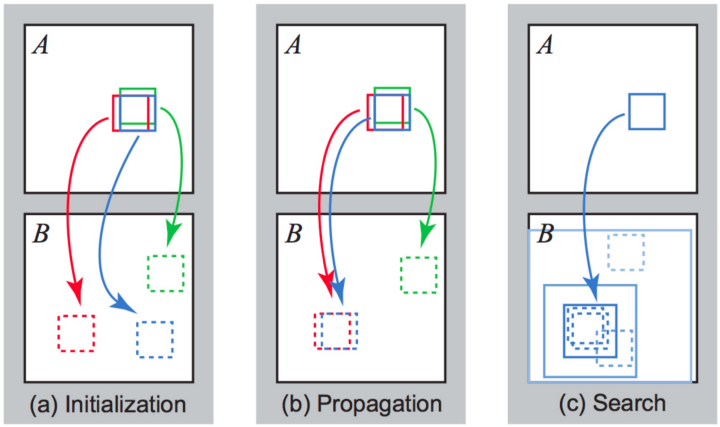


FIGURE 2.1: Illustration of the PatchMatch algorithm. A represents a masked image in which the masked patches are ignored and B is the resulting image. They demonstrate the construction of random patch initialization with subsequent propagation to neighboring patches in (b) and a random search for improved correspondences in (c).

FIGURE 2.2: Example of PatchMatch limitation due to insufficient semantic understanding: the algorithm was unable to complete or properly extend the bench lines, incorrectly taking parts from inappropriate areas.

However, the algorithm had limitations in handling more complex scenarios [2.2]. It lacks the ability to understand and interpret the context or semantics of the image. This shortfall was particularly evident in images with large missing areas or those containing unique, non-repetitive elements. In such situations, simple copying and pasting of patches often led to visually disjointed or contextually inappropriate results. Although efficient for certain textures, the reliance on randomized matching failed to account for the nuanced comprehension required for more sophisticated and coherent image restoration, underscoring the need for advanced methods with a deeper understanding of image content.

2.2 Deep Learning Based Methods

2.2.1 Context Encoders

After the introduction of CNN, the era of deep learning-based image inpainting methods began. One of the first successful methods was the Context Encoders [2.3] [Pathak et al., 2016], which laid the baseline and general approach to many follow-up works: use CNN to encode the features of the image in a learnable way, then process it in a downscaled feature space, and decode it using a learnable CNN decoder.

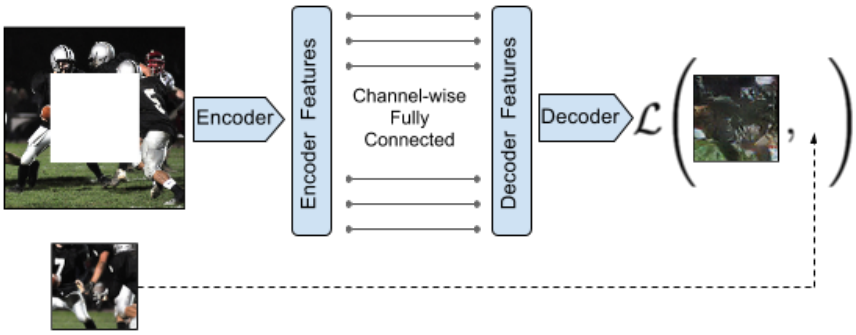
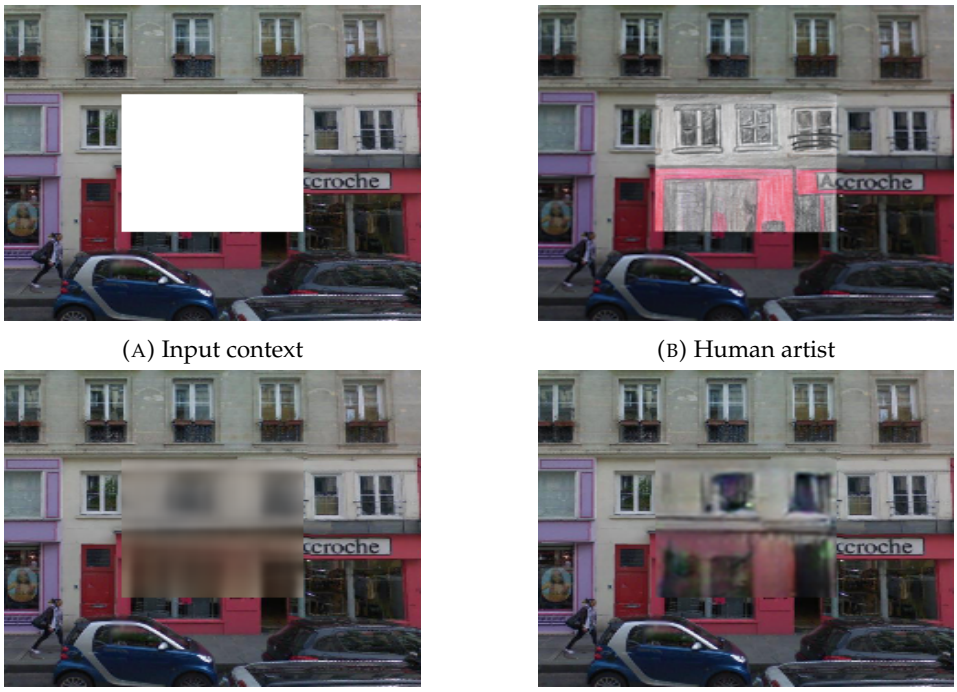


FIGURE 2.3: Diagram of the Context Encoders pipeline. An encoder embeds the image into a feature space, followed by fully connected layers that refine these features, and a decoder that reconstructs the image in pixel space. [Pathak et al., 2016]

Another significant advancement was in the exploration of loss functions for training models based on inpainting [2.4]. Previously, many methods employed distance-based losses such as $L1$ or $L2$. These, however, tended to result in blurriness and color mismatches (or their averaging) in image-to-image translation tasks [Pathak et al., 2016]. Advancements in GANs have improved the inpainting results of Content Encoders by incorporating adversarial loss [Goodfellow et al., 2014], a type of loss derived from the output of the discriminator, which is trained concurrently with the main network (generator), alongside distance loss.



(A) Input context

(B) Human artist

(C) Context Encoder ($L2$ loss)

(D) Context Encoder ($L2$ + Adversarial loss)

FIGURE 2.4: Comparison of losses.(a) Input, (b) a human artist inpainting it. Automatic inpainting using the *context encoder* trained with the loss of reconstruction $L2$ is shown in (c), and using both $L2$ and adversarial losses in (d). [Pathak et al., 2016]

2.2.2 Perceptual Loss

Besides distance and adversarial loss, most deep learning methods incorporate semantic-based losses. Presented for tasks of style transfer and super resolution, perceptual loss [Johnson, Alahi, and Fei-Fei, 2016] also improves the quality of inpainting networks. The core idea of this loss is to compare the correlation between the ground truth image and the predicted image using some pre-train feature extractor [2.5], usually VGG-16 is used, which is proven to extract meaningful semantic information of the image. One of drawbacks of this loss is the need of some external network to extract semantic features of predicted image, and not utilizing inpainting network's inner features.

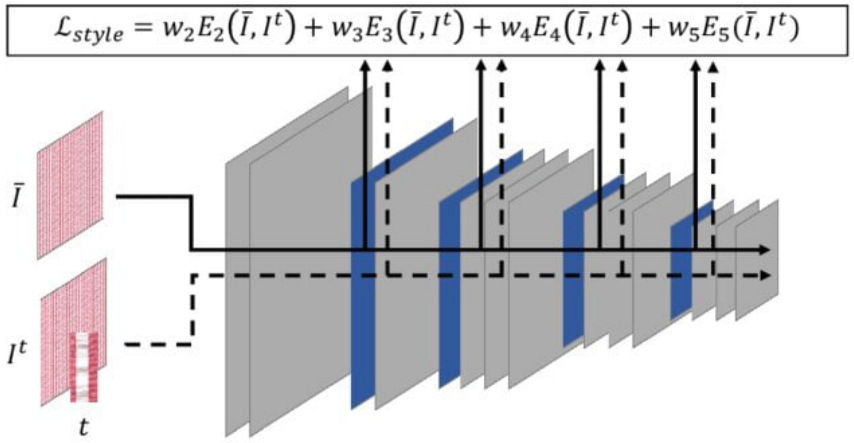


FIGURE 2.5: Illustration of Perceptual loss pipeline. Features are extracted from various depth levels using VGG-16 from both the ground truth and predicted images, and the loss is computed as the correlation among these features. [Rodríguez Pardo et al., 2019]

2.2.3 LaMa

Most previous approaches used the standard convolution block, with some modifications of the dilation in the convolution block. But in LaMa [Suvorov et al., 2022] the authors argue that a receptive field is essential for inpainting task. In LaMa model [2.6] incorporated Fast Fourier Convolution [Chi, Jiang, and Mu, 2020] blocks to more effectively handle large missing areas in images by capturing a broad spectrum of image frequencies. These FFC blocks, which utilize Fourier transformations, excel in processing large missing areas, enabling detailed and accurate reconstruction to maintain textural integrity and visual coherence, especially where traditional convolutional methods fail. LaMa's proficiency in managing high-frequency details makes it particularly adept for high-resolution images, preserving finer details up to 1024-pixel resolution.

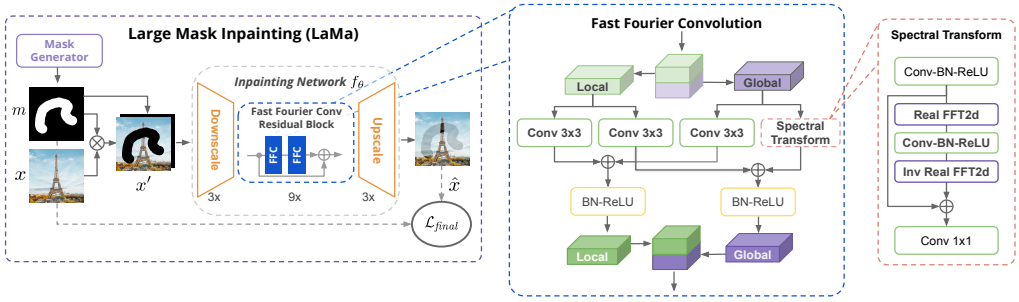


FIGURE 2.6: Structure of LaMa: A mask is concat with the masked image, then passed through downscaling (or encoding) layers, followed by a sequence of FFC blocks. These blocks include a local branch that performs convolution in the feature space and a global branch that applies convolution on features converted to Fourier space, making convolution's receptive field global related to features. [Suvorov et al., 2022]

However, it encounters limitations beyond this, needing help maintaining sharpness and clarity at higher resolutions, often resulting in a perceptible loss of fine details and blurriness. This challenge highlights a significant hurdle in current inpainting methods: effectively scaling the inpainting process to very high resolutions while retaining the quality and fidelity of intricate image details. An effort to achieve high-resolution inpainting was explored in the Guided PatchMatch [Zhang et al., 2022] paper, where the authors suggest directing the patch selection process from PatchMach with a novel guidance approach: utilizing LaMa results at 512 scale for pixel information and employing depth, structure, and segmentation for patch selection [2.7]. Although this approach improves quality at high resolutions, it still inherits the limitations of both methods and faces significant challenges in choosing appropriate guidance and requires manual selection method of best guidance (Curration Module).

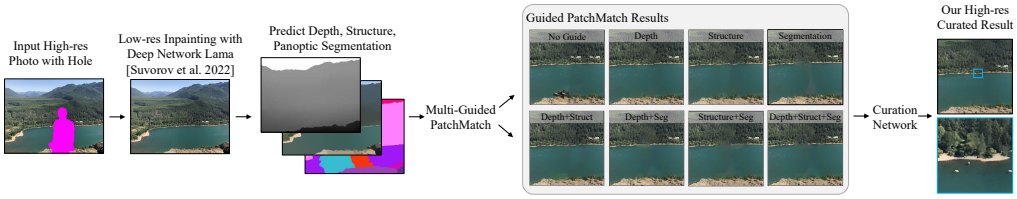


FIGURE 2.7: Pipeline of Guided PatchMatch of different guidance with selection of best result trough Curration Network. [Zhang et al., 2022]

2.2.4 LaMa refiner

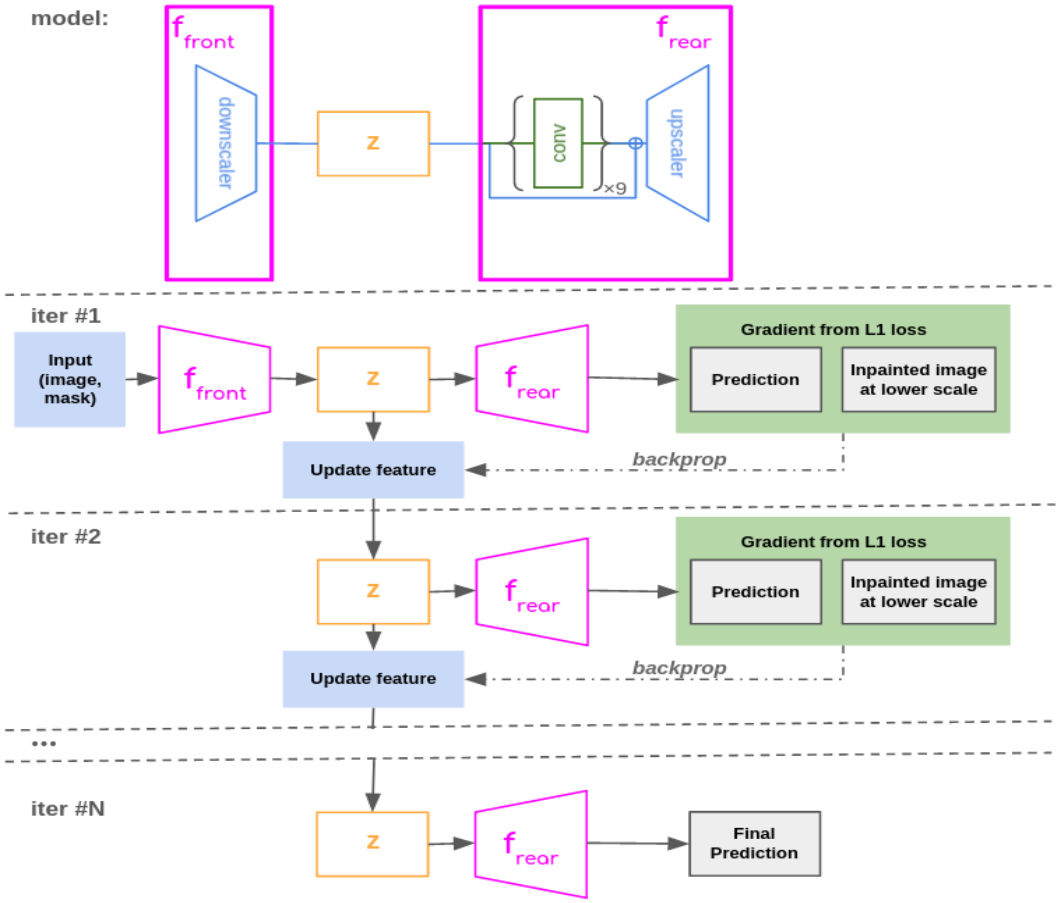


FIGURE 2.8: Architecture of LaMa refiner [Kulshreshtha, Pugh, and Jiddi, 2022]

Building on the foundation of the LaMa model, the LaMa Refiner [Kulshreshtha, Pugh, and Jiddi, 2022] introduces a specialized approach to enhance high-resolution image inpainting. This advancement capitalizes on the observation that inpainting at lower resolutions typically yields sharper results with less blurriness. The LaMa Refiner [2.8] employs a dual-resolution strategy to leverage this insight while simultaneously considering a lower-resolution image and the corresponding high-resolution features. The core of this method lies in aligning these two aspects: the high-resolution image's latent features are optimized online, meaning during each inference, to minimize the L1 loss between the reconstructed lower-resolution image and its high-resolution counterpart. This process aims to transfer the accuracy achieved at lower resolutions to the higher-resolution image [2.9], effectively reducing the blurriness that often plagues high-resolution inpainting. The LaMa Refiner thus represents a novel step in image inpainting, focusing on the intricate balance and optimization of latent features across different resolutions to enhance the overall quality of inpainted images in high-resolution settings.

FIGURE 2.9: From left to right, first row: (i) input image, (ii) inpainting at size 512, (iii) inpainting at size 1024 (iv) inpainting at 1024 with LaMa Refinement. Second row: zoomed-in corresponding inpainted areas. [Kulshreshtha, Pugh, and Jiddi, 2022]

2.2.5 FcF

A key feature of inpainting output is its diversity of results. Typically, there is no straightforward ground truth available; instead, there exists a range of potential outcomes, making it challenging to determine the optimal one. A common approach, as utilized in LaMa, involves employing discriminator networks to ensure visual accuracy of the output, a distance metric to reconstruct known regions, and perceptual loss to maintain the semantic integrity of the entire image. In the FcF [Jain et al., 2023] paper, the authors suggest the use of standard Gaussian noise, mapped through the Mapping Network (StyleGan2 [Karras et al., 2020] is used) as mean and variance, for Instance Normalization [Ulyanov, Vedaldi, and Lempitsky, 2016] (instead of Batch Normalization) within the model.

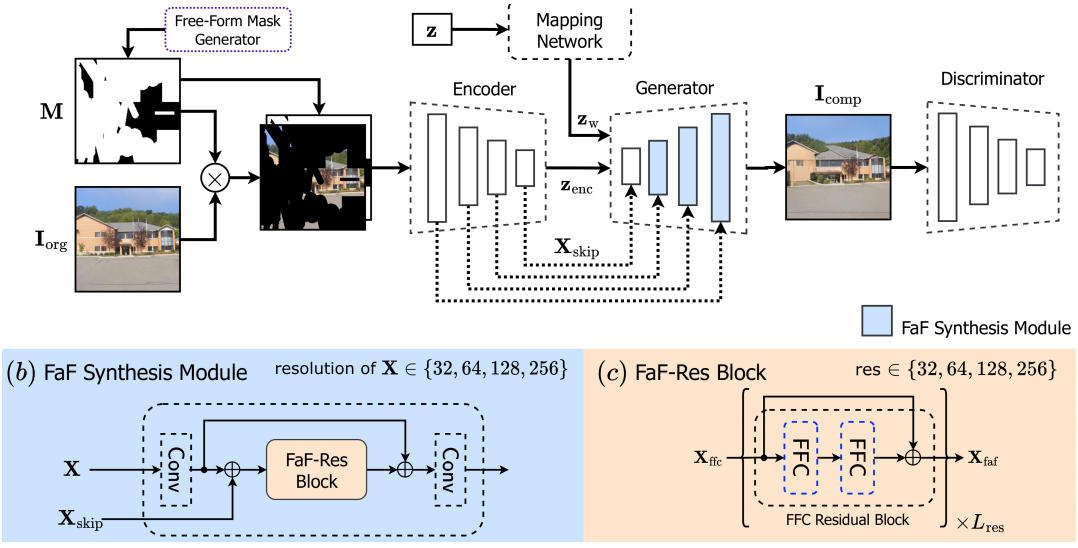


FIGURE 2.10: Architecture of FcF [Jain et al., 2023]

However, introducing noise might lead to overly unpredictable model output, potentially resulting in too novel and unexpected results [2.11] (or to hallucinate).

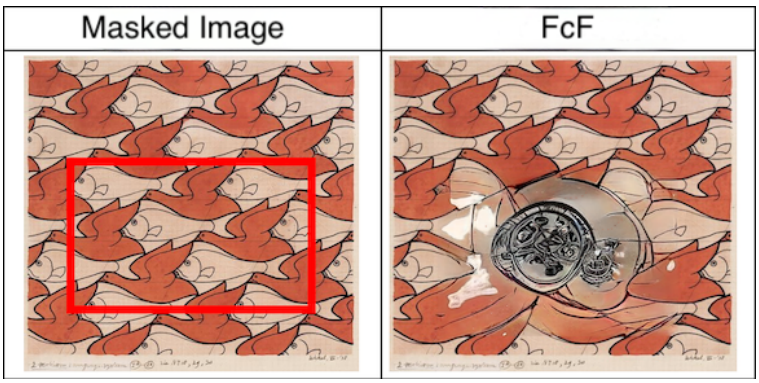


FIGURE 2.11: Illustration of hallucination for noise guided network [Cipolina-Kun, Caenazzo, and Mazzei, 2022]

2.3 Diffusion-based

Recent progress in text-to-image synthesis using conditional diffusion-based methods has surpassed [Dhariwal and Nichol, 2021] GAN-based approaches, redirecting research focus towards them. The primary distinction between these approaches lies in their handling of noise [2.12]. Whereas GANs attempt to create images directly from pure noise in a single forward pass, diffusion models employ a stochastic process to iteratively remove noise from an image. Initially, these models operate with high-variance noise, enabling the generation of coarse image details. As the process progresses, the noise variance is reduced, allowing for the refinement of finer image details.

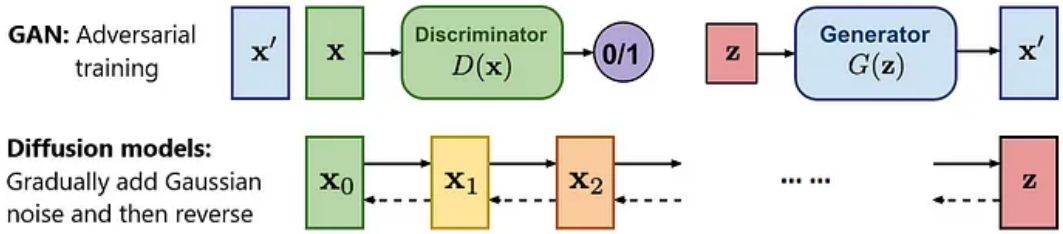


FIGURE 2.12: Comparison of GAN and diffusion models. In the GAN framework, noise z is sampled once and converted into image x', followed by the discriminator's assessment of whether the image originates from the actual distribution or a synthetic one. Conversely, in the diffusion model, noise of varying variances is sampled at each step, and the model attempts to reverse this noise. [Jain et al., 2023]

2.3.1 GLIDE

The initial approach to conditional inpainting was introduced in the GLIDE paper. This model uses the execution of a standard diffusion process within pixel space. It does not require any architectural modifications to the model to facilitate the inpainting task; however, it needed a modification to the diffusion process. In the denoising approach, the predicted noise is subtracted from the noisy image to generate the full image. In contrast, for inpainting, we need to generate only some part

of image, thus, the predicted noise is applied solely to the masked areas, whereas the remaining areas use generated known noise, for saving known parts of image.

2.3.2 Stable Diffsion

Earlier pixel-based diffusion models were capable of achieving acceptable outcomes, but operating directly with pixels required substantial resources and extended inference times. This requirement increased with the number of diffusion steps, leading to resource-intensive training and inference that make these models impractical.

In the Stable Diffusion(SD) [2.13] paper [Rombach et al., 2021], the authors introduce the concept of latent diffusion. The core concept involves transitioning the diffusion process from a pixel space to a latent space, which is facilitated by VQ-VAE [Van Den Oord, Vinyals, et al., 2017]. Using encoded features that reduce spatial dimensions by a factor of eight, the network can handle images with a resolution of 512 without extensive resource demands. Crucially, within this latent space, the image information is semantically compressed, enabling the diffusion process to focus primarily on generating high-quality semantic content of the image, with subsequent reliance on the VQ-decoder for information reconstruction, enabling even faster model and better image generation.

Standard SD is primarily used for generating images from text, but it can also be adapted for inpainting. Initially, the GLIDE method is used; however, it tends to produce suboptimal outcomes because its denoising process prioritizes semantic content over pixel accuracy. A superior approach involves integrating the image features from VQ-VAE with a downscaled mask and refining the model.

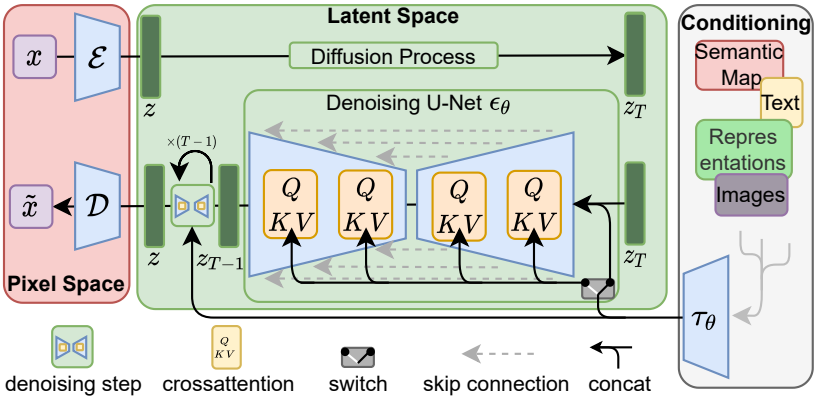


FIGURE 2.13: Illustration of the Latent diffusion model (or SD). On the left, there is an encoder-decoder setup for projecting to and from pixel space. In the center, a diffusion denoising network is shown, featuring various types of conditioning on the left. [Rombach et al., 2021]

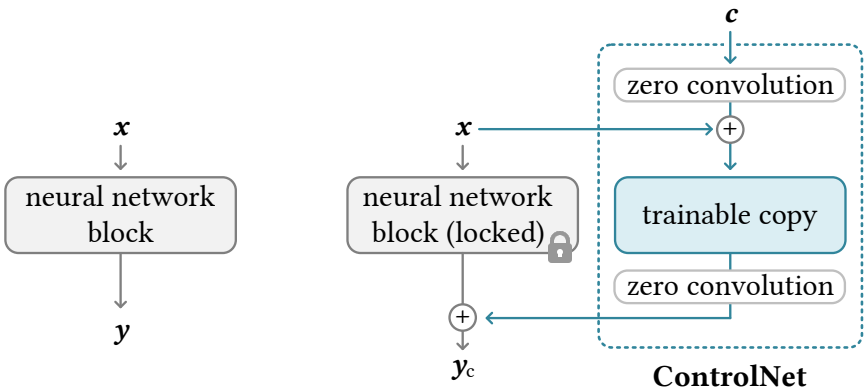
Notably, Stable Diffusion tends to underperform [2.1] relative to the Big LaMa version (an enhanced LaMa with more FFC blocks), primarily due to its approach to noise guidance similar to that of FcF. This large deviation from the average noise can cause the model to hallucinate, resulting in outputs that are less semantically meaningful.

	40-50% masked		All samples	
Method	FID ↓	LPIPS ↓	FID ↓	LPIPS ↓
SD	**9.39**	0.246± 0.042	**1.50**	0.137± 0.080
LaMa	12.31	**0.243**± 0.038	2.23	**0.134**± 0.080

TABLE 2.1: Comparison of inpainting results between Stable Diffusion and LaMa. LaMa outperforms in LPIPS, indicating superior semantic integrity. Stable Diffusion excels in FID, as FID assesses the statistical variance in image features, influenced by the varied noise instances in the SD model. [Rombach et al., 2021]

2.3.3 ControlNet

Another significant technique in the realm of SD is ControlNet [Zhang, Rao, and Agrawala, 2023]. This method enables the incorporation of extra guidance elements such as text or masks into the diffusion denoising network, thus eliminating the requirement to retrain the large model for different guidance types. The primary strategy employed is the 'zero-convolution' method, which involves convolutions that start with weights set to zero [2.14]. It is crucial to avoid random weight initialization as it can significantly affect the main frozen layers, potentially complicating the initial stages of training. The zero convolution method allows for the gradual integration of information into the main branch, thereby subtly modifying the network output in response to the guidance provided.



(a) Before (b) After

FIGURE 2.14: Illustration of ControlNet modification [Zhang, Rao, and Agrawala, 2023]

2.4 Conclusion of related work

In this chapter, we explore key developments in the field of inpainting techniques, beginning with the traditional copy-paste method of PatchMatch. Subsequently, this approach was supplanted by contextual encoders, which were enhanced through the incorporation of several critical losses: reconstruction, adversarial, and perceptual. These, combined with the significant inductive biases presented in the LaMa paper, currently establish the state-of-the-art. Additionally, we examine a concurrent approach involving diffusion-based inpainting that proposes to operate in latent space, a core element of this thesis. For our solution, LaMa model was used as strong inpainting baseline. Latent loss was motivated by LaMa Refiner work, but differs in autoencoder setup and in one-time inference. Mask Control inspired by ControlNet and changes the default alpha channel configuration.

Chapter 3

Problem setting

In this section, we outline the dataset configuration for the upcoming experiment and detail the evaluation metrics.

3.1 Data setup

The Places2 [Zhou et al., 2016] dataset is a comprehensive scene-centric database that contains more than 10 million images that span more than 400 unique scene categories [3.1]. It is designed for training and evaluating scene understanding models and is widely used in various computer vision tasks, including scene recognition, image synthesis, and inpainting. This dataset has become a standard for training, as it enables the generation of a significantly higher number of mask samples, which are independent of image distribution, thus facilitating the training of robust models.

FIGURE 3.1: Samples from Places2 dataset [Zhou et al., 2016]

In the LaMa paper, the mask creation process [3.2] is crucial for the model's image inpainting capabilities. This approach consistently applies samples from polygonal chains expanded with a randomly large width (wide masks) and rectangles with various aspect ratios (box masks).

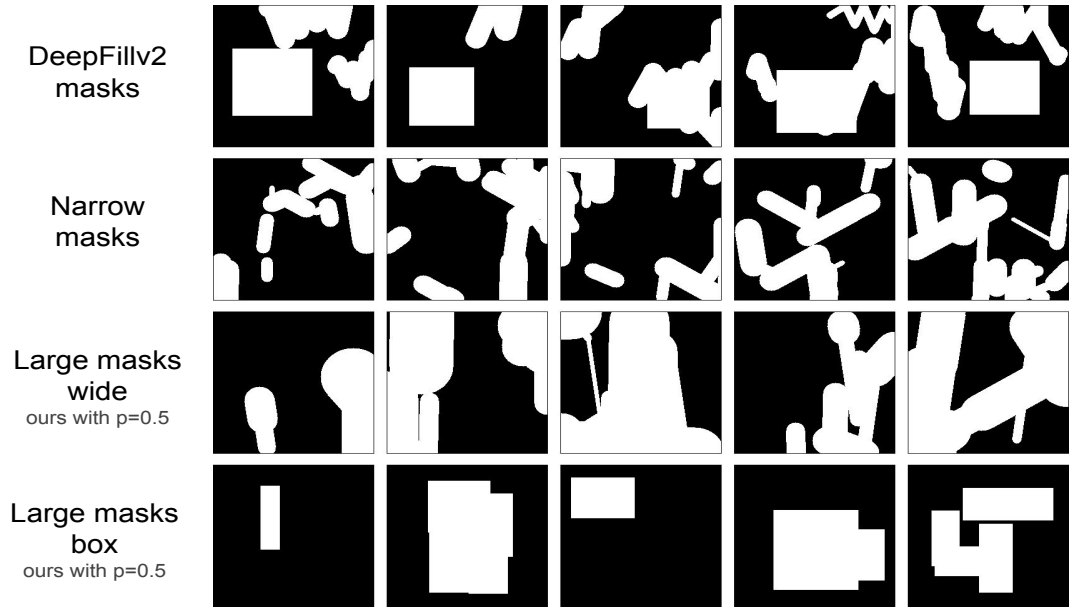


FIGURE 3.2: Mask generation strategy [Suvorov et al., 2022]

Crucially, in the inpainting task, there isn't just one definitive ground truth image; instead, there exists a variety of possible outcomes. Therefore, the model cannot depend entirely on a single result. Typically, generating a wide array of random masks and using the original image for verification is adequate to train the model, and this approach is considered standard in many networks.

3.2 Evaluation

To quantitatively assess the experiments, we will utilize standard metrics commonly employed in inpainting research, namely FID and LPIPS. Additionally, due to the implementation of experiments focusing solely on pure distance loss training, we will also incorporate SSIM.

The Fréchet Inception Distance (FID) is a metric used to evaluate the quality of images generated by models, particularly in the field of GANs. It measures the similarity between two sets of images, typically generated images and real images, by comparing the statistics of their features extracted by the Inception v3 model. The FID is calculated by computing the Fréchet distance between two multivariate Gaussians, defined by the mean and covariance of the feature sets from the real and generated images. The formula for FID is given by:

$$FID = ||\mu_r - \mu_g||^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}) \tag{3.1}$$

Where μ_r and μ_g denote the mean values of the features for real and generated images along each dimension, and Σ_r and Σ_g are the variances of these features

LPIPS (Learned Perceptual Image Patch Similarity) is a metric used to assess the perceptual similarity between images. It is designed to better align with human judgment compared to traditional metrics such as MSE or PSNR. LPIPS computes the distance between deep features extracted from a pre-trained neural network(usually VGG-19), reflecting the perceptual differences perceived by humans. The formula for LPIPS is given by the following:

$$LPIPS(x_r, x_g) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} ||f_l^{x_r}(h,w) - f_l^{x_g}(h,w)||_2^2 \tag{3.2}$$

Where f denotes the feature extractor.

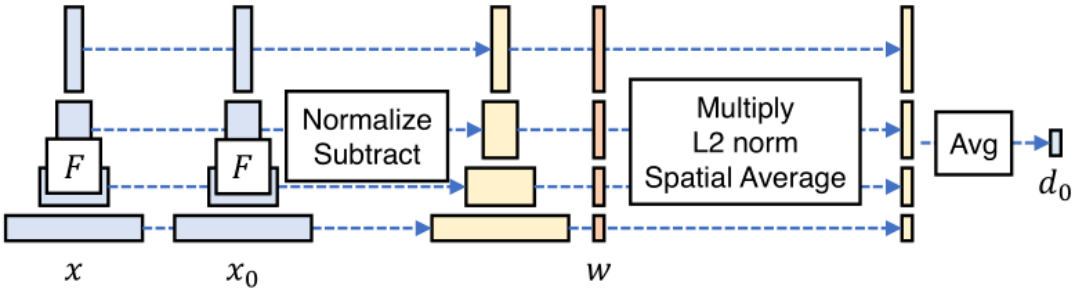


FIGURE 3.3: LPIPS metric pipeline. F represents a network designed to extract features from images x and x_0, which then normalizes and subtracts these features, followed by a process of multiplication and averaging. [Zhang et al., 2018]

The Structural Similarity Index (SSIM) is a method to measure the statistical similarity between two images. SSIM is designed to improve traditional methods such as peak signal-to-noise ratio (PSNR) and mean squared error (MSE), which have proven to be inconsistent with human eye perception.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{3.3}$$

Chapter 4

Proposed Solution

4.1 Framework description

In this section, we introduce a proposed framework [4.1] in the context of the original LaMa model. However, it is not restricted to this model alone and could potentially be adapted to any model featuring an encoder-decoder architecture with an intermediary refinement block.

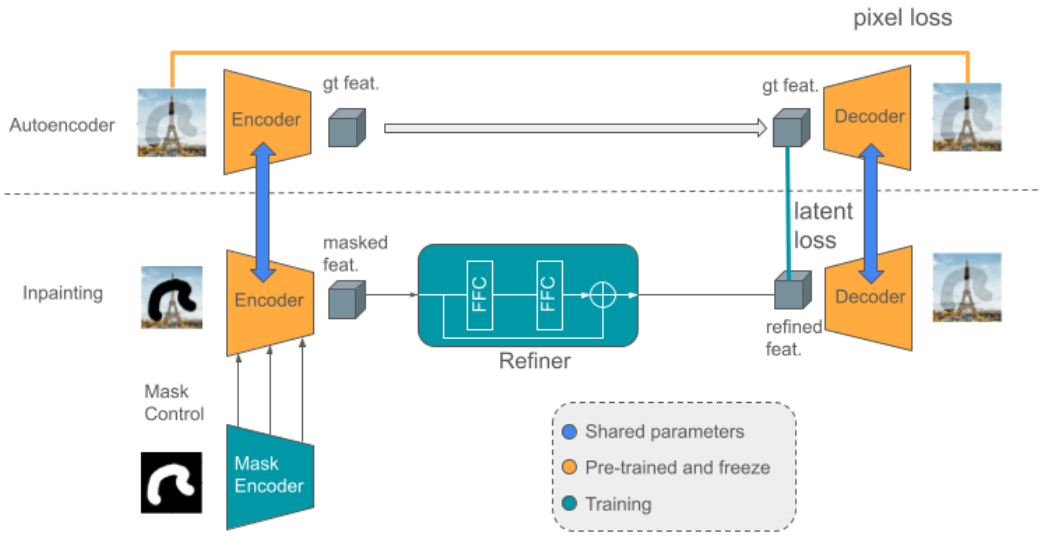


FIGURE 4.1: Proposed framework. Initial pre-training phase of an autoencoder employs MSE loss at the top. Below the dashed line lies the core pipeline of the LaMa model integrated with our framework. The encoder-decoder shares weights with the autoencoder and is frozen. A Mask Encoder is incorporated to derive features from the mask image to condition the image encoder. Subsequently, a Refiner processes these features and performs inpainting within the latent space, succeeded by a latent loss computation (L1 distance between the refined features and the actual ground truth features).

4.1.1 Mask Control

An important component of the image encoder is its usage of mask data. In the original LaMa model, pixels where a mask value is one are zeroed out, with the mask being concatenated as the fourth alpha channel in the encoder input. A different masking technique in SD is proposed that utilizes an autoencoder that uses a three-channel input (RGB image). The authors opted to concatenate the encoded features at the latent space level rather than at the pixel level. A significant advantage of SD's compared to previous diffusion-based methods, such as GLIDE, is that they are pre-train and freeze autoencoder during the costly denoising training, thus significantly

accelerating the process, although they do not utilize the mask most effectively. In our approach, we aim to retain the advantage of the pre-trained autoencoder while enhancing its mask handling capabilities. Inspired by ControlNet, which was the primary design to keep denoising part of the model frozen, we implemented a control mechanism to direct the image encoder with mask data, eliminating the need to unfreeze the encoder.

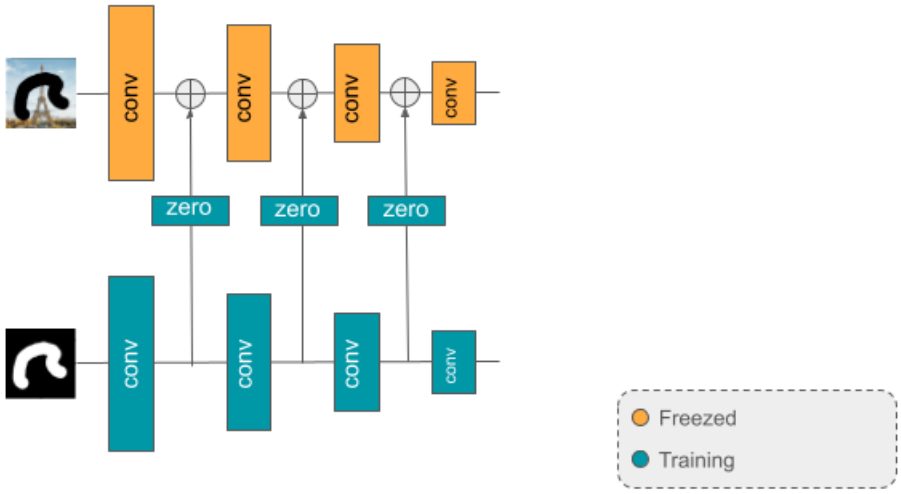


FIGURE 4.2: Diagram of the Mask Control. 'Conv' refers to convolution blocks, and 'zero' refers to convolutions where the weights are initially zero. Each convolution block does not need to be identical, but should yield an output of the same shape.

In our approach, the mask encoder is designed to generate intermediate features on various scales [4.2]. It is designed to ensure that the dimensions of these features, both spatial and channel-wise, align with those of the image encoder at specific intermediate points. Following this, the features undergo convolution and are merged with the image encoder's intermediate features, moving forward to the next layer of the image encoder. The adoption of 'zero-convolution' kernels aids in the gradual merging of data from the mask encoder features, which helps to prevent the image embeddings from being immediately influenced by the initially noisy kernels. The fundamental reason for employing this mask integration is as follows: the image encoder is adept at extracting semantically rich information; however, applying zeros to masked regions would compromise the semantic integrity within the receptive fields of these areas, thereby rendering the semantics less informative. Our method aims to restore these areas with the help of mask embeddings. Subsequent experiments demonstrate the benefits of this approach. Remarkably, while conducting these experiments, we identified a concurrent study of motion comprehension [Shi et al., 2024] within the latent space that also utilizes this type of mask control, confirming its versatility.

4.1.2 Autoencoder

Initially, the establishment of a latent space through an autoencoder configuration is essential. This setup facilitates the creation of a latent space capable of being decoded into a meaningful representation. Crucially, the encoder function acts as the inverse transformation of the decoder, implying that an image can be transformed

into features and these features can be decoded back into the original image [4.3]. Encoding the ground truth image (expected inpainting outcome) in this space is expected to result in a refined feature that closely matches the encoded ground truth feature. This crucial observation enhances our framework by reducing computational demands as it avoids the extensive decoding process and pixel-level manipulation.

Consequently, the first step in our methodology involves setting up the encoder-decoder model to operate as an autoencoder. This setup typically involves self-supervised training [Balestriero et al., 2023] to allow the model to both compress and decompress the data, effectively reproducing the original input as the output. Training uses a set of image samples fed into the network, with the final MSE loss calculated between the input and output images to facilitate the learning process of image compression and decompression.

It is crucial to note that the SD autoencoder - VQ-VAE [Van Den Oord, Vinyals, et al., 2017] is pre-trained and frozen, and it does not employ any losses before decoding into pixel space. In our experiment configurations, we opt to pre-train a standard encoder-decoder architecture from LaMa, allowing for a direct comparison with the LaMa baseline, which remains a strong benchmark for the inpainting task; however, it remains comparable to SD inpainting as the SD paper includes direct comparisons and metrics with the LaMa model.

4.1.3 Latent Loss

The most critical and novel aspect of the framework is learning in latent space [4.3]. Facilitated by the autoencoder configuration, we target both the ground truth feature and the refined feature for optimization. After decoding both the ground truth and the refined feature, we aim to generate images that closely resemble each other. To achieve this, we have chosen the L1 distance metric, aligning with the LaMa pixel L1 loss and inspired by LaMa Refiner's selection of L1 for their optimization. However, the choice of measurement is not limited to this metric. Therefore, we expect the features to be identical before decoding (since the decoder is unchanging), which justifies the use of this particular loss. Additionally, the rationale for this is the superior signal derived from latent loss compared to pixel loss, which must pass through the decoder layer.

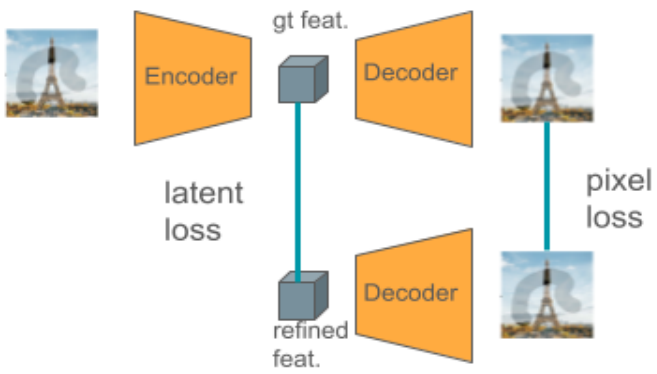


FIGURE 4.3: Comparison of latent and pixel losses

4.1.4 Latent space visualization

To justify the implementation of latent loss, we created a visualization prior to training our proposed model. For this purpose, we employ our pre-trained encoder-decoder(E and D) models configured in an autoencoder setup, along with the I_{gt} ground truth image and I_{masked}, which is a masked image containing a zeroed region within the mask. By inputting these images into the encoder, they are projected into the latent space. Unlike the ground truth image, the features of the masked image are anticipated to exhibit a damaged region, which could potentially be inferred directly from the features of the ground truth image. To validate this statement, we visualize this difference:

$$out = D[E(I_{gt}) - E(I_{masked})] \tag{4.1}$$

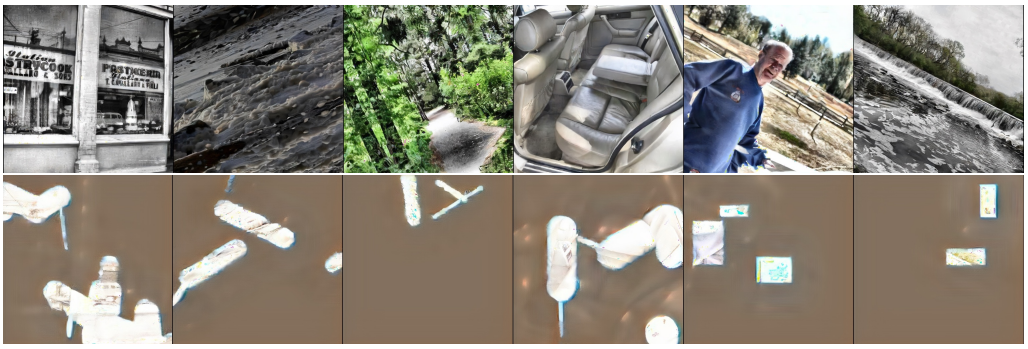


FIGURE 4.4: Visualization of latent space for inpainting. The top row displays the image, while the bottom row shows the decoded difference between the ground truth features and the masked features. [4.1]

Visualization [4.4] demonstrates that the latent features of ground truth possess sufficient information to learn, as the differences with the masked feature are distinctly decoded in the required pixel information. Additionally, the presence of nonzero values outside the mask suggests that the embedding suffered damage, not just near the mask but across the receptive field. This highlights the importance of integrating mask control into all stages of image encoding to prevent such feature degradation.

Chapter 5

Experiments

The primary goal of our experimental approach is to perform a quantitative assessment using the original LaMa model as part of our proposed framework. All training data and evaluation described in Chapter 3.

5.1 Mask Control

The initial stage of the experiments is to assess the impact of incorporating mask control into the standard LaMa framework. During this stage, the focus is on independently evaluating this modification against the traditional LaMa setup, using only the L1 loss in pixel space for simplicity.

Crucially, in every experiment, the encoder-decoder is initialized with weights from pre-training of the autoencoder. This is necessary to ensure uniform conditions for comparison. Additionally, the architecture retains all original layers, with the exception of modifications to the encoder's first convolution or the inclusion of a mask encoder, as detailed in the Mask column of table [5.1].

In baseline (1)[5.1], the original LaMa layers remain unchanged, with a 4-channel convolution for the input image with mask in the alpha channel, All components are unfrozen and only L1 loss is used. Conversely, our experiment (2)[5.1] maintains the same loss, layers, and initialization conditions, but additionally freezes the encoder-decoder and incorporates our Mask Encoder. For training time, it has increased to 7% due to the need to run an additional mask encoder, but all metrics have increased to 3% in FID, 1.3% in LPIPS, and 2.1% in SSIM, which show the effectiveness of the proposed mask control modification.

Method	Loss	Encoder-Decoder	Mask	Train Time↓	FID ↓	LPIPS ↓	SSIM ↑
(1) Baseline	L1 in pixel	Training	As alpha	6h	16.6	0.155	0.865
(2) Baseline with control	L1 in pixel	Freeze	Mask Encoder	6.4h +7%	16.0 -3%	0.153 -1.3%	0.884 -2.1%
(3) Latent	L1 in latent	Train only first conv	As alpha	**4h** -33%	18.3 +10%	0.163 +5%	0.878 +1%
(4) Latent with control	L1 in latent	Freeze	Mask Encoder	4.3h -28%	**12.4** -25%	**0.149** -4%	**0.888** +2%

TABLE 5.1: Comparison of different results. Percentage calculated compared to baseline(first row)

5.2 Latent Loss

The next training is implemented to compare the relative enhancement obtained by switching the loss function from pixel-based to latent-based. In row (3)[5.1], the setup involves the same 4-channel input convolution, the absence of mask control, and latent loss, with the encoder-decoder frozen. The results indicate a substantial reduction in training time by 33%, accompanied by a noticeable decrease of 10% in FID and 5% in LPIPS, while SSIM shows a minor improvement of 1%. The notable

reduction in training duration is attributed to the elimination of the need for the costly decoder component of the network and reduced computation in the latent space for L1 distnace, although this only affects the training phase, the inference time remains unchanged.

To visually illustrate the changes in these metrics, we present our sample in Figure [5.1]. Despite both outcomes being pure, due to the lack of perceptual and adversarial losses, which is indicated in FID and LPIPS, the outcome from latent training exhibits superior texture quality, as demonstrated by SSIM metrics. This highlights the advantages of latent loss in improving texture compared to pixel loss.

FIGURE 5.1: Comparative visualization of inpainting on a test image. The left image is trained with pixel loss, while the middle one is with latent loss and the right one is with latent and mask control. The middle image demonstrates improved textures of the result and right also improve visual quality.

One more final experiment is needed to evaluate the effectiveness of combined latent loss with mask control over baseline training. For (4)[5.1] we use mask control, frozen autoencoder, and latent loss for training. The mask encoder increases the time to 5% compared to training with latent loss only (and no mask encoder), but this time is still a significant improvement over baseline in 28% percent. Furthermore, alongside the reduction in time, there is a modest improvement in FID by 25%, a slight increase of 4% in LPIPS, and a steady improvement 2% in SSIM.

Besides the qualitative results, there is also a minor enhancement in the visual appearance of the image, yet the outcome remains unconvincing. This prompted our final experiment involving the integration of adversarial loss, anticipated to significantly boost the quality.

Moreover, this experiment revealed a noteworthy outcome: training within the latent space, employing both latent loss and mask control, enhances FID and LPIPS scores, which rely on an external feature extraction network. This discovery could be instrumental in future research focused on transferring perceptual or even adversarial losses to the latent space, thereby eliminating the requirement to decode features into pixel space, significantly improving training efficiency.

5.3 Full training

Prior experiments were conducted to assess the impact of proposed modifications in a smaller baseline training only with distance loss at the pixel or latent levels, insufficient to achieve results that can be compared. Therefore, in our final experiment, we implemented a complete training of the baseline's LaMa model, maintaining the same number of training steps and incorporating all losses. Building on (1)[5.1], we incorporated all necessary blocks to align the training with that described in the original LaMa paper, but retained the autoencoder initialization to ensure uniform model weight initialization and, thereby, a fair comparison. For comparison, we adjusted (4)[5.1] similarly by incorporating all losses from the original LaMa training, but on top of it adding the latent loss and mask control. In this configuration, we completely replicate the original LaMa training and compare it to the LaMa training that includes our integrated framework. **Note**, we reduced the number of training

steps as the original training required approximately one week on a GPU cluster, which was inaccessible to us. Consequently, we adjusted the batch size and the number of epochs to fit our available training resources while ensuring satisfactory performance. For more information, see the latent-inpainting[1] repo.

Method	Loss	Enc-Dec	Mask	Train Time↓	FID ↓	LPIPS ↓	SSIM ↑
(0) LaMa baseline*	Original	Training	As alpha	∼1week	0.63	0.035	-
(1) LaMa baseline	Original	Training	As alpha	40h	1.5	0.095	0.897
(2) LaMa w/ framework	Orig + Latent	Freeze	Mask Enc	41h +2%	1.0 -33%	0.087 -9%	0.902 +0.5%

TABLE 5.2: Comparison of final training. (0) row is full steps-epochs training from authors (no SSIM provided). (1) is our training trough original setup and code, but with less training time. And (2) is modification of (1) using our proposed framework. *-checkpoint taken from authors.

As observed in the table, there is a modest 2% increase in training time, while FID maintains a 33% high improvement over reduced training with one loss. Furthermore, LPIPS shows a tendency to improve up to 9%, and SSIM decreases marginally from 2% to 0.5%. As anticipated, visual evaluations [5.2] reveal significantly improved results compared to the smaller experiments discussed in the previous section. However, when comparing results directly, the differences are not substantial, although training incorporating latent loss does demonstrate minor but visually discernible enhancements.

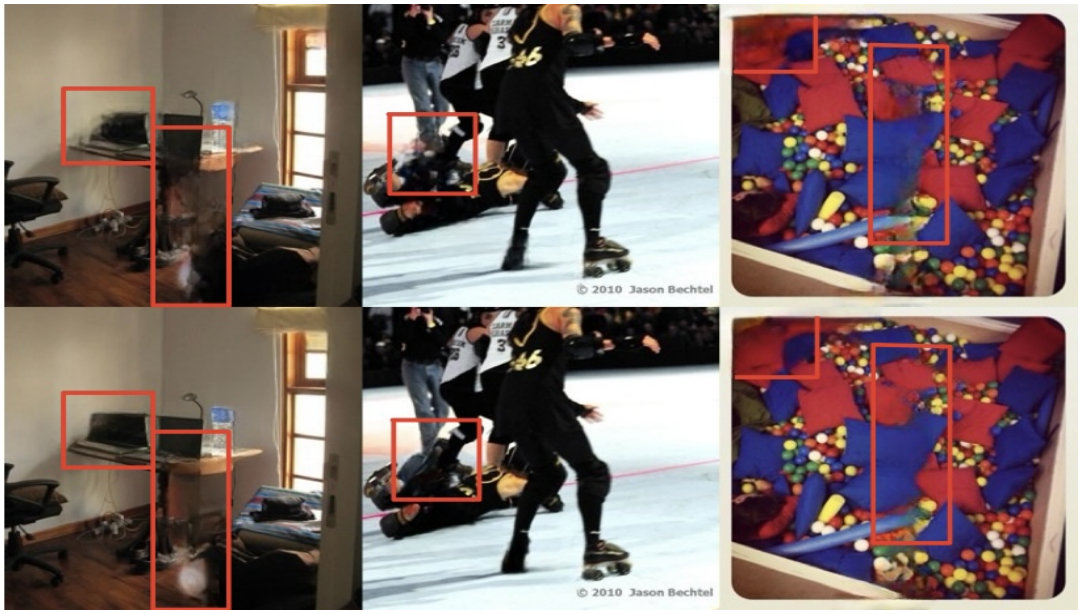


FIGURE 5.2: Comparison of LaMa baseline and LaMa with proposed framework

Furthermore, our experimental results indicate that a reduced coefficient of latent loss is essential relative to other pixel-based losses. Despite the losses having the same magnitudes, the coefficient for latent loss must be reduced by four orders of magnitude (to 0.001) to facilitate training convergence. This requirement may be driven by a significantly stronger gradient signal from the latent loss, as pixel-based losses are subjected to a decoder, which might diminish the gradient magnitudes.

[1] https://github.com/igor185/latent-inpainting

Chapter 6

Conclusion

We showcased the effectiveness of using latent loss and conditioning input in latent space by applying the proposed framework to the inpainting problem. In particular, the state-of-the-art LaMa method was modified to work in latent space by incorporating latent loss and conditioning mask input in intermediate encoding layers, which resulted in improved inpainting results. Mask control is responsible for an improved reconstruction of 1.6% and a reduction in perceptual error by 25% with a negligible increase in training time. The results imply that conditioning the intermediate encoding layers of the image is critical to preventing the latent space from being corrupted by the mask. This contrasts with the state-of-the-art, which only conditions the image in the image space. Furthermore, changing pixel loss to latent loss led to a substantial 33% reduction in training time and reduced the reconstruction error by 1%. This result implies that a latent loss can mitigate errors introduced by the decoder and should always be incorporated when the decoder is frozen, which is common when using off-the-shelf autoencoders.

Incorporating both components of our framework into LaMa resulted in a significant improvement in FID by 33%, in LPIPS by 9%, and in SSIM by 0.5% with a negligible increase in training time, which shows that the framework produces a higher fidelity latent space as desired.

Appendix A

Delta prediction

A potential enhancement for the framework could be the implementation of skip connections that bypass FFC blocks. This approach aims to forecast delta variations within the feature block, potentially improving the refinement of features or facilitating their reusability, such as sharing weights for FFC blocks, similar to the methods used in optical flow [Shen, Kerofsky, and Yogamani, 2023] estimation models such as RAFT [Teed and Deng, 2020]. The rationale behind this is to enable the model to iteratively predict minor enhancements to the feature, thereby avoiding the need to retain all previously known information.

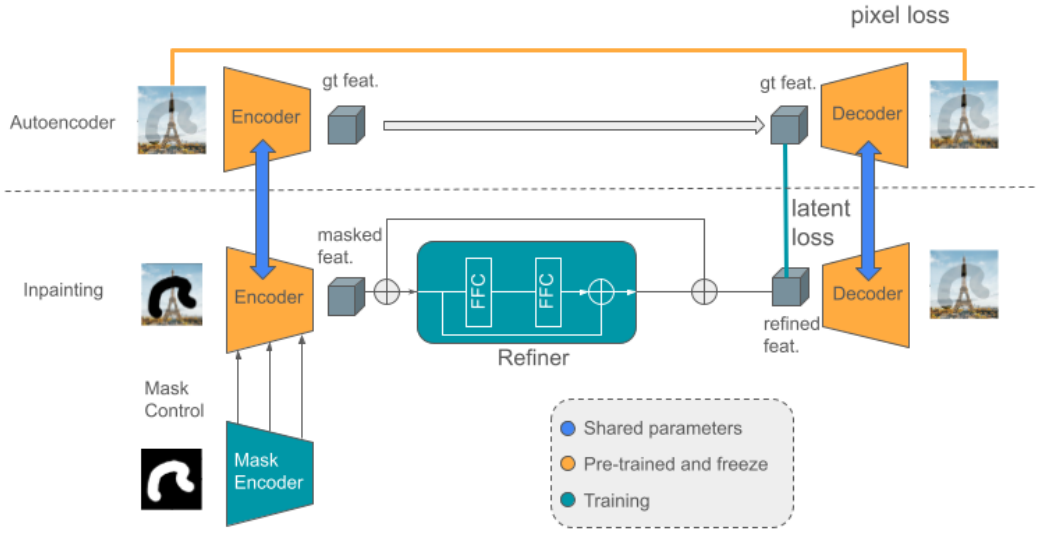


FIGURE A.1: Proposed model with added skip connection (delta) modification

Upon evaluating the method with the skip connections, it was observed that there was no improvement in the metrics, probably due to the inductive bias of the FFC blocks aimed at reconstructing the complete frequency spectrum of the images, but it could give some improvements under different models for refining features.

Method	FID ↓	LPIPS ↓	SSIM ↑
Baseline with pixel loss (full)	**16.6**	**0.1555**	0.865
Baseline with pixel loss (delta)	17.96	0.1643	**0.883**
Autoencoder with control with latent loss (original)	**12.4**	**0.149**	**0.888**
Autoencoder with control with latent loss (delta)	19.06	0.157	0.884

TABLE A.1: Comparison of 'delta' and 'original' setup

Appendix B

Code of Framework changes

The full code you can find in the GitHub repository of the latent-inpainting[1] repo

B.1 Mask Control

```
class DefaultInpaintingTrainingModule(BaseInpaintingTrainingModule):
    def __init__(...):
    ...
        self.zero_conv2 = nn.Conv2d(8, 8, kernel_size=1, stride=1,
                padding=0, bias=False)
        self.zero_conv3 = nn.Conv2d(16, 16, kernel_size=1, stride=1,
                padding=0, bias=False)
        self.zero_conv4 = nn.Conv2d(32, 32, kernel_size=1, stride=1,
                padding=0, bias=False)
        self.zero_conv5 = nn.Conv2d(64, 64, kernel_size=1, stride=1,
                padding=0, bias=False)

        self.zero_conv2.weight.data = torch.zeros_like(self.zero_conv2.weight)
        self.zero_conv3.weight.data = torch.zeros_like(self.zero_conv3.weight)
        self.zero_conv4.weight.data = torch.zeros_like(self.zero_conv4.weight)
        self.zero_conv5.weight.data = torch.zeros_like(self.zero_conv5.weight)

        self.mask_encoder = nn.Sequential(
                    nn.Conv2d(1, 8, kernel_size=1, stride=1, padding=0),
                    nn.BatchNorm2d(8),
                    nn.ReLU(),
                    nn.Conv2d(8, 16, kernel_size=3, stride=2, padding=1),
                    nn.BatchNorm2d(16),
                    nn.ReLU(),
                    nn.Conv2d(16, 32, kernel_size=3, stride=2, padding=1),
                    nn.BatchNorm2d(32),
                    nn.ReLU(),
                    nn.Conv2d(32, 64, kernel_size=3, stride=2, padding=1),
                    nn.BatchNorm2d(64),
                    nn.ReLU(),
        )
        ...

         def forward(self, batch, mode='train'):
```

[1] https://github.com/igor185/latent-inpainting

```
            ...

            ms = []
            m = mask.float()
            for l in self.mask_encoder:
                m = l(m)
                ms.append(m)


            for i, l in enumerate(encoder):
                masked_img = l(masked_img)
                if i == 1:
                    masked_img = self.zero_conv2(ms[2]) + masked_img
                if i == 2:
                    masked_img = self.zero_conv3(ms[5]) + masked_img
                if i == 3:
                    masked_img = self.zero_conv4(ms[8]) + masked_img
                if i == 4:
                    masked_img = self.zero_conv5(ms[11]) + masked_img
            masked_feat = masked_img

            ...
```

B.2 Latent Loss

```
class DefaultInpaintingTrainingModule(BaseInpaintingTrainingModule):
    def forward(self, batch, mode='train'):
        ...
        batch["refined_feat"] = refined_feat
        if use_latent_l2:
            batch['gt_feat'] = encoder(img)
        ...

    def generator_loss(self, batch):
        ...


        if 'gt_feat' in batch:
            l1_latent = F.l1_loss(torch.cat(batch['refined_feat'], dim=1),
            torch.cat(batch['gt_feat'], dim=1))
            total_loss = total_loss + coef_latent * l1_latent
        ...
```

Bibliography

Balestriero, Randall et al. (2023). "A cookbook of self-supervised learning". In: *arXiv preprint arXiv:2304.12210*.

Barnes, Connelly et al. (Aug. 2009). "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing". In: *ACM Transactions on Graphics (Proc. SIGGRAPH)* 28.3.

Brunet, Dominique (2012). "A study of the structural similarity image quality measure with applications to image processing". In.

Chi, Lu, Borui Jiang, and Yadong Mu (2020). "Fast Fourier Convolution". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., pp. 4479–4488. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/2fd5d41ec6cfab47e32164d5624269b1-Paper.pdf.

Cipolina-Kun, Lucia, Simone Caenazzo, and Gaston Mazzei (2022). "Comparison of CoModGans, LaMa and GLIDE for Art Inpainting Completing MC Escher's Print Gallery". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 716–724.

Dhariwal, Prafulla and Alexander Nichol (2021). "Diffusion models beat gans on image synthesis". In: *Advances in neural information processing systems* 34, pp. 8780–8794.

Goodfellow, Ian et al. (2014). "Generative adversarial nets". In: *Advances in neural information processing systems* 27.

Heusel, Martin et al. (2017). "Gans trained by a two time-scale update rule converge to a local nash equilibrium". In: *Advances in neural information processing systems* 30.

Jain, Jitesh et al. (2023). "Keys to better image inpainting: Structure and texture go hand in hand". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 208–217.

Johnson, Justin, Alexandre Alahi, and Li Fei-Fei (2016). "Perceptual losses for real-time style transfer and super-resolution". In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*. Springer, pp. 694–711.

Karras, Tero et al. (2020). "Analyzing and improving the image quality of stylegan". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8110–8119.

Kulshreshtha, Prakhar, Brian Pugh, and Salma Jiddi (2022). "Feature refinement to improve high resolution image inpainting". In: *arXiv preprint arXiv:2206.13644*.

LeCun, Yann, Yoshua Bengio, et al. (1995). "Convolutional networks for images, speech, and time series". In: *The handbook of brain theory and neural networks* 3361.10, p. 1995.

Pathak, Deepak et al. (2016). "Context encoders: Feature learning by inpainting". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2536–2544.

Rodríguez Pardo, Carlos et al. (Oct. 2019). "Automatic Extraction and Synthesis of Regular Repeatable Patterns". In: *Computers and Graphics* 83, pp. 33–41. DOI: 10.1016/j.cag.2019.06.010.

Rombach, Robin et al. (2021). "High-Resolution Image Synthesis with Latent Diffusion Models". In: *CoRR* abs/2112.10752. arXiv: 2112.10752. URL: https://arxiv.org/abs/2112.10752.

Shen, Shihao, Louis Kerofsky, and Senthil Yogamani (2023). "Optical flow for autonomous driving: Applications, challenges and improvements". In: *arXiv preprint arXiv:2301.04422*.

Shi, Xiaoyu et al. (2024). "Motion-I2V: Consistent and Controllable Image-to-Video Generation with Explicit Motion Modeling". In: *arXiv preprint arXiv:2401.15977*.

Suvorov, Roman et al. (2022). "Resolution-robust large mask inpainting with fourier convolutions". In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2149–2159.

Teed, Zachary and Jia Deng (2020). "Raft: Recurrent all-pairs field transforms for optical flow". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, pp. 402–419.

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky (2016). "Instance normalization: The missing ingredient for fast stylization". In: *arXiv preprint arXiv:1607.08022*.

Van Den Oord, Aaron, Oriol Vinyals, et al. (2017). "Neural discrete representation learning". In: *Advances in neural information processing systems* 30.

Zhang, Lingzhi et al. (2022). "Inpainting at modern camera resolution by guided patchmatch with auto-curation". In: *European Conference on Computer Vision*. Springer, pp. 51–67.

Zhang, Lvmin, Anyi Rao, and Maneesh Agrawala (2023). "Adding conditional control to text-to-image diffusion models". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847.

Zhang, Richard et al. (2018). "The unreasonable effectiveness of deep features as a perceptual metric". In: pp. 586–595.

Zhou, Bolei et al. (2016). "Places: An image database for deep scene understanding". In: *arXiv preprint arXiv:1610.02055*.