Spatially-Adaptive Multilayer GAN Inversion and Editing

CMU-RI-TR-22-26

Gaurav Parmar

April 1, 2022

The Robotics Institute

Carnegie Mellon University

Pittsburgh, Pennsylvania 15213

Thesis Committee:

Prof. Jun-Yan Zhu, Advisor

Prof. Shubham Tulsiani

Yufei Ye

Thesis proposal submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics

Abstract

Existing GAN inversion and editing methods are well suited for only a target images that contain aligned objects with a clean background, such as portraits and animal faces, but often struggle for more difficult categories with complex scene layouts and object occlusions, such as cars, animals, and outdoor images. We propose a new method to invert and edit such complex images in the latent space of GANs, such as StyleGAN2. Our key idea is to explore inversion with a collection of layers, spatially adapting the inversion process to the difficulty of the image. We learn to predict the "invertibility" of different image segments and project each segment into a latent layer. Easier regions can be inverted into an earlier layer in the generator's latent space, while more challenging regions can be inverted into a later feature space. Experiments show that our method obtains better inversion results compared to the recent approaches on complex categories, while maintaining downstream editability.

i

Acknowledgements

I would like to thank my advisor, Prof. Jun-Yan Zhu, for the support and advice throughout the two years of my study. His guidance and frequent feedback through the project meetings and the one-on-one meetings were crucial in the development and the success of the project.

Next, I would also like to thank my collaborators Krishna Kumar Singh, Richard Zhang, Jingwan (Cynthia) Lu, and Yijun Li for the fruitful and regular project discussions.

Finally, I would like to thank the other members in the lab - Sheng-Yu Wang, Nupur Kumari, Kangle Deng, George Cazenavette, Ruihan Gao, and Chonghyuk (Andrew) Song for the conversations about the broader research field and specific discussion in the group meetings.

Funding

Contents

List of Figures

x

List of Tables

Chapter 1

Introduction

The recent advances of Generative Adversarial Networks [19], such as Pro-GAN [29], the StyleGAN family of models [31–33], and BigGAN [12], have revived the interest in GAN inversion and editing [13, 63]. In GAN editing pipelines, one first projects an image into the latent space of a pre-trained GAN, by minimizing the distance between the generated image and an input image. We can subsequently change the latent code according to a user edit, and synthesize the output accordingly. The latent code can then be changed, in order to satisfy a user edit. The final output image is synthesized with the updated latent code. Several recent methods have achieved impressive editing results for real images [2, 8, 40, 62] using scribbles, text, attribute, and object class conditioning. However, existing methods work well for human portraits and animal faces but are less applicable to more complex classes such as cars, horses, and cats. Compared to faces, these objects have more diverse visual appearance and cluttered backgrounds. In addition, they tend to be less aligned and more often occluded, all of which make inversion more challenging.

1.1 Contributions

In this work, our goal is address this limitation of existing methods and invert challenging images better. We build our method upon two key observations.

(1) *Spatially-adaptive invertibility*: first, the inversion difficulty varies across different regions within an image. Even if the entire image cannot be inverted in the early latent spaces (e.g., W and W^+ space of StyleGAN2 [33]), if we break the image into multiple segments, the easier regions can still be inverted in these latent spaces with high fidelity. For example, in Figure 1.1, while the car and sky regions are well-modeled by the LSUN CAR generator, shrubs and fences are not, as they appear less frequently in the dataset. Besides, both regions are occluded by the foreground car.

(2) *The trade-off between invertibility and editability*: as noted by prior work [51, 65], the choice of layer can determine how precisely an image can be reconstructed and the range of downstream edits that can be performed. Early latent layers of a generative model (W, W^+) are often unable to reconstruct challenging images, but allow meaningful global and local editing. In contrast, inversion using later intermediate layers reconstructs the image more precisely at the cost of reduced editing capability. As invertibility increases in later layers, the editability decreases. The first two rows in Figure 1.1 show these trade-offs concretely for a real car image.

Considering the spatially-varying difficulty and the trade-off between editability and invertibility, we perform spatially-adaptive multilayer (SAM) inversion by choosing different features or latent spaces to use when inverting each image region. We train a prediction network to infer an invertibility map for an input image indicating the latent spaces to be used per segment
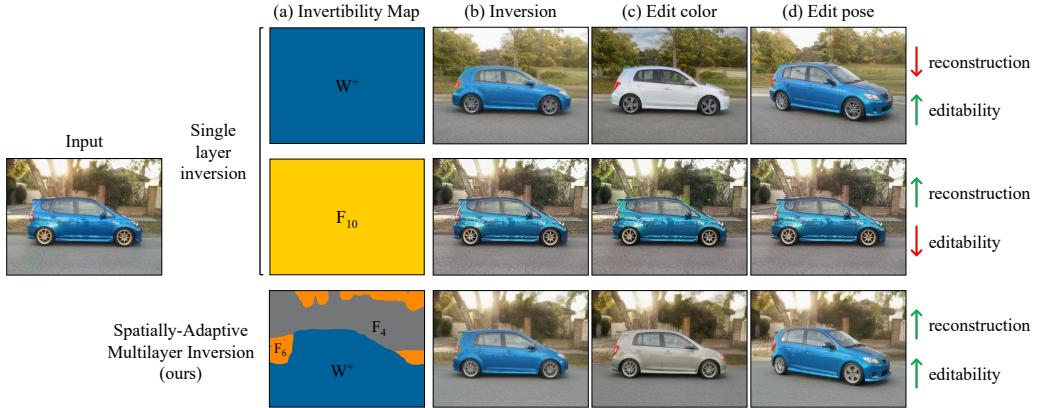
Figure 1.1: **Inverting and editing an image with spatially adaptive multilayer latent codes.** Choosing a single latent layer for GAN inversion leads to a dilemma between obtaining a faithful reconstruction of the input image and being able to perform downstream edits (1st and 2nd row). In contrast, our proposed method automatically selects the latent space tailored for each region to balance the reconstruction quality and editability (3rd row). Given an input image, our model predicts an invertibility map (a), which contains the layer index used for each region. This allows us to precisely reconstruct the input image (b) while preserving editability (c,d).

as shown in the second column of Figure 1.1. Our approach enables generating images very close to the target input images while maintaining the downstream editing ability.

1.2 Results

In order to validate our proposed method, we conduct experiments on multiple domains such as FACES, CARS, HORSES, and CATS. The results show that our method can maintain editability while reconstructing even challenging

images more precisely. We measure reconstruction with standard metrics such as PSNR and LPIPS. Concretely, we show an improvement of about 2 PSNR and 0.2 LPIPS across all domains.

Next, we evaluate the image quality and the downstream editing ability by conducting user preference studies. The results show that the images generated by our method is preferred over competing method by the users. Finally, we demonstrate the generality of our idea on different generator architectures (StyleGAN2 [33], BigGAN-deep [12]), and different paradigms (optimization based or encoder based).

1.3   Overview

In Chapter 2, we begin with a discussion of prior works that attempt the task of GAN Inversion, defining different latent spaces in a pretrained GAN, and finding edit directions. After reviewing the related works, Chapter 3 describes the different components of proposed inversion algorithm, including the selection of optimal latent space for each image region, fusing of different latent spaces, the objective function optimized during inversion, and the steps used for subsequent downstream editing. Chapter 4 compares the proposed inversion method with other recent approaches, and shows results of ablation experiments motivating the importance of each component in the method.

Chapter 2

Related Works

2.1 GAN Inversion and Editing

Since the introduction of GANs [19], several methods have proposed projecting an input image into the latent space of GANs for various editing and synthesis applications [14, 35, 42, 63]. This idea of using GANs as a strong image prior was later used in image inpainting, deblurring, compositing, denoising, colorization, semantic image editing, and data augmentation [7, 8, 15, 16, 20, 54, 59]. See a recent survey [57] for more details. The enormous progress of large-scale GANs [12, 28–33, 61] allows us to adopt GAN inversion for high-resolution images [1,2]. One popular application is portrait editing [3, 4, 37, 50].

Current methods can be categorized into three groups: optimization-based, encoder-based, and hybrid methods. The optimization-based methods [1,2,33,36,63] aim to minimize the difference between the optimization output and the input image. Despite achieving fairly accurate results, the slow process requires many iterations and may get stuck in local optimum.

To accelerate the process, several works [14,35,42,43,51,53,63] learn an encoder to predict the latent code via a single feed-forward pass. However, the learned encoder is sometimes limited in reconstruction quality compared to the optimization-based scheme. Naturally, hybrid approaches that combine the best of both schemes emerge [5,8,10,24,53,63], but the trade-off between quality and speed still persists. Existing inversion and editing methods work well for simple classes like faces but give sub-optimal results for more challenging images. We propose a solution method that is reliable even for such images.

2.2 Choosing the Latent Space

Several previous methods [1,2] focus on inverting the input image into the latent space of StyleGAN models [32,33] that use AdaIN layers [23] to control the "style" of an image. In addition to exploring different projection schemes, they demonstrate that the choice of latent space is a key factor due to the unique style-based design of the StyleGAN. Instead of projecting an image into the latent space [14,63], recent works propose projecting an image into style parameter space [1,2,55] and convolutional feature space [64]. As noted by recent work [51,65], there exists a trade-off between the invertibility and editability, and no layer can maximize both criteria at the same time.

To handle complex images, recent papers propose using multiple codes of the same layer [20,26,49], splitting image into segments [18], using consecutive images [58], explicitly handling misaligned objects [6,24,27], modifying the generator architecture for better editing ability [34,39], adopting a class-conditional GAN [24,38,49], and fine-tuning the generator to an input

image [8, 38, 44].

Different from the above methods that operate on a single layer, we take into account the inversion difficulty across different input image segments and perform the inversion separately for each segment by using multiple latent spaces. We show that our method outperforms a concurrent generator fine-tuning method [44] in our experiments.

2.3 Finding Edit Directions

After inversion, we can edit the inverted code by traversing semantically meaningful directions computed using supervised [9, 25, 47] or unsupervised approaches [17,21,41,48,52]. Most of these methods compute these directions offline [9,25,48] and provide them as pre-canned options for users. Other works calculate the editing directions during inference time to support more flexible editing interfaces with scribbles [63] and text inputs [40]. We show that our method can work well with different types of directions.

Chapter 3

Approach

We aim to invert images using a pretrained GAN while maintaining editability. We begin by learning to predict an invertibility map that indicates which latent spaces should be used for each image region. Next, we fuse features from different latent spaces to generate an image that matches our input and can be edited in the latent space.

3.1 Predicting Invertibility

As discussed previously, different latent spaces have different inversion capabilities. We learn a network to predict what parts of the image are invertible using any given latent space. Here we use "invertibility" to indicate how closely our generated result can match the input image. In Figure 3.1 (left), we show how we learn invertibility predictor for different latent spaces. We collect a dataset of image pairs that consists of the input image $x \in R^{H \times W \times 3}$ and its reconstruction $\hat{x}_l \in R^{H \times W \times 3}$ into the lth latent space, following the optimization-based inversion suggested by Karras et al. [33]. We consider 5
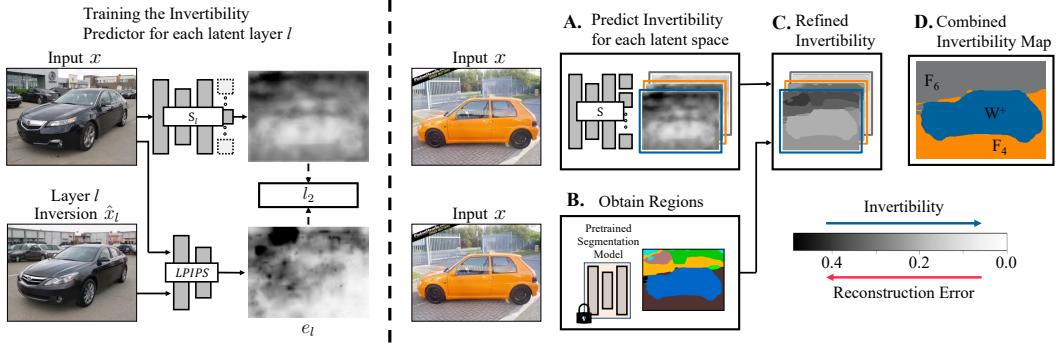
Figure 3.1: **Training the Invertibility Segmenter.** On the left we show how each of the invertibility predictior S_l are trained. We invert all images in the training set using one of the five candidate latent spaces and use the LPIPS [60] spatial error map e_l as supervision. Next (right) we show how the trained invertibility models are used to generate the final inversion latent map. We first predict how difficult each region of the image is to invert for every latent layer using our aforementioned invertibility network. Subsequently we refine the predicted map using a pre-trained semantic segmentation network and combine them using the user-specified threshold τ. This combined invertibility map shown on the right is used to determine the latent layer to be used for inverting each segment in the image.

different latent spaces $\Phi = \{W^+, F_4, F_6, F_8, F_{10}\}$, where the index of F corresponds to the feature layer index of the StyleGAN2 generator and W^+ is the concatenation of different vectors from W space, in which W space is the output space of the MLP network of StyleGAN2. We choose W^+ instead of W, as it provides better inversion results and more fine-grained and disentangled control when performing the downstream edits. Next, we compute the reconstruction loss as follows

$$e_l = \mathcal{L}_{\text{LPIPS}}(x, \hat{x}_l), \tag{3.1}$$

9

where $e_l \in R^{H \times W}$ is the LPIPS spatial error map [60] between x and their inversions $\hat{x}_l$ for each latent space.

The parts that are easy to invert have smaller spatial errors, whereas difficult regions induce larger errors. We subsequently train a network to predict the invertibility for each latent space, regressing to the LPIPS spatial error map via an ℓ_2 loss. The training loss can be formulated as follows:

$$S_l = \arg \min_{S_l} \ell_2 \big(S_l(x), e_l \big). \tag{3.2}$$

Once trained, this network predicts the invertibility for any input image, at any layer, in a feed-forward fashion. Our trained invertibility network shares a common backbone with different prediction heads corresponding to each individual latent space. Next, we leverage our predicted invertibility to assign different regions to different latent spaces for the inversion. However, our prediction can be noisy and may not be consistent within the same semantic region. This could potentially result in inconsistent inversions and edits, as different parts of the same region can be assigned to different latent codes. We refine our prediction using a pretrained segmentation model. For every segment, we compute the average predicted invertibility in the region and use the value for the entire segment. As shown in Figure 3.1 (right), such a refining step helps us align the invertibility map with natural object boundaries in the image.

3.2 Adaptive Latent Space Selection

We observe that latent spaces have an inherent trade-off between reconstructing the input image and utility for downstream image editing tasks,
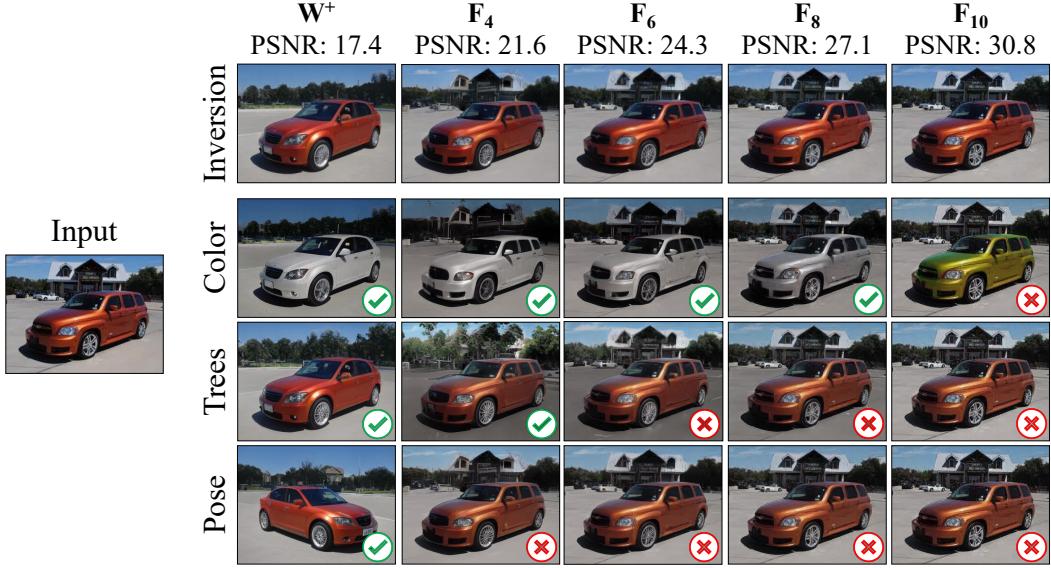
Figure 3.2: **Trade-offs between invertibility and editibility.** We show inversion and editing when the input is inverted using different *single* latent layers. As we go down in feature space reconstruction improves but editing capabilities decreases. The improvement in reconstruction is shown visually for a single image and quantitatively with PSNR using 1000 images. Whether the edit was applied successfully is indicated by ⊘ and ⊗.

as also noted by recent work [51,65]. For example, choosing the latent space to be W^+ would result in an inverted latent vector that is amenable for editing, but sub-optimal for obtaining a faithful reconstruction for difficult input images. On the other hand, choosing activation block F_{10} (close to the generated pixel space) would have great reconstruction, but limited editing ability. In Figure 3.2, we show this trade-off for different choices of latent spaces explicitly. We invert the input image using a single latent layer, and observe that the reconstruction quality improves monotonically as we use layers increasingly closer to the output pixels.

Committing to a *single* latent layer for the whole image forces us to a single operating point on the trade-off between editability and reconstruction, across the whole image. Instead, we aim to *adapt* the latent layer selection, depending on the image content in a region. To do this, for each image segment, we choose the earliest latent layer, such that the reconstruction still meets some minimum criteria.

More concretely, for each segment, we choose the most editable latent space from Φ (W^+ being most editable and F_{10} being least), with predicted invertibility above threshold τ for that segment. We choose this threshold value empirically such that the inversion is perceptually close to the input image, without severely sacrificing editability. In Figure 3.3, we show our final inversion map, with different latent spaces assigned to different segments in the input image. The simple car region gets assigned to the W^+ space, whereas the difficult to generate background regions, which typically cannot be generated by the native latent space, gets assigned to the later F_4 and F_6 latent spaces.

3.3 Training Objective

We implement our multilayer inversion in two settings: 1) optimization-based and 2) encoder-based. In the optimization-based approach, we directly optimize the latent space ϕ for each image. For the encoder-based approach, we train a separate encoder for each latent space. Our encoder takes in input image along with invertibility map as input to predict the latent space. The encoder-based approaches are faster than optimization based approach but are typically achieve worse reconstruction. The goal of both paradigms is to find the latent code that reconstructs the input accurately,
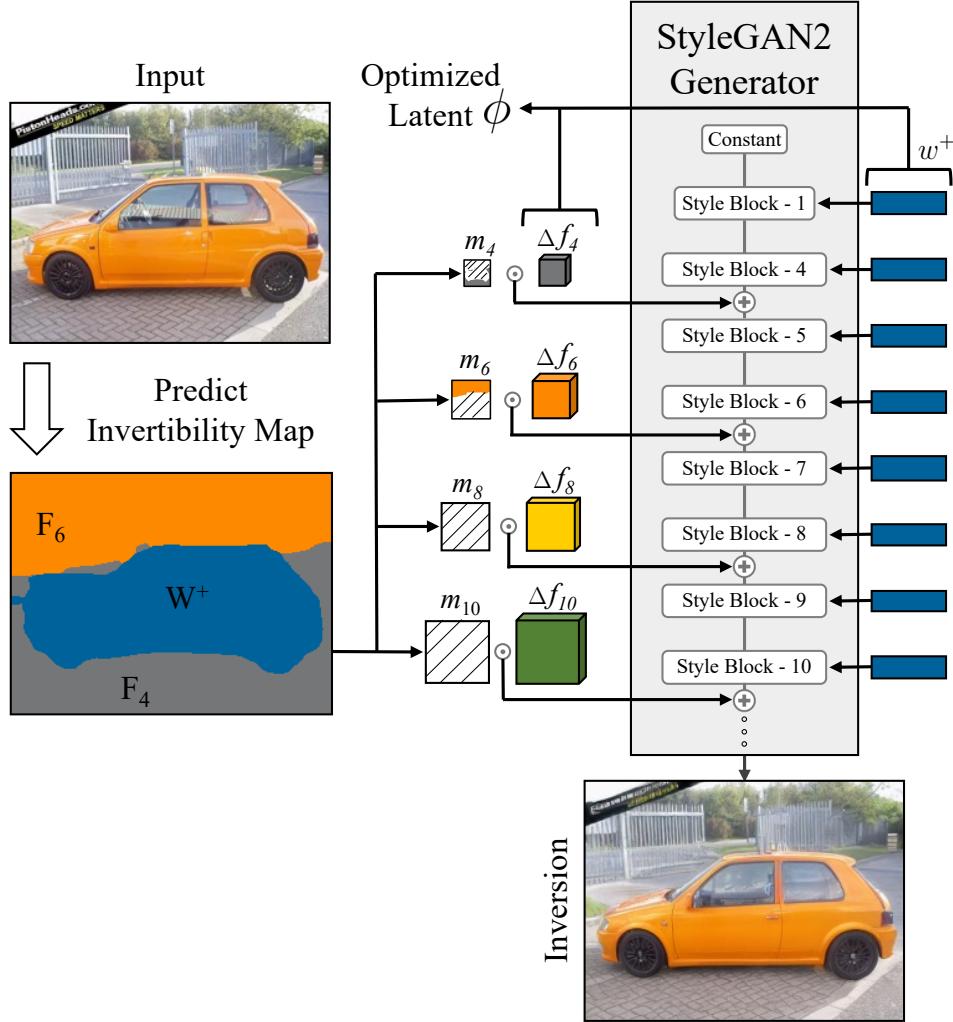
Figure 3.3: **Image formation using spatially adaptive latent codes.** We show how the predicted invertibility map is used in conjunction with multiple latent codes to generate the final image. $w^+ \in W^+$ directly modulates the StyleBocks of the pretrained StyleGAN2 model. For intermediate feature space F_i, we predict the change in layer's feature value Δf_i and add it to the feature block after masking with the corresponding binary mask m_i.

while maintaining editability.

Image formation model. In Figure 3.3, we show how the latent codes are combined to generate the final image. Our predicted $w^+ \in W^+$ is directly used to modulate the layers of pre-trained StyleGAN2. For feature spaces $F \in \{F_4, F_6, F_8, F_{10}\}$, we predict the change in values Δf for the regions that are to be inverted in that layer. We predict the change in layer's feature, rather than directly predicting the feature itself, as propagating the features from earlier layers provides a meaningful initialization to adjust from.

The output feature value is a combination of both w^+ and Δf masked by a binary mask indicating which region should be inverted in that layer. For example, to produce the feature $f_4 \in F_4$, we have:

$$f_4 = g_{0 \to 4}(c, w^+) + m_4 \odot \Delta f_4, \tag{3.3}$$

where $g_{i \to j}$ denotes the module from the i-th to the j-th layers in the convolutional layers of the StyleGAN2, c is the input constant tensor used in StyleGAN2, m_4 is the refined, predicted invertibility mask bilinearly downsampled to corresponding tensor size, and $\odot$ denotes the Hadamard product. Note that $g_{i \to j}$ is modulated by the corresponding part of the extended latent code w^+. Similarly, we can calculate all the features and the final output image as follows:

$$f_6 = g_{4 \to 6}(f_4, w^+) + m_6 \odot \Delta f_6$$
$$f_8 = g_{6 \to 8}(f_6, w^+) + m_8 \odot \Delta f_8$$
$$f_{10} = g_{8 \to 10}(f_8, w^+) + m_{10} \odot \Delta f_{10}$$
$$\hat{x} = g_{10 \to 16}(f_{10}, w^+). \tag{3.4}$$

Next, we present our objective functions to optimize the latent code $\phi = \{w^+, \Delta f_4, \Delta f_6, \Delta f_8, \Delta f_{10}\}$. We reconstruct the input image while regularizing the latent codes, in order to be meaningful for downstream editing tasks.

Reconstruction losses. We use the $\mathcal{L}_2$ distance between the inverted image $\hat{x}$ and the input image x along with LPIPS difference as our reconstruction losses.

$$\mathcal{L}_{\text{rec}} = \ell_2(x, \hat{x}) + \lambda_{\text{LPIPS}} \mathcal{L}_{\text{LPIPS}}(x, \hat{x}), \qquad (3.5)$$

where λ_{LPIPS} is the weight term.

W**-space regularization.** As noted in [51,56], inverting an image with just reconstruction losses results in latent codes that are not useful for editing. For our inversion methods, we use different latent regularization losses for different latent spaces. For w^+, we use the following:

$$\mathcal{L}_W = \sum_n^N \left[(\hat{w}_n - \mu)^T \Sigma (\hat{w}_n - \mu) + ||w_n^+ - w_0^+||^2 \right], \qquad (3.6)$$

where w_n^+ is the nth component of the w^+ vector, $\hat{w}_n = \texttt{LeakyReLU}(w_n^+, 5.0)$, μ and Σ are the empirical mean and covariance matrix of randomly sampled and converted $\hat{w}$ vectors respectively. The first term applies a Mutlivariate Gaussian prior [56], and the second term minimizes the variation between the individual style codes and the first style code.

F**-space regularization.** For the feature space, we enforce our predicted change Δf to be small, so that our final feature value does not deviate much from the original value.

$$\mathcal{L}_F = \sum_{\Delta f \in \phi \backslash w^+} ||\Delta f||^2 \qquad (3.7)$$

15

Final objective. Our full objective is written as follows:

$$\arg \min_{\phi} \mathcal{L}_{\text{rec}} + \lambda_W \mathcal{L}_W + \lambda_F \mathcal{L}_F, \tag{3.8}$$

where λ_W and λ_F control the weights for each term.

3.4   Image Editing

After obtaining the inverted latent codes, we edit the images by applying the edit direction vector to the inverted w^+ latent vector. We use GANSpace [21] and StyleCLIP [40] for finding an editing direction δw^+ in the W^+ latent space. Segments inverted in W^+ space get modulated by the entire code $w^+ + \delta w^+$, whereas segments inverted in intermediate feature spaces $\{F_4, F_6, F_8, F_{10}\}$ get modulated by $w^+ + \delta w^+$ only for the layers which come after that feature space layer. For example, segments inverted in F_{10} space get modulated by w^+ for the layers until the 10th layer, and $w^+ + \delta w^+$ for the layers afterward. This is necessary, as our inverted feature would not be compatible with $w^+ + \delta w^+$.

Chapter 4

Experiments

In this chapter we perform detailed quantitative and qualitative analysis to show effectiveness of our inversion method across different datasets.

4.1   Datasets

We test our method on pretrained StyleGAN2 and BigGAN-deep generators trained on a variety of different challenging domains and follow the commonly used protocol for the different domains [5,43,44]. For all experiments we use the official released StyleGAN2 [33] trained on LSUN Cars, LSUN Horses, LSUN Cats, and FFHQ [32] datasets, and the official released BigGAN-deep [12] trained on ImageNet [45]. We use a subset of 10,000 images from the dataset for training our invertibility prediction network S and 1000 images for the evaluation. More concrete details about the dataset splits are mentioned below.

Cars.   For the car domain, we train our invertibility networks using the train split of the Stanford cars dataset and evaluate our method using 1,000 im-

ages of size 512×384 randomly selected from the LSUN Cars dataset. The StyleGAN2 generator used is trained on the full LSUN Cars dataset. For the encoder inversion regime, we train our encoders on the full training split of Starford cars dataset.

Horses. For the horses domain, we use 10,000 images sampled from the LSUN Horses dataset for training our invertibility networks and a different set of 1000 images from the same dataset for evaluating. The StyleGAN2 generator used is trained on the full LSUN Horses dataset. For our encoder training, we use the same set of 10,000 images from the LSUN Horses dataset used for training the invertibility networks.

Cats. Similarly, we use 10,000 images from the LSUN Cats dataset for training our invertibility networks and a different set of 1000 images for the evaluation. The StyleGAN2 generator used is trained on the full LSUN Cats dataset. For our encoder training, we use the same set of 10,000 images from the LSUN Cats dataset used for training the invertibility networks.

Faces. We use 10,000 images from the FFHQ dataset for training our invertibility networks and a 1000 images from the CelebA-HQ dataset for the evaluation. The StyleGAN2 generator used is trained on the full FFHQ dataset.

4.2   Evaluation

We evaluate the performance of various inversion methods on two tasks - reconstruction and editibility. The reconstruction between the inverted image and the input image is measured using PSNR and LPIPS [60]. Note

that different prior inversion methods use different LPIPS backbones. We use LPIPS-VGG for all of our experiments and comparisons. As pointed out by [51], measuring the editing ability of the latent codes is difficult and image quality metrics such as IS [46], FID [22] and KID [11] do not correlate with the user preference. Therefore we show qualitative comparisons and conduct user preference studies to evaluate the quality of inverted and edited images.

4.3 Reconstruction Comparison

We first compare our inversion method to other state-of-the-art GAN inversions methods in the optimization-based regime. *StyleGAN2 Inversion* and *StyleGAN2 Inversion using W^+* invert image in W and W^+ latent space respectively. [56] applies multi-variant Gaussian prior constraint while doing the inversion. We also compare against *Hybrid W^+ Inversion* that uses a pre-trained *e4e* encoder [51] for initialization. Recently proposed pivotal tuning inversion (*PTI*) [44] additionally finetunes the weights of pre-trained Style-GAN2 after inverting the image in the W space. Table 4.1 shows that our method achieves better reconstruction across all the metrics compared to baselines. Our approach is able to invert difficult regions using intermediate layers feature space, whereas baselines struggle to invert by just relying on single W and W^+ space. *PTI* has the ability to change the StyleGAN2 weights to invert the image, but it uses heavy locality regularization to discourage the deviation from original weights, which limits its inversion capability. Also, for simpler image parts, we get better inversion as our W^+ latent code just focuses on parts that it can invert. In contrast, other approaches try to invert both easy and difficult parts using the same code, resulting in sub-

Method	Cars		Horses		Cats		Faces	
	LPIPS (↓)	PSNR (↑)	LPIPS (↓)	PSNR (↑)	LPIPS (↓)	PSNR (↑)	LPIPS (↓)	PSNR (↑)
StyleGAN2 [33] (W)	0.34	14.44	0.45	13.46	0.44	14.47	0.28	18.32
StyleGAN2 [33] (W^+)	0.24	17.29	0.34	15.74	0.35	17.11	0.20	22.10
Gaussian Prior [56]	0.45	15.92	0.42	17.19	0.49	17.01	0.15	25.18
Hybrid e4e [51]	0.36	17.05	0.42	16.68	0.42	17.91	0.15	25.13
PTI [44]	0.38	19.39	0.43	18.73	0.41	20.45	0.26	22.36
SAM - optimization (ours)	**0.16**	**22.81**	**0.23**	**21.07**	**0.22**	**22.91**	**0.13**	**26.89**
e4e [51]	0.47	14.57	0.55	13.98	0.56	14.68	0.34	19.39
ReStyle (pSp) [5]	0.43	16.44	0.45	16.53	0.48	17.58	**0.29**	**21.47**
ReStyle (e4e) [5]	0.45	15.61	0.52	14.50	0.53	15.64	0.34	19.72
SAM - encoder (ours)	**0.28**	**19.21**	**0.34**	**18.61**	**0.37**	**18.59**	0.29	21.10

Table 4.1: **Reconstruction comparison to prior methods.** We use PSNR and the LPIPS-VGG for the evaluating the reconstruction using 1000 images. For the challenging categories, we achieve a better reconstruction than all baseline approaches in both the optimization based and encoder based paradigms. The faces images are simpler and contain fewer challenging regions. Subsequently, our method performs slightly better than prior methods when inverting with optimization and similar to the best performing ReStyle (pSp) with encoders.

optimal inversion even for the easier part. We perform similar comparisons of encoder based methods and show that an encoder trained using our proposed method outperforms the encoder baselines [5,51] on challenging images. On faces, our encoder obtains a similar reconstruction with just a single forward pass as the best performing baseline *ReStyle (pSp)*, which requires five forward passes. We also compare the runtime of optimization-based and encoder-based inversion methods using 1000 Car images in Figure 4.1. In both paradigms, our method obtains a better reconstruction in a shorter amount of time.
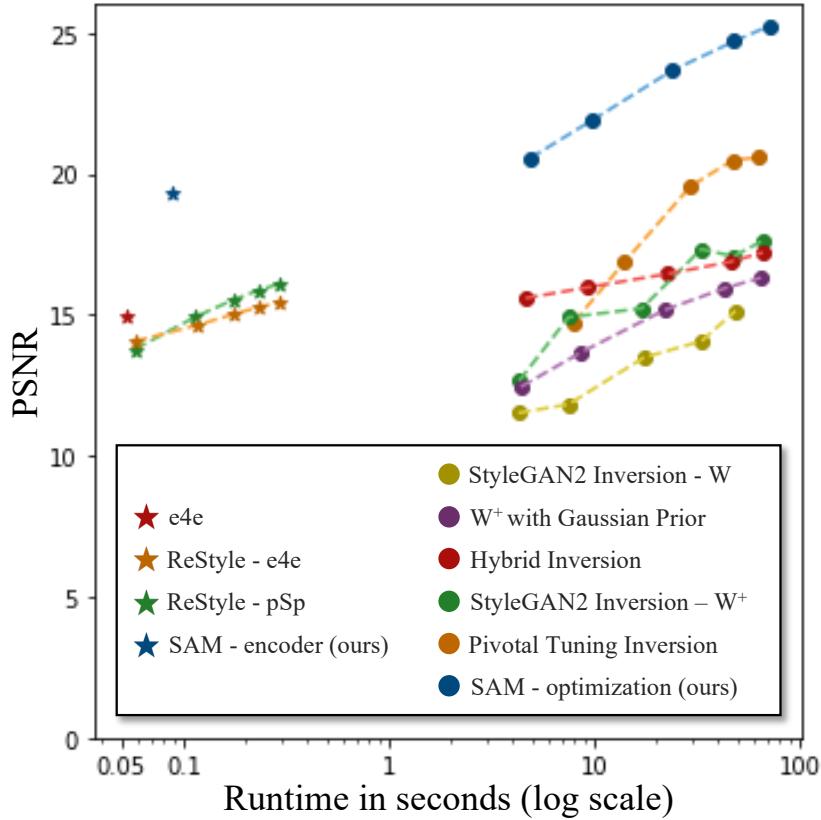
Figure 4.1: **Reconstruction at different runtimes.** We compare the reconstruction of different GAN inversion methods in the optimization and encoder regimes using 1000 car images. Each of the method uses a single NVIDIA RTX 3090 GPU. Our proposed method achieves a closer reconstruction to the input in a shorter amount of time for both the optimization and encoder paradigms.

4.4 Qualitative Results

Next, we show our ability to edit the reconstructed complex images in Figure 4.2. In the third column, we show our ability to reconstruct difficult regions using more capable latent layers F_4 and F_6, whereas the easy-to-

generate regions use the more editable W^+. This separation allows us to perform challenging edits while faithfully reconstructing the target image. Figure 4.4 shows inversion and editing results for a class-conditioned Big-GAN model.

In Figure 4.3, we observe that we get much closer inversion and realistic edits than baselines approaches. In some cases such as the first image, we can preserve even fine-grained details like the type of light and car wheels during editing stage. *StyleGAN2 inversion using W^+* generates realistic looking images but does not matches the input images well whereas *PTI* generates images that are closer but lack realism, especially after editing. We hypothesize that this is due to the incompatibility between the finetuned weights and the edit directions learned before finetuning.

4.5 User Study

We additionally conduct a user preference study to evaluate the realism of inverted and edited images. Table 4.2 compares our method to three closest baselines methods (*PTI* [44], *StyleGAN2 Inversion using W^+*, and *Hybrid W^+ Inversion*) using 500 different target images from each category. Every pair is evaluated by 3 randomized and different users, resulting in 1500 comparisons per baseline per category. The results show that users prefer our results over the baselines for all challenging image categories.

4.6 Ablation Studies

In this section we ablate the different components of our method and show their importance. More concretely we study effects of latent space regular-

Method	Inversion			Editing		
	Cars	**Horses**	**Cats**	**Cars**	**Horses**	**Cats**
PTI [44]	7.0%	11.6%	11.6%	18.4%	16.4%	38.0%
SAM (ours)	**93.0%**	**88.4%**	**88.4%**	**81.6%**	**83.6%**	**62.0%**
SG2-W+	28.0%	24.7%	20.5%	28.8%	35.7%	35.0%
SAM (ours)	**72.0%**	**75.3%**	**79.5%**	**71.2%**	**64.3%**	**65.0%**
e4e hybrid	23.2%	21.8%	22.4%	36.4%	38.3%	44.6%
SAM (ours)	**76.8%**	**78.2%**	**77.6%**	**62.6%**	**61.7%**	**55.4%**

Table 4.2: **User preference comparison with prior methods.** We invert and edit 500 from each of the image categories and ask 3 different users (1500 pairs per comparison). Results show that images generated by our method are preferred by the users. The spread in the values computed with boot-strapping is $< 2.5\%$.

ization, refining of the invertibility map, and using the object class labels as the invertibility map.

Regularizing the latent spaces. The Equation 3.3 regularizes the feature space to ensure our predicted feature values do not deviate too much from original feature space distribution. This is necessary to ensure that our edited direction is compatible with our predicted feature values. In Figure 4.5, we show two inversion and editing results on the same input image. In the top row, the inversion is performed without the regularization $\mathcal{L}_F$ and the bottom row shows the results when the feature space is regularized. The images in the right column show the edited images when an edit direction

corresponding to adding trees is applied with the *same magnitude* to both the inverted latent codes. Both the cases achieve a good inversion, the lack of feature space regularization results in an inversion that is not editable.

Refining the invertibility map. Next, we show the importance of the refinement step introduced in Section 3.1 of the main paper. This step ensures that a semantic class gets assigned to a single latent space. In Figure 4.6, we show that edited image become inharmonious without such a refinement step. In particular, the top row in Figure 4.6 shows that different parts of the car get inverted using different latent spaces without the refinement step. As a result, we get incoherent size change edits for the car as the edits will be applied to different layers for different car regions depending on which latent space it uses.

Class segmentation as invertibility map. The well-performing GAN models are typically trained using an object specific dataset. Here, we consider an inversion approach that uses per-pixel class labels instead of our predicted invertibility map. For a GAN trained on the car images, we invert the car segment with W^+ and the rest of the image background is inverted using F_6.
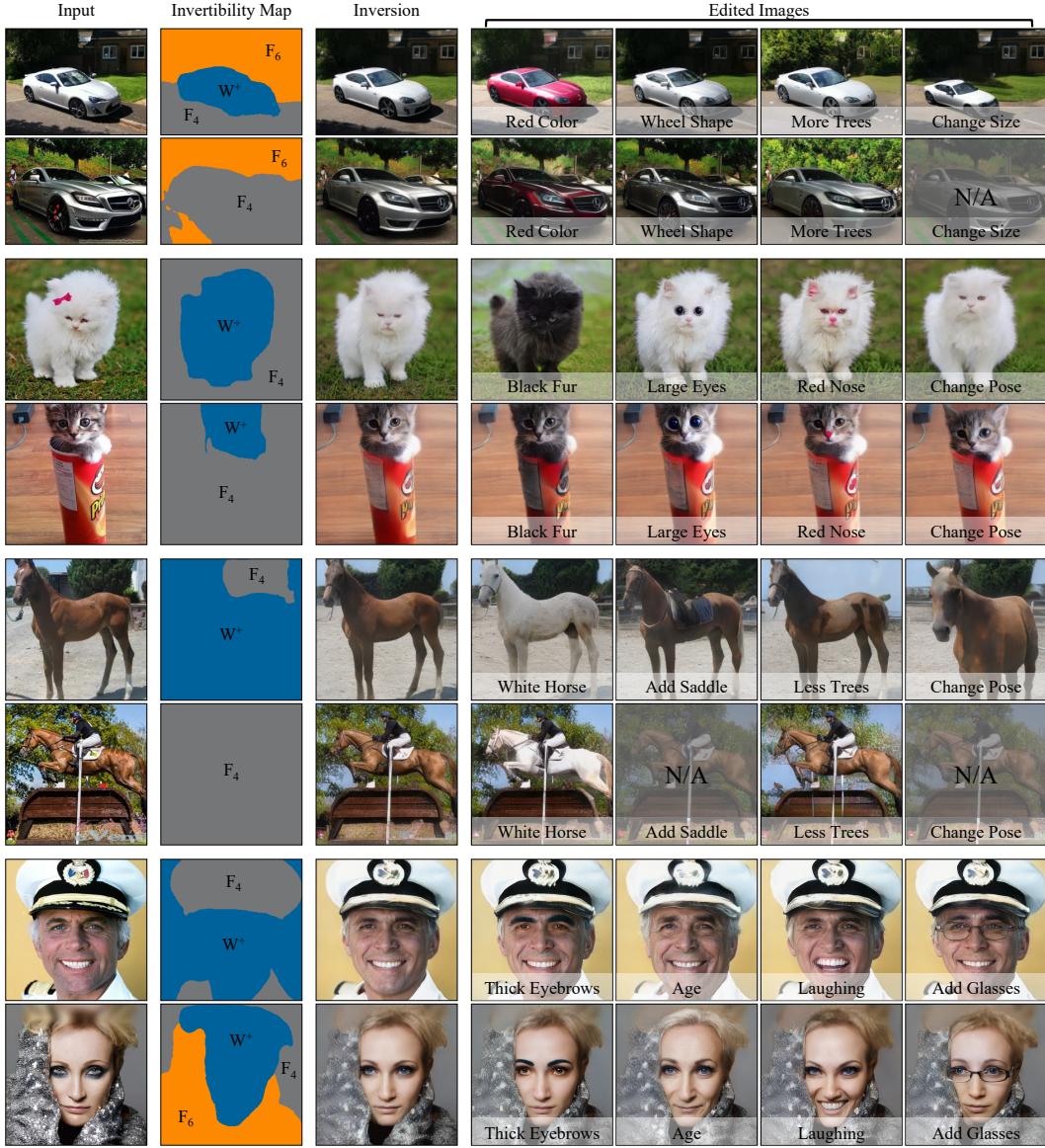
Figure 4.2: **Qualitative inversion and editing results.** In the first column we show input images for which we predict the invertibility map shown in the second column. We are able to obtain inverted images which closely match the input as shown in third column. In the remaining columns, we show our edit results. We can apply complex spatial edits like pose and size changes in seamless fashion even though different segments were inverted in different latent spaces.

25
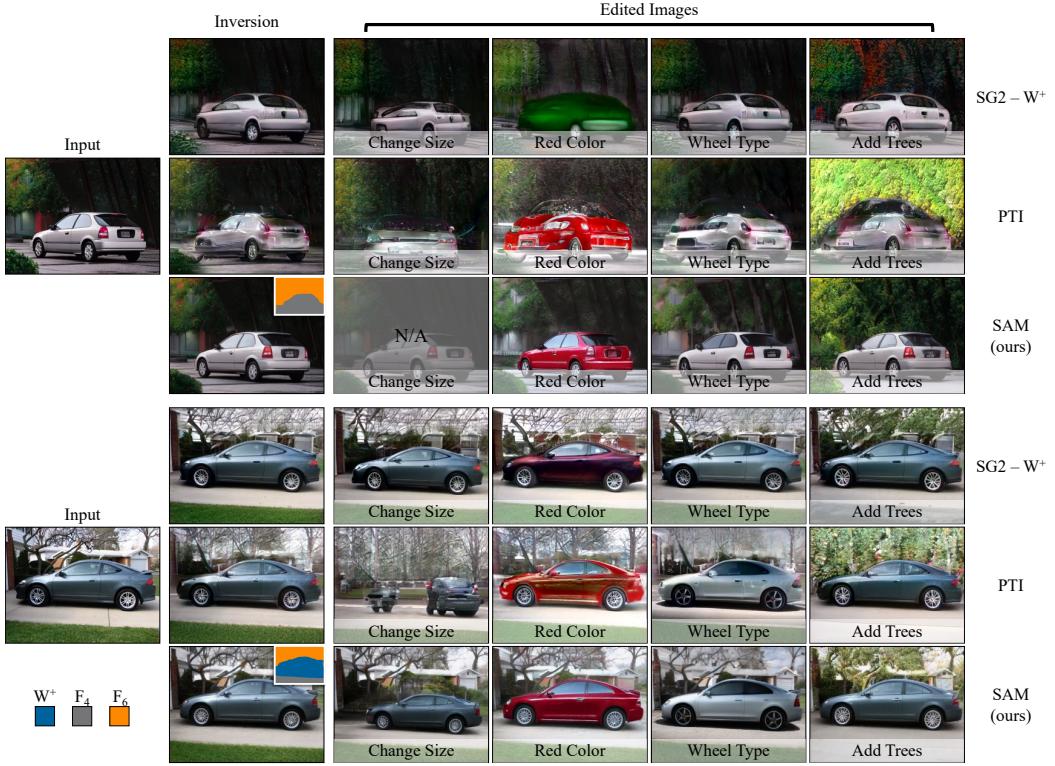
Figure 4.3: **Comparison with other optimization based inversion methods.** We compare our inversion and editing results with StyleGAN2 W^+ inversion and pivotal tuning. We obtain much closer and detailed inversion to the target image compared to other approaches. Also, we are able to apply semantic edits while maintaining the realism of the image. We are able to perform both low level edits like color change as well as high level edits like size changes.

Input	Invertibility Map	Inversion

F_2

Z^+

Edit latent codes

Add Grass | Change Pose | Change Pose

Edit class embedding

Great Pyrenees | Golden Retriever | Rottweiler
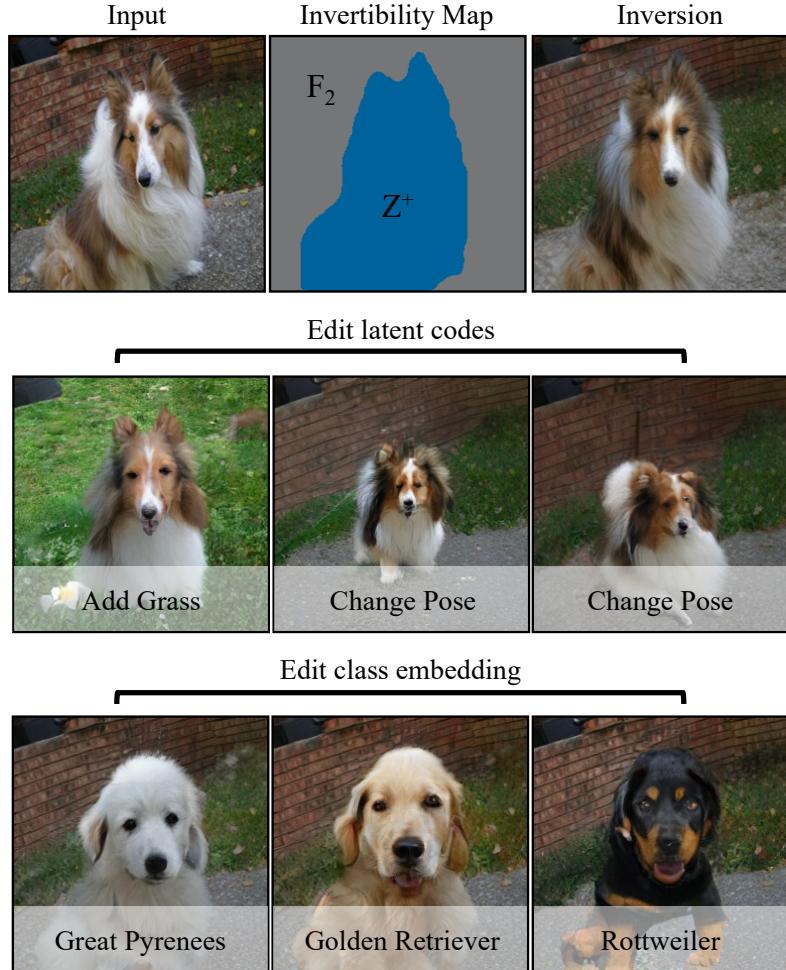
Figure 4.4: **Inversion and editing using BigGAN-deep.** We show that our spatially-adaptive method of using different latent layers (Z^+, F_2) can be applied to class-conditional models such as BigGAN-deep [12] trained on ImageNet. In the third column we show that the inversion obtained is very close to the input image. Subsequent edits can be performed using either changing the latent code (middle row) or modifying class embedding vector (bottom row).
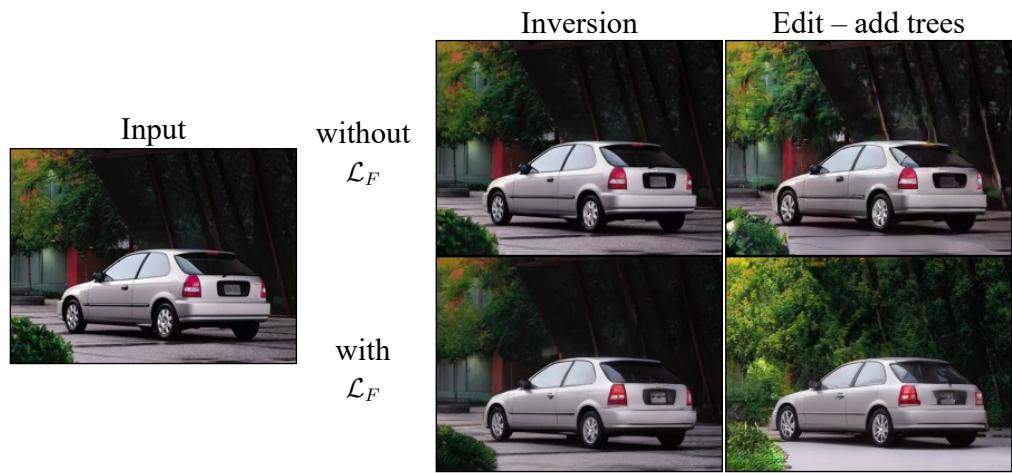
Figure 4.5: **Regularization in the features space F.** In top row, we show results without feature space regularization. We can see that edit to add tree does not work as without regularization, our predicted feature space may not be close original feature space distribution and edit direct would not be compatible anymore.
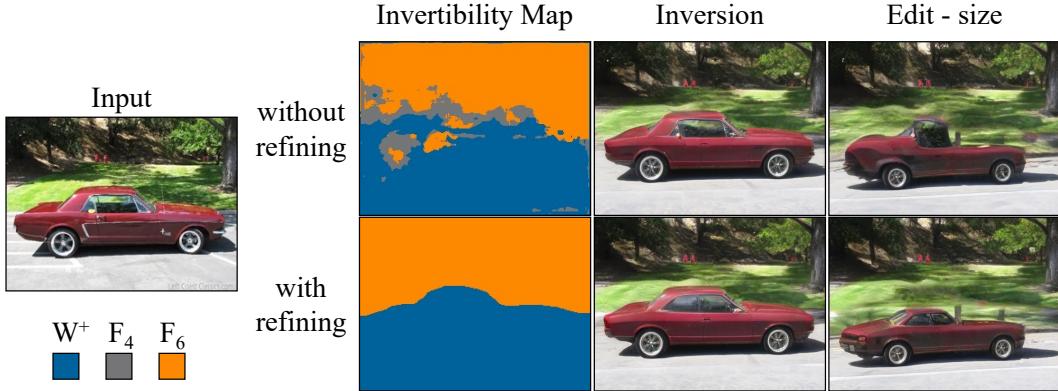
Figure 4.6: **Necessity for refining the invertibility map.** In top row, without refining the car segment get assigned to multiple feature space which results in inconsistent edit with artifacts. Where as with refining in bottom row, the entire car region get assigned to W^+ space which gives us consistent edit of changing the car size.
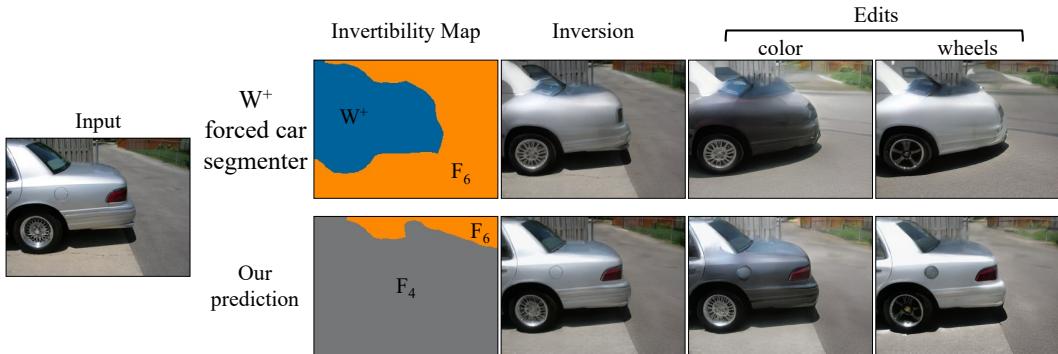
Figure 4.7: **Using a car segmenter.** We first invert a given target image with the assumption that the car regions of the image should be invertible with the native W^+. This assumption leads to a poor inversion that is not able to reconstruct the target car image. Whereas our method correctly predicts a good latent space for the different regions and consequently generated better inversions and edits which retain the identity of the original car better.

Chapter 5

Conclusions

Our key idea is that different regions of an image are best inverted using different latent layers. We use this insight to train networks that predict the "inversion difficulty" of dif- ferent latent layers for any given input image. Image regions that are easy to reconstruct can be inverted using early latent layers, whereas difficult image regions should use the more capable feature space of the intermediate layers. We show inversion and editing results using our proposed mul- tilayer inversion method on multiple challenging datasets. A limitation of this approach is that if a given input image is extremely difficult, our method will predict the use of the later latent layer that will correspond to being able to edit only limited things.

Bibliography

[1] R. Abdal, Y. Qin, and P. Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[2] R. Abdal, Y. Qin, and P. Wonka. Image2stylegan++: How to edit the embedded images? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[3] R. Abdal, P. Zhu, N. J. Mitra, and P. Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (TOG)*, 40(3):1–21, 2021.

[4] Y. Alaluf, O. Patashnik, and D. Cohen-Or. Only a matter of style: Age transformation using a style-based regression model. *ACM Trans. Graph.*, 40(4), 2021.

[5] Y. Alaluf, O. Patashnik, and D. Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021.

[6] R. Anirudh, J. J. Thiagarajan, B. Kailkhura, and P.-T. Bremer. Mimicgan: Robust projection onto image manifolds with corruption mimicking. *International Journal of Computer Vision*, pages 1–19, 2020.

[7] M. Asim, F. Shamshad, and A. Ahmed. Blind image deconvolution using deep generative priors. *arXiv preprint arXiv:1802.04073*, 2018.

[8] D. Bau, H. Strobelt, W. Peebles, J. Wulff, B. Zhou, J.-Y. Zhu, and A. Torralba. Semantic photo manipulation with a generative image prior. *ACM SIGGRAPH*, 38(4):1–11, 2019.

[9] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba. Gan dissection: Visualizing and understanding generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2019.

[10] D. Bau, J.-Y. Zhu, J. Wulff, W. Peebles, H. Strobelt, B. Zhou, and A. Torralba. Seeing what a gan cannot generate. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[11] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying mmd gans. In *ICLR*, 2018.

[12] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations (ICLR)*, 2019.

[13] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. Neural photo editing with introspective adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[14] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. Neural photo editing with introspective adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[15] L. Chai, J. Wulff, and P. Isola. Using latent space regression to analyze and leverage compositionality in gans. In *International Conference on Learning Representations (ICLR)*, 2021.

[16] L. Chai, J.-Y. Zhu, E. Shechtman, P. Isola, and R. Zhang. Ensembling with deep generative views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14997–15007, 2021.

[17] E. Collins, R. Bala, B. Price, and S. Susstrunk. Editing in style: Uncovering the local semantics of gans. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[18] D. Futschik, M. Lukáč, E. Shechtman, and D. Sỳkora. Real image inversion via segments. *arXiv preprint arXiv:2110.06269*, 2021.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

[20] J. Gu, Y. Shen, and B. Zhou. Image processing using multi-code gan prior. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[21] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris. Ganspace: Discovering interpretable gan controls. In *Advances in Neural Information Processing Systems*, 2020.

[22] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017.

[23] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[24] M. Huh, R. Zhang, J.-Y. Zhu, S. Paris, and A. Hertzmann. Transforming and projecting images into class-conditional generative networks. In *European Conference on Computer Vision (ECCV)*, 2020.

[25] A. Jahanian, L. Chai, and P. Isola. On the "steerability" of generative adversarial networks. In *International Conference on Learning Representations*, 2020.

[26] O. Kafri, O. Patashnik, Y. Alaluf, and D. Cohen-Or. Stylefusion: A generative model for disentangling spatial segments. *arXiv preprint arXiv:2107.07437*, 2021.

[27] K. Kang, S. Kim, and S. Cho. Gan inversion for out-of-range images with geometric transformations. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[28] A. Karnewar and O. Wang. Msg-gan: Multi-scale gradients for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[29] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations (ICLR)*, 2018.

[30] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila. Training generative adversarial networks with limited data. *NIPS*, 33, 2020.

[31] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila. Alias-free generative adversarial networks. *arXiv preprint arXiv:2106.12423*, 2021.

[32] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[33] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of stylegan. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[34] H. Kim, Y. Choi, J. Kim, S. Yoo, and Y. Uh. Exploiting spatial dimensions of latent in gan for real-time image editing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[35] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In *International Conference on Machine Learning (ICML)*, 2016.

[36] Z. C. Lipton and S. Tripathi. Precise recovery of latent vectors from generative adversarial networks. *arXiv preprint arXiv:1702.04782*, 2017.

[37] X. Luo, X. Zhang, P. Yoo, R. Martin-Brualla, J. Lawrence, and S. M. Seitz. Time-travel rephotography. *arXiv preprint arXiv:2012.12261*, 2020.

[38] X. Pan, X. Zhan, B. Dai, D. Lin, C. C. Loy, and P. Luo. Exploiting deep generative prior for versatile image restoration and manipulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2021.

[39] T. Park, J.-Y. Zhu, O. Wang, J. Lu, E. Shechtman, A. A. Efros, and R. Zhang. Swapping autoencoder for deep image manipulation. In *Advances in Neural Information Processing Systems*, 2020.

[40] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[41] W. Peebles, J. Peebles, J.-Y. Zhu, A. Efros, and A. Torralba. The hessian penalty: A weak prior for unsupervised disentanglement. In *European Conference on Computer Vision (ECCV)*. Springer, 2020.

[42] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez. Invertible conditional gans for image editing. In *NIPS Workshop on Adversarial Training*, 2016.

[43] E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. *arXiv preprint arXiv:2008.00951*, 2020.

[44] D. Roich, R. Mokady, A. H. Bermano, and D. Cohen-Or. Pivotal tuning for latent-based editing of real images. *arXiv preprint arXiv:2106.05744*, 2021.

[45] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[46] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, 2016.

[47] Y. Shen, C. Yang, X. Tang, and B. Zhou. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[48] Y. Shen and B. Zhou. Closed-form factorization of latent semantics in gans. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[49] R. Suzuki, M. Koyama, T. Miyato, T. Yonetsuji, and H. Zhu. Spatially controllable image synthesis with internal representation collaging. *arXiv preprint arXiv:1811.10153*, 2018.

[50] A. Tewari, M. Elgharib, G. Bharaj, F. Bernard, H.-P. Seidel, P. Pérez, M. Zollhofer, and C. Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[51] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or. Designing an encoder for stylegan image manipulation. In *ACM SIGGRAPH*, 2021.

[52] A. Voynov and A. Babenko. Unsupervised discovery of interpretable directions in the gan latent space. In *International Conference on Machine Learning (ICML)*, 2020.

[53] T. Wei, D. Chen, W. Zhou, J. Liao, W. Zhang, L. Yuan, G. Hua, and N. Yu. A simple baseline for stylegan inversion. *arXiv preprint arXiv:2104.07661*, 2021.

[54] Y. Wu, X. Wang, Y. Li, H. Zhang, X. Zhao, and Y. Shan. Towards vivid and diverse image colorization with generative color prior. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[55] Z. Wu, D. Lischinski, and E. Shechtman. Stylespace analysis: Disentangled controls for stylegan image generation. In *CVPR*, 2021.

[56] J. Wulff and A. Torralba. Improving inversion and generation diversity in stylegan using a gaussianized latent space. *arXiv preprint arXiv:2009.06529*, 2020.

[57] W. Xia, Y. Zhang, Y. Yang, J.-H. Xue, B. Zhou, and M.-H. Yang. Gan inversion: A survey. *arXiv preprint arXiv:2101.05278*, 2021.

[58] Y. Xu, Y. Du, W. Xiao, X. Xu, and S. He. From continuity to editability: Inverting gans with consecutive images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13910–13918, 2021.

[59] R. A. Yeh, C. Chen, T. Yian Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do. Semantic image inpainting with deep generative models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[60] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[61] S. Zhao, Z. Liu, J. Lin, J.-Y. Zhu, and S. Han. Differentiable augmentation for data-efficient gan training. In *Advances in Neural Information Processing Systems*, 2020.

[62] J. Zhu, Y. Shen, D. Zhao, and B. Zhou. In-domain gan inversion for real image editing. In *ECCV*, 2020.

[63] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision (ECCV)*, 2016.

[64] P. Zhu, R. Abdal, J. Femiani, and P. Wonka. Barbershop: Gan-based image compositing using segmentation masks. *arXiv preprint arXiv:2106.01505*, 2021.

[65] P. Zhu, R. Abdal, Y. Qin, J. Femiani, and P. Wonka. Improved stylegan embedding: Where are the good latents? *arXiv preprint arXiv:2012.09036*, 2020.