This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

ReStyle: A Residual-Based StyleGAN Encoder via Iterative Refinement

Yuval Alaluf Or Patashnik Daniel Cohen-Or

Blavatnik School of Computer Science, Tel Aviv University

Abstract

Recently, the power of unconditional image synthesis has significantly advanced through the use of Generative Adversarial Networks (GANs). The task of inverting an image into its corresponding latent code of the trained GAN is of utmost importance as it allows for the manipulation of real images, leveraging the rich semantics learned by the network. Recognizing the limitations of current inversion approaches, in this work we present a novel inversion scheme that extends current encoder-based inversion methods by introducing an iterative refinement mechanism. Instead of directly predicting the latent code of a given real image using a single pass, the encoder is tasked with predicting a residual with respect to the current estimate of the inverted latent code in a self-correcting manner. Our residual-based encoder, named ReStyle, attains improved accuracy compared to current state-of-the-art encoder-based methods with a negligible increase in inference time. We analyze the behavior of ReStyle to gain valuable insights into its iterative nature. We then evaluate the performance of our residual encoder and analyze its robustness compared to optimization-based inversion and state-of-the-art encoders. Code is available via our project page: https://yuval-alaluf.github.io/restyle-encoder/

1. Introduction

Recently, Generative Adversarial Networks (GANs) have grown in popularity thanks to their ability to synthesize images of high visual quality and diversity. Beyond their phenomenal realism and fidelity on numerous domains, recent works have shown that GANs, e.g., StyleGAN [24, 25, 23], effectively encode semantic information in their latent spaces [16, 36, 21]. Notably, it has been shown that StyleGAN's learnt latent space $\mathcal{W}$ has disentanglement properties [9, 36, 46] which allow one to perform extensive image manipulations by leveraging a well-trained StyleGAN generator. Such manipulations, however, have often been applied to synthetic images generated by the GAN itself. To apply such edits on *real* images, one must first *invert* the given image into StyleGAN's latent space. That is, retrieve the latent code $\mathbf{w}$ such that passing $\mathbf{w}$ to the pre-trained StyleGAN generator returns the original image. To do so, it has become common practice to invert real images into an extension of $\mathcal{W}$, denoted $\mathcal{W}+$ [1].

Input Iterative Outputs ⟶

Figure 1. Different from conventional encoder-based inversion techniques, our residual-based ReStyle scheme incorporates an iterative refinement mechanism to progressively converge to an accurate inversion of real images. For each domain, we show the input image on the left followed by intermediate inversion outputs.

Previous works have explored learning-based inversion approaches and train encoders to map a given real image into its corresponding latent code [10, 32, 50, 15, 35, 40]. Compared to per-image latent vector optimization [28, 10, 1, 2, 25], encoders are significantly faster, as they invert using a single forward pass, and converge to areas of the latent space which are more suitable for editing [50, 40]. However, in terms of reconstruction accuracy, there remains a significant gap between learning-based and optimization-based inversion methods. Hence, while significant progress has been made in learning-based inversions, designing a proper encoder and training scheme remains a challenge with many works still resorting to using a per-image optimization.

Recognizing that obtaining an accurate inversion in a single shot is difficult, we introduce a novel encoder-based inversion scheme tasked with encoding real images into the extended $\mathcal{W}+$ StyleGAN latent space. Unlike typical encoder-based inversion methods that infer the input's inverted latent code using a single forward pass, our scheme introduces an iterative feedback mechanism. Specifically, the inversion is performed using several forward passes by feeding the encoder with the output of the previous iteration along with the original input image. This allows the encoder to leverage knowledge learned in previous iterations to focus on the relevant regions needed for achieving an accurate reconstruction of the input image. Viewing this formulation in terms of the latent space, our *residual encoder* is trained to predict the residual, or an offset, between the current latent code and the new latent code at each step. Doing so allows the encoder to progressively converge its inversion toward the target code and reconstruction, see Figure 1. Note also that the inversion is predicted solely using the encoder with *no* per-image optimization performed thereafter.

In a sense, our inversion scheme, named *ReStyle*, can be viewed as *learning* to perform a *small* number of steps (e.g., 10) in a residual-based manner within the latent space of a pre-trained unconditional generator. ReStyle is generic in the sense that it can be applied to various encoder architectures and loss objectives for the StyleGAN inversion task.

We perform extensive experiments to show that ReStyle achieves a significant improvement in reconstruction quality compared to standard feed-forward encoders. This is achieved with a negligible increase in inference time, which is still an order of magnitude faster than the time-costly optimization-based inversion. We also analyze the iterative nature of our approach. Specifically, we first demonstrate which image regions are refined at each iterative feedback step demonstrating that our scheme operates in a coarse-to-fine manner. Second, we show that the absolute magnitude of change at each step decreases, with the predicted residuals converging after only a small number of steps.

To demonstrate the generalization of ReStyle beyond the StyleGAN inversion task and its appealing properties compared to current inversion techniques, we continue our analysis by exploring the robustness of our scheme on downstream tasks and special use-cases. To this end, we perform latent space manipulations [16, 36, 37] on the inverted latent codes to see if the embeddings are semantically meaningful. We then explore an *encoder bootstrapping* technique allowing one to leverage two well-trained encoders to obtain a more faithful translation of a given real image.

2. Background and Related Works

The idea of employing an iterative refinement scheme is not new. Carreira *et al*. [6] introduced an iterative feedback mechanism for human pose estimation. Other works have proposed using iterative refinement for optical flow [20], object pose estimation [43, 18], object detection [34], and semantic segmentation [48] among other tasks. To the best of our knowledge, we are the first to adopt an iterative refinement approach for a learned inversion of real images.

2.1. GAN Inversion

The task of *GAN Inversion* was first introduced by Zhu *et al*. [51] for projecting real images into their latent representations. In their pioneering work, the authors demonstrate how performing such an inversion enables one to leverage the semantics of the GAN's latent space for performing various image manipulation tasks. Some works [51, 28, 10, 1, 2, 25, 39] approach this task by directly optimizing the latent vector to minimize the reconstruction error for a given image. These works typically achieve high reconstruction quality but require several minutes per image. Other approaches design an encoder [51, 32, 50, 15, 35, 40] to learn a direct mapping from a given image to its corresponding latent vector. While these methods are substantially more efficient than pure optimization, they typically achieve inferior reconstruction quality. Attempting to balance this trade-off, some works have additionally proposed a hybrid approach and combine the two by using an encoder for initializing the optimization [51, 5, 15, 50]. We refer the reader to Xia *et al*. [44] for a comprehensive survey on GAN inversion.

2.2. Latent Space Embedding via Learned Encoders

To perform image manipulations on real images, methods typically follow an "invert first, edit later" approach. There, an image is first embedded into its corresponding latent code, which is then edited in a semantically meaningful manner. Diverging from the above, recent works [30, 35, 4, 7] have proposed end-to-end methods for leveraging the high-quality images generated by GANs for various image-to-image translation and image editing tasks. In these works, a real input image is directly encoded into the transformed latent code which is then fed into the generator to obtain the desired transformed image. By training an encoder with some additional constraint, these works are able to directly solve various tasks without the need for inverting the images beforehand. Other works [45] have explored utilizing features produced by a learned StyleGAN encoder for solving various down-stream tasks such as face verification and layout prediction. These works further emphasize the advantage of training a powerful encoder into the latent space of a pre-trained unconditional generator.

2.3. Latent Space Manipulation

With the recent advancements in image synthesis through GANs [14], many works have proposed diverse methods for understanding and controlling their latent representations for performing extensive image manipulations.

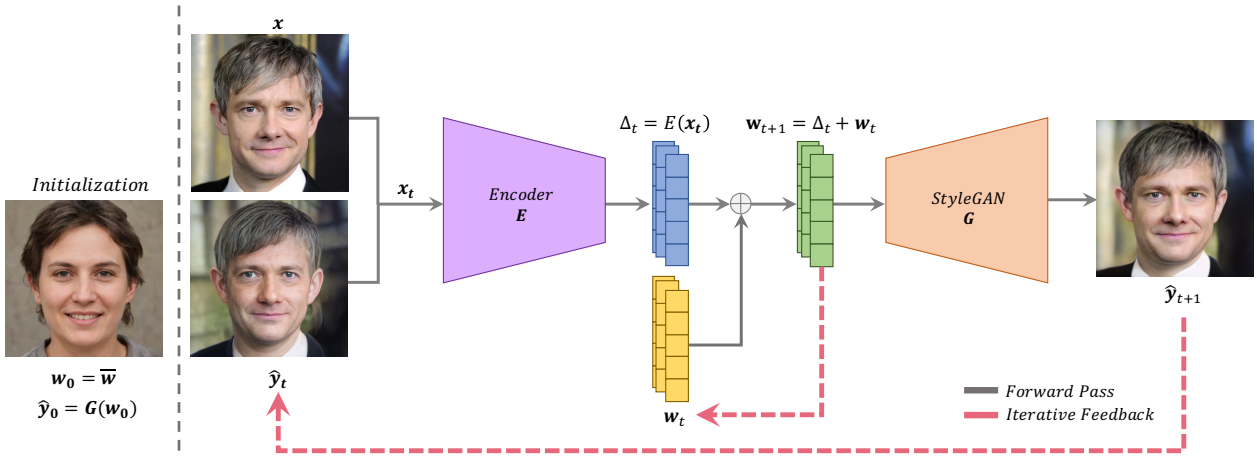

Figure 2. ***Our ReStyle iterative inversion scheme***. Given an input image $\mathbf{x}$, the scheme is initialized with the average latent code $\mathbf{w}_0$ and its corresponding image $\hat{\mathbf{y}}_0$. Consider step t. ReStyle operates on an extended input obtained by concatenating $\mathbf{x}$ with the image $\hat{\mathbf{y}}_t$ corresponding to the current inversion prediction $\mathbf{w}_t \in \mathcal{W}+$ (shown in yellow). The encoder E is then tasked with predicting a *residual* latent code, $\Delta_t \in \mathcal{W}+$ (shown in blue). The predicted residual is then added to the previous latent code $\mathbf{w}_t$ to obtain the updated latent code prediction $\mathbf{w}_{t+1}$ (shown in green). Finally, passing the newly computed latent code to the generator G results in an updated reconstruction $\hat{\mathbf{y}}_{t+1}$, which is then passed as input in the following step. During training, the loss objectives are computed at each forward pass with back-propagation performed accordingly. A similar multi-step process is performed during inference.

Various works [12, 13, 36] use fully-supervised approaches to find latent directions corresponding to various attributes such as age, gender, and expression. On the other end of the supervision spectrum, several methods [16, 41, 42] find directions in a completely unsupervised manner. Others have explored techniques that go beyond a linear traversal of the latent space. Tewari *et al*. [38] employ a pre-trained 3DMM to learn semantic face edits. Shen *et al*. [37] learn versatile edit directions through the eigenvector decomposition of the generator weights. Abdal *et al*. [3] learn non-linear paths via normalizing flows conditioned on a target attribute. Finally, Patashnik *et al*. [31] utilize CLIP to manipulate images using an input text-prompt. By designing an efficient and accurate inversion method, one is able to leverage these works for manipulating real images.

3. Preliminaries

3.1. Encoder-Based Inversion Methods

Recall that our goal is to train an encoder tasked with inverting real images into the latent space of a pre-trained StyleGAN generator. Let E and G denote our encoder and StyleGAN generator, respectively. Given a source image $\mathbf{x}$, our goal is to generate an image $\hat{\mathbf{y}} = G(E(\mathbf{x}))$ such that $\hat{\mathbf{y}} \approx \mathbf{x}$. Observe that in conventional encoder-based inversion methods, the reconstructed image $\hat{\mathbf{y}}$ is simply computed using a single forward pass through E and G via StyleGAN's latent space representation.

For learning to perform the inversion, these methods introduce a set of losses used to train the encoder network E on the reconstruction task. For training the encoder, most encoder-based methods employ a weighted combination of a pixel-wise L2 loss and a perceptual loss (e.g., LPIPS [49]) to guide the training process. Recently, Richardson *et al*. [35] extend these losses and introduce a dedicated identity loss to achieve improved reconstruction on the human facial domain. To attain improved editability over the inverted latent codes, Tov *et al*. [40] additionally introduce two regularization losses during training. Observe that during training, the pre-trained generator network G typically remains fixed.

4. Method

We now turn to describe our ReStyle scheme and build on the conventional, single-pass encoding approach introduced above. Given an input image $\mathbf{x}$, ReStyle performs $N > 1$ steps to predict the image inversion $\mathbf{w} = E(\mathbf{x})$ and corresponding reconstruction $\hat{\mathbf{y}}$. Here, we define a *step* to be a single forward pass through E and G. As such, observe that the conventional encoding process, being performed with a single step, is a special case of ReStyle where $N = 1$.

For training the encoder network E, we define a single training *iteration* to be a set of N steps performed on a batch of images. As with conventional encoding schemes, ReStyle uses a curated set of loss objectives for training E on the inversion task while the pre-trained generator G remains fixed. Observe that the loss objectives are computed at each forward pass (i.e., step) with the encoder weights updated accordingly via back-propagation (i.e., back-propagation occurs N times per batch).

During inference, the same multi-step process (without the loss computation) is performed to compute the image inversion and reconstruction. Notably, for a given batch of images, we find that a small number of steps are needed for convergence (e.g., $N < 10$), resulting in fast inference time.

We now turn to more formally describe ReStyle's inversion process, illustrated in Figure 2. At each step t, ReStyle operates on an expanded input by concatenating $\mathbf{x}$ with the current prediction for the reconstructed image $\hat{\mathbf{y}}_t$:

$$\mathbf{x}_t := \mathbf{x} \parallel \hat{\mathbf{y}}_t. \tag{1}$$

Given the extended 6-channel input $\mathbf{x}_t$, the encoder E is tasked with computing a residual code Δ_t, with respect to the latent code predicted in the previous step. That is,

$$\Delta_t := E(\mathbf{x}_t). \tag{2}$$

The new prediction for the latent code corresponding to the inversion of the input image $\mathbf{x}$ is then updated as:

$$\mathbf{w}_{t+1} \leftarrow \Delta_t + \mathbf{w}_t. \tag{3}$$

This new latent $\mathbf{w}_{t+1}$ is passed through the generator G to obtain the updated prediction for the reconstructed image:

$$\hat{\mathbf{y}}_{t+1} := G(\mathbf{w}_{t+1}). \tag{4}$$

Finally, the updated prediction $\hat{\mathbf{y}}_{t+1}$ is set as the additional input channels in the next step, as defined by Equation 1.

This procedure is initialized with an initial guess $\mathbf{w}_0$ and corresponding image $\hat{\mathbf{y}}_0$. In our experiments, these are set to be the generator's average style vector and its corresponding synthesized image, respectively.

Observe that constraining the encoder to invert the given image in a single step, as is typically done, imposes a hard constraint on the training process. Conversely, our training scheme can, in a sense, be viewed as relaxing this constraint. In the above formulation, the encoder learns how to best take *several* steps in the latent space with respect to an initial guess $\mathbf{w}_0$ guided by the output obtained in the previous step. This relaxed constraint allows the encoder to iteratively narrow down its inversion to the desired target latent code in a self-correcting manner. One may also view the ReStyle steps in a similar manner to the steps of optimization, with the key difference that here the steps are *learned* by the encoder for efficiently performing the inversion.

4.1. Encoder Architecture

To show that the presented training scheme can be applied to different encoder architectures and loss objectives, we apply the ReStyle scheme on the state-of-the-art encoders from Richardson *et al*. [35] (pSp) and Tov *et al*. [40] (e4e). These two encoders employ a Feature Pyramid Network [27] over a ResNet [17] backbone and extract the style features from three intermediate levels. Such a hierarchical encoder is well-motivated for well-structured domains such as the facial domain in which the style inputs can be roughly divided into three levels of detail. With that, we find such a design to have a negligible impact on less-structured, multi-modal domains while introducing an increased overhead. Moreover, we find that the multi-step nature of ReStyle alleviates the need for such a complex encoder architecture.

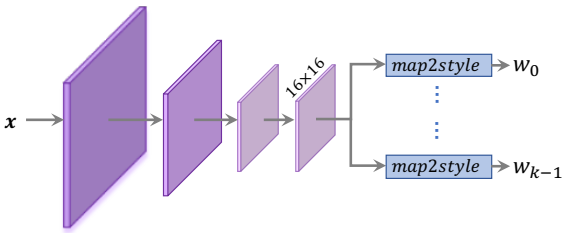

Figure 3. *Our simplified encoder architecture.* All k input style vectors of the generator are extracted from the encoder's final 16×16 feature map which is passed through k *map2style* blocks [35].

We therefore choose to design simpler variants of the pSp and e4e encoders. Rather than extracting the style features from three intermediate levels along the encoder, all style vectors are extracted from the final 16×16 feature map. Given a StyleGAN generator with k style inputs, k different *map2style* blocks introduced in pSp are then used to down-sample the feature map to obtain the corresponding 512-dimensional style input. A high-level overview of the architecture is provided in Figure 3 with additional details and ablations provided in the supplementary materials.

5. Experiments

5.1. Settings

Datasets. We conduct extensive evaluations on a diverse set of domains to illustrate the generalization of our approach. For the human facial domain we use the FFHQ [24] dataset for training and the CelebA-HQ [29, 22] test set for evaluation. For the cars domains, we use the Stanford Cars [26] dataset for training and evaluation. Additional evaluations are performed on the LSUN [47] Horse and Church datasets as well as the AFHQ Wild [8] dataset.

Baselines. Throughout this section, we explore and analyze encoder-based, optimization-based, and hybrid inversion techniques. For encoder-based methods, we compare our ReStyle approach with the IDInvert encoder from Zhu *et al*. [50], pSp from Richardson *et al*. [35], and e4e from Tov *et al*. [40]. For optimization-based methods, we compare our results with the inversion technique from Karras *et al*. [25]. For each of the above encoder-based inversion methods we also perform optimization on the resulting latents for a comparison with hybrid approaches. Additional details can be found in the supplementary materials.

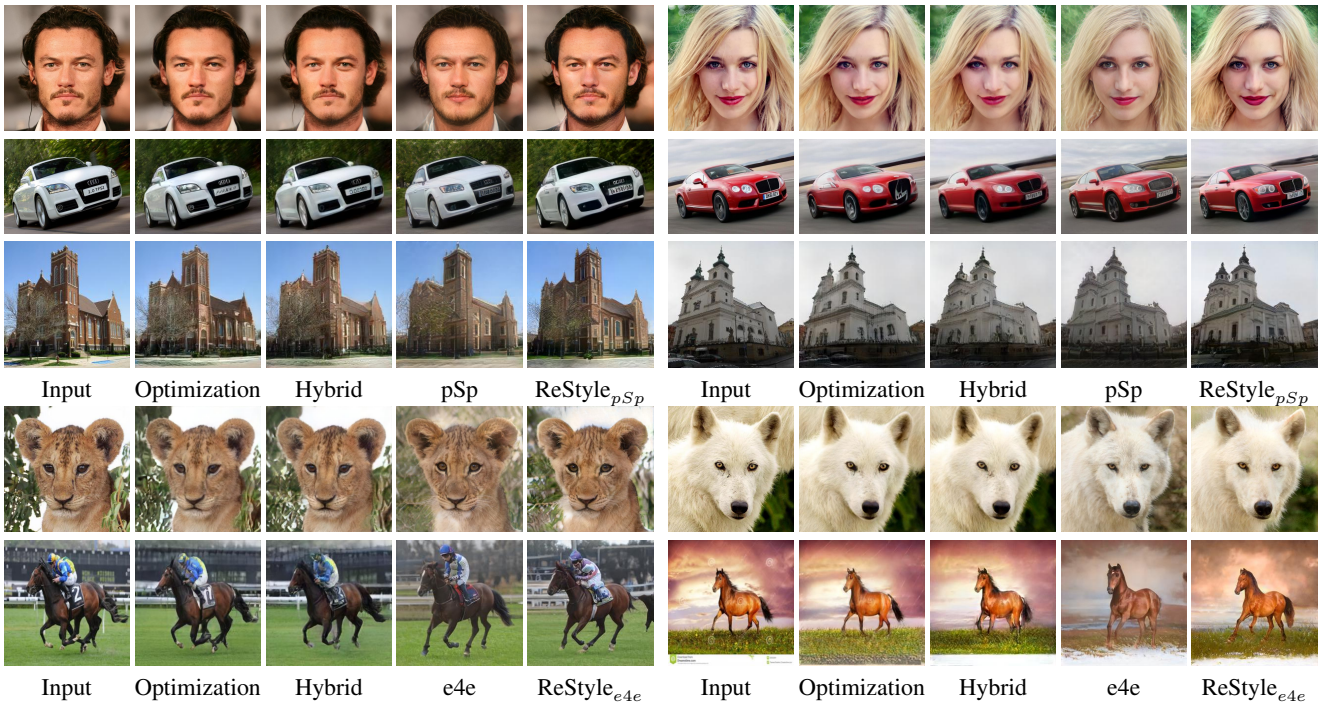

Figure 4. *Qualitative Comparison.* We compare various encoder-based and optimization-based inversion methods with our ReStyle scheme applied over pSp [35] and e4e [40] (denoted by ReStyle$_{pSp}$ and ReStyle$_{e4e}$). Hybrid results are obtained by performing optimization on the latent codes obtained by the adjacent encoder. Additional comparisons in the supplementary materials. Best viewed zoomed-in.

Architecture and Training Details. For the facial domain, we employ the ResNet-IRSE50 architecture from Deng *et al*. [11] pre-trained for facial recognition. For all other domains, we use a ResNet34 network pre-trained on ImageNet. These networks have a modified input layer to accommodate the 6-channel input used by ReStyle. All results were obtained using StyleGAN2 [25] generators.

Throughout this section, we apply ReStyle on pSp [35] and e4e [40] using the loss objectives and training details (e.g., batch size, loss weights) as originally defined in their respective works. Note that when applying ReStyle, we utilize the simplified encoder architecture presented in Section 4.1 for extracting the image inversion. All ReStyle encoders are trained using $N = 5$ steps per batch.

5.2. Comparison with Inversion Methods

We first compare ReStyle with current state-of-the-art StyleGAN inversion techniques. While per-image optimization techniques have achieved superior image reconstruction compared to learning-based approaches, they come with a significantly higher computational cost. Therefore, when analyzing the inversion approaches, it is essential to measure reconstruction quality with respect to inference time, resulting in a so-called *quality-time trade-off*.

Qualitative Evaluation. We begin by showing a qualitative comparison of ReStyle and the alternative inversion approaches across various domains in Figure 4. It is important to emphasize that we do not claim to achieve superior reconstruction quality over optimization. The comparison instead serves to show that ReStyle is visually comparable to the latter. Attaining comparable reconstruction quality with a significantly lower inference time places ReStyle at an appealing point on the quality-time trade-off curve.

With that, we do note the improved reconstruction obtained by ReStyle in comparison with the pSp and e4e encoders, especially in the preservation of fine details. For example in the comparison with pSp (the top three rows), observe the collar of the man in the top left and the hair of the woman in the top right. Similarly, observe the Audi symbol and the license plate in the car comparison on the left-hand side. In the comparison with e4e (the bottom two rows), observe how ReStyle better captures the background of the wild animals and the pose of the horse.

Quantitative Evaluation. We now perform a quantitative comparison of the different inversion approaches across various data domains. To measure both pixel-wise and perceptual similarities we apply the commonly-used L_2 and LPIPS [49] metrics. In addition, for the human facial domain, to measure each method's ability to faithfully preserve identity, we measure the identity similarity between the reconstructed images and their source using the state-of-the-art CurricularFace [19] facial recognition method.

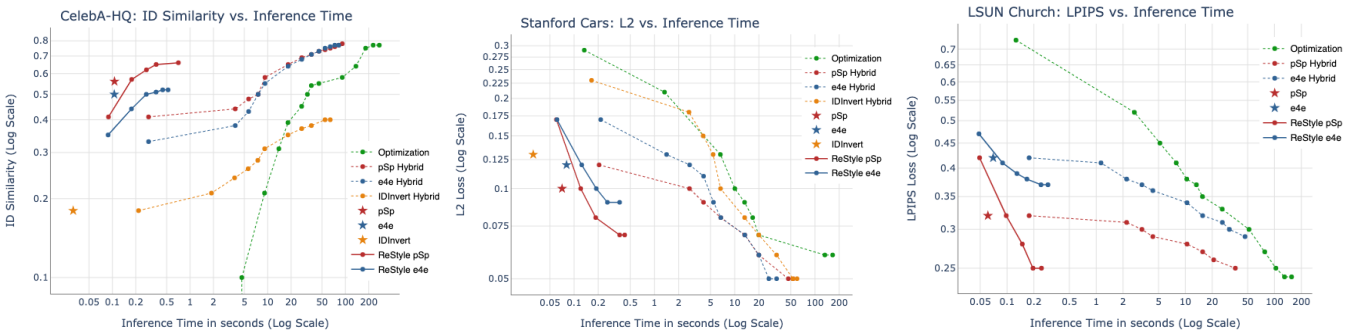

Figure 5. *Quantitative comparison.* We compare ReStyle with current state-of-the-art optimization-based and encoder-based methods by analyzing reconstruction via three evaluation metrics — ID similarity for faces, L2 loss for cars, and LPIPS loss for churches — while measuring each method's inference time. Each encoder-based method is represented using a $\star$ symbol. The corresponding hybrid method is marked using a dashed line of the same color with the ReStyle applied over the base method shown using a solid line of the same color. Optimization results are shown using a dashed green line. Methods based on pSp are shown in red with methods based on e4e shown in blue. Finally, results obtained using IDInvert [50] are shown in orange. Note that both axes are shown in log-scale.

To illustrate the trade-off between the different methods we additionally measure each method's inference time per image. As mentioned, both optimization and ReStyle can be viewed as a continuous curve on a quality-time graph — with each additional step, we attain improved reconstruction quality at the cost of additional inference time.

To provide a complete comparison of all inversion methods, we construct a quality-time graph for each domain. Such graphs can be visualized in Figure 5. To form each graph, we performed the following evaluations for each inversion technique. For each encoder-based inversion, we ran a single forward pass to obtain the reconstruction image, resulting in a single point on the graph. For measuring the optimization technique from [25], we invert the input image using a varying number of steps from 1 optimization step up to $1,500$ steps. For hybrid approaches, given the computed latent codes obtained from the corresponding encoder, we performed optimization with an increasing number of steps between 1 to 500 steps. Finally, for our two ReStyle encoders, we performed up to 10 feedback loops.

We begin by analyzing the facial domain. Compared to the conventional pSp and e4e encoders, our ReStyle variants match or surpass their counterparts. More notably, while optimization techniques achieve improved identity similarity compared to ReStyle, they require $\approx 20\times$ more time to match the similarity attained by ReStyle. A similar trade-off can be observed in the cars domain where now the advantage of ReStyle over typical encoders is more pronounced when evaluating the L_2 loss of the reconstructions. In the unstructured churches domain, ReStyle applied over pSp nearly matches both optimization and hybrid techniques in reconstruction quality with a significantly lower inference time. Observe that the first output of ReStyle may be *worse* than that of a conventional encoder due to the more relaxed training formulation of ReStyle as it is trained to perform multiple steps at inference. With that, ReStyle quickly matches or surpasses the quality of single-shot encoders.

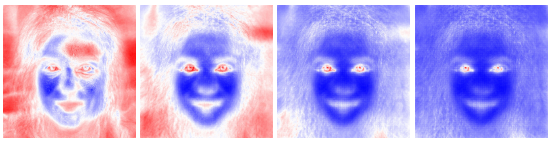

Figure 6. In each sub-image, we display a heatmap showing which image regions changed the most (in red) and which regions changed the least (in blue) between the specified iterations.

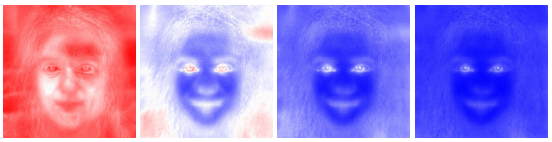

Figure 7. Similar to Figure 6, with the difference that here all images are normalized with respect to each other. As shown, the magnitude of change decreases with each step.

These comparisons point to the appealing nature of ReStyle: although optimization typically achieves superior reconstruction, ReStyle offers an excellent balance between reconstruction quality and inference time. See the supplementary materials for results on all domains and metrics.

5.3. ReStyle Analysis

In this section, we explore various aspects of ReStyle to gain a stronger understanding of its behavior and attain key insights into its efficiency. Specifically, we analyze the main details focused on by the encoder at each step and analyze the number of steps needed for convergence during inference. Additional analyses in both the image space and latent space can be found in the supplementary materials.

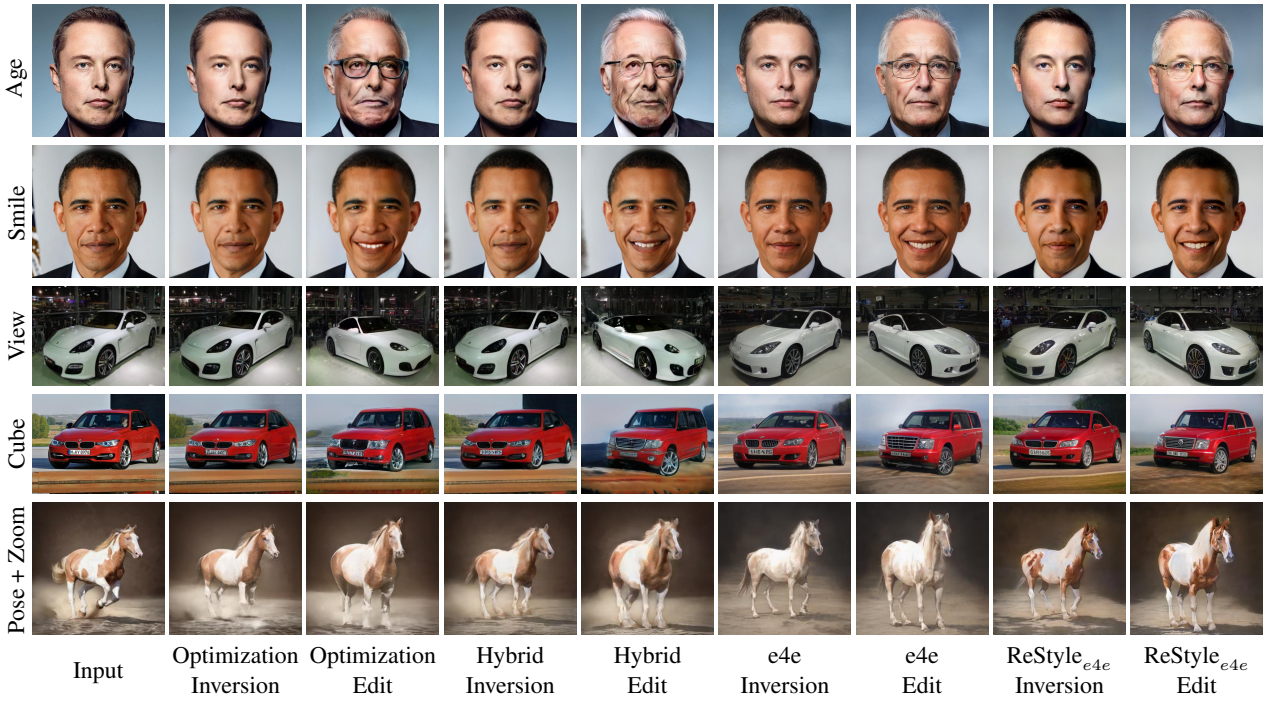

Figure 8. *Editing Comparison.* We apply edits on inversions obtained from several methods. For performing the edits in the human facial domain we use InterFaceGAN [36], for the cars domain we use GANSpace [16], and for the horse domain we use SeFa [37].

Where's the focus? We begin by exploring which regions of the image are focused on by the encoder at each step during inference. To do so, we consider the human facial domain. For each step t and each input image $\mathbf{x}$, we compute the squared difference in the image space between the generated images at steps t and $t-1$. That is, we compute $d = \left\|\mathbf{y}_t - \mathbf{y}_{t-1}\right\|_2$ where $\mathbf{y}_t$ is defined in Equation 4.

Averaging over all test samples we obtain the average image difference between the two steps. Finally, we normalize the average image to the range $[0, 1]$ and visualize the regions of the image that incur the most change at the current step t. We visualize this process in Figure 6 showing ReStyle's incremental refinements. As can be seen, in the early steps the encoder focuses on refining the background and pose while in subsequent steps the encoder moves its focus to adjusting finer details along the eyes and hair.

In Figure 6 we show only the magnitude of change *within* each step. That is, the absolute magnitude of change may vary between the different steps. To show that the overall amount of change decreases with each step, we refer the reader to Figure 7. There, all images are normalized with respect to each other allowing one to see how the largest changes occur in the first step and decrease thereafter.

In a sense, the encoder operates in a coarse-to-fine manner, beginning by concentrating on low frequency details which are then gradually complemented by adjusting high frequency, fine-level details.

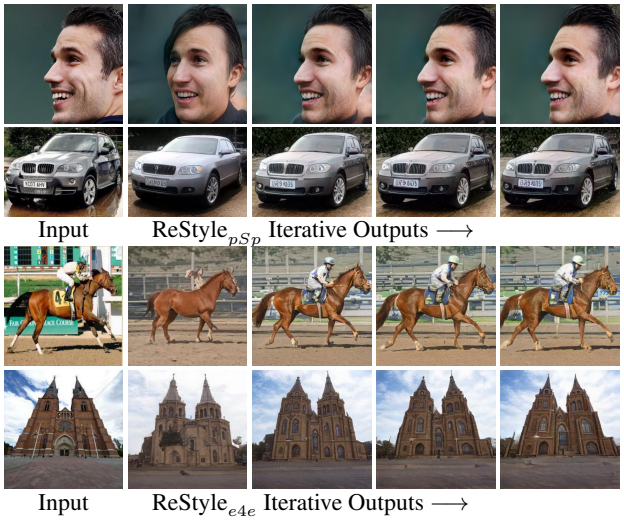

Figure 9. Given the input image on the left, we visualize the intermediate outputs of ReStyle applied over pSp [35] and e4e [40].

ReStyle's iterative progress. We now turn to Figure 9 and show how the reconstruction quality incrementally improves with each additional step of ReStyle. Specifically, observe how ReStyle$_{pSp}$ is able to gradually improve the reconstruction of the highly non-frontal input image in the top row. Similarly, notice how ReStyle$_{e4e}$ is able to iteratively refine the posture of the horse rider and capture the skewed structure of the church building.

5.4. Editability via Latent Space Manipulations

Previous works [50, 52, 40, 52] have discussed the importance of evaluating the editability of inversion methods. Here, we show that the editability achieved by ReStyle is comparable to that of the conventional encoders. Since e4e is designed specifically for image manipulations, we choose to show that inversions obtained by combining e4e with ReStyle are still editable. We show visual examples in Figure 8. Compared to e4e, ReStyle is able to better reconstruct the input while still allowing for realistic edits. Notably, observe the more plausible edits over ReStyle's inversions compared to those obtained via optimization. For example, observe the artifacts in the front bumpers of the car edits when applied over the optimization-based inversions.

5.5. Encoder Bootstrapping

Finally, we explore a new concept, which we call *encoder bootstrapping*. To motivate this idea, let us consider the image toonification task in which we would like to translate real face images into their toonified, or animated, version. Pinkney *et al*. [33] propose solving this image-to-image task by projecting each real input image to its closest toon image in the latent space of a toon StyleGAN obtained via fine-tuning the FFHQ StyleGAN generator. In a similar sense, ReStyle can be applied over pSp to solve this task. Here, ReStyle is initialized with the average toon latent code and its corresponding image. Then, N steps are performed to translate the image to its toonified version.

With encoder bootstrapping, we take a slightly different approach. Rather than initializing the iterative process using the average toon image, we first pass the given real image to an encoder tasked with embedding real images into the latent space of a StyleGAN trained on FFHQ. Doing so will result in an inverted code $\mathbf{w}_1$ and reconstructed image $\hat{\mathbf{y}}_1$. This inverted code and reconstructed image are then taken to initialize the toonification translation using ReStyle. This idea is illustrated in Figure 10. Notice that this technique is possible thanks to the residual nature of ReStyle. By utilizing the FFHQ encoder to obtain a better initialization, we are able to more easily learn a proper residual for translating the input image while more faithfully preserving identity.

We compare several real-to-toon variants in Figure 11. Observe how bootstrapping the toonification process with the FFHQ code results in translations able to better capture the input characteristics and toonify style. Observe the ability of the bootstrapped variant to better preserve make-up, eyeglasses, hairstyle, and expression. In Figure 12, we visualize the inverted real image used to initialize the toonify encoder followed ReStyle's toonified outputs.

The bootstrapping technique is intriguing as it is not immediately clear why the code in the FFHQ latent space results in a meaningful code in the toonify space. We refer the reader to the supplementary materials for further analysis.

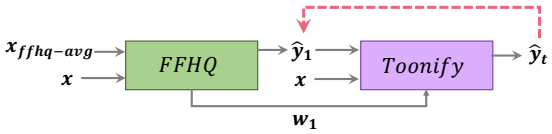

Figure 10. *Encoder bootstrapping overview.*

Input Toonify [33] ReStyle$_{pSp}$ ReStyle$^{BS}_{pSp}$

Figure 11. *Toonify comparison*. Applying ReStyle with bootstrapping, denoted ReStyle$^{BS}_{pSp}$, is able to better preserve the identity characteristics of the input real image.

Input Inverted Iterative Outputs $\longrightarrow$

Figure 12. For each input, we show the inverted image obtained after a single step of our ReStyle$_{pSp}$ FFHQ encoder followed by the iterative outputs of our ReStyle$_{pSp}$ toonify encoder.

6. Conclusions

In our work, we have focused on improving the inversion accuracy of encoders and presented a new scheme for training GAN encoders. Instead of predicting the inversion in one shot, we perform multiple forward passes, which more accurately and quickly converge to the target inversion. In a sense, this scheme allows the encoder to *learn* how to efficiently guide its convergence to the desired inversion. Moreover, the encoder is trained on a larger, richer set of images consisting not only of the original dataset itself, but also the intermediate reconstructions. We also explored pairing the ReStyle scheme with a bootstrapping technique for the image toonification task. We view this bootstrapping idea and the resulting transformations to be intriguing and may further open the door for additional tasks, leveraging the nature of our residual-based, iterative scheme.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE international conference on computer vision*, pages 4432–4441, 2019. 1, 2

[2] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8296–8305, 2020. 1, 2

[3] Rameen Abdal, Peihao Zhu, Niloy Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows, 2020. 3

[4] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Only a matter of style: Age transformation using a style-based regression model, 2021. 2

[5] Baylies. stylegan-encoder, 2019. Accessed: February 2021. 2

[6] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback, 2016. 2

[7] Lucy Chai, Jonas Wulff, and Phillip Isola. Using latent space regression to analyze and leverage compositionality in gans. In *International Conference on Learning Representations*, 2021. 2

[8] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains, 2020. 4

[9] Edo Collins, Raja Bala, Bob Price, and Sabine Susstrunk. Editing in style: Uncovering the local semantics of gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5771–5780, 2020. 1

[10] Antonia Creswell and Anil Anthony Bharath. Inverting the generator of a generative adversarial network. *IEEE transactions on neural networks and learning systems*, 30(7):1967–1974, 2018. 1, 2

[11] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019. 5

[12] Emily Denton, Ben Hutchinson, Margaret Mitchell, and Timnit Gebru. Detecting bias with generative counterfactual face attribute augmentation. *arXiv preprint arXiv:1906.06439*, 2019. 3

[13] Lore Goetschalckx, Alex Andonian, Aude Oliva, and Phillip Isola. Ganalyze: Toward visual definitions of cognitive image properties, 2019. 3

[14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2672–2680, Cambridge, MA, USA, 2014. MIT Press. 2

[15] Shanyan Guan, Ying Tai, Bingbing Ni, Feida Zhu, Feiyue Huang, and Xiaokang Yang. Collaborative learning for faster stylegan embedding. *arXiv preprint arXiv:2007.01758*, 2020. 1, 2

[16] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *arXiv preprint arXiv:2004.02546*, 2020. 1, 2, 3, 7

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 4

[18] Shao-Kang Huang, Chen-Chien Hsu, Wei-Yen Wang, and Cheng-Hung Lin. Iterative pose refinement for object pose estimation based on rgbd data. *Sensors*, 20(15):4114, 2020. 2

[19] Yuge Huang, Yuhan Wang, Ying Tai, Xiaoming Liu, Pengcheng Shen, Shaoxin Li, Jilin Li, and Feiyue Huang. Curricularface: adaptive curriculum learning loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5901–5910, 2020. 5

[20] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation, 2019. 2

[21] Ali Jahanian, Lucy Chai, and Phillip Isola. On the "steerability" of generative adversarial networks, 2020. 1

[22] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 4

[23] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data, 2020. 1

[24] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 1, 4

[25] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020. 1, 2, 4, 5, 6

[26] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 4

[27] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017. 4

[28] Zachary C Lipton and Subarna Tripathi. Precise recovery of latent vectors from generative adversarial networks. *arXiv preprint arXiv:1702.04782*, 2017. 1, 2

[29] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild, 2015. 4

[30] Yotam Nitzan, Amit Bermano, Yangyan Li, and Daniel Cohen-Or. Disentangling in latent space by harnessing a pretrained generator. *arXiv preprint arXiv:2005.07728*, 2020. 2

[31] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery, 2021. 3

[32] Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. Adversarial latent autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14104–14113, 2020. 1, 2

[33] Justin N. M. Pinkney and Doron Adler. Resolution dependent gan interpolation for controllable image synthesis between domains, 2020. 8

[34] Rakesh N Rajaram, Eshed Ohn-Bar, and Mohan M Trivedi. Refinenet: Iterative refinement for accurate object localization. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1528–1533. IEEE, 2016. 2

[35] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2, 3, 4, 5, 7

[36] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9243–9252, 2020. 1, 2, 3, 7

[37] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. *arXiv preprint arXiv:2007.06600*, 2020. 2, 3, 7

[38] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhöfer, and Christian Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. *arXiv preprint arXiv:2004.00121*, 2020. 3

[39] Ayush Tewari, Mohamed Elgharib, Mallikarjun B R., Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhöfer, and Christian Theobalt. Pie: Portrait image embedding for semantic control, 2020. 2

[40] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation, 2021. 1, 2, 3, 4, 5, 7, 8

[41] Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. *arXiv preprint arXiv:2002.03754*, 2020. 3

[42] Binxu Wang and Carlos R Ponce. A geometric analysis of deep generative image models and its applications. In *International Conference on Learning Representations*, 2021. 3

[43] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3343–3352, 2019. 2

[44] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. Gan inversion: A survey, 2021. 2

[45] Yinghao Xu, Yujun Shen, Jiapeng Zhu, Ceyuan Yang, and Bolei Zhou. Generative hierarchical features from synthesizing images. In *CVPR*, 2021. 2

[46] Ceyuan Yang, Yujun Shen, and Bolei Zhou. Semantic hierarchy emerges in deep generative representations for scene synthesis, 2020. 1

[47] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop, 2016. 4

[48] Chi Zhang, Guosheng Lin, Fayao Liu, Rui Yao, and Chunhua Shen. Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5217–5226, 2019. 2

[49] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018. 3, 5

[50] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. *arXiv preprint arXiv:2004.00049*, 2020. 1, 2, 4, 6, 8

[51] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European conference on computer vision*, pages 597–613. Springer, 2016. 2

[52] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. Improved stylegan embedding: Where are the good latents?, 2020. 8