

A Simple Baseline for StyleGAN Inversion

Tianyi Wei¹, Dongdong Chen², Wenbo Zhou¹, Jing Liao³,
Weiming Zhang¹, Lu Yuan², Gang Hua⁴, Nenghai Yu¹

¹University of Science and Technology of China ²Microsoft Cloud AI

³City University of Hong Kong ⁴Wormpex AI Research

{bestwty@mail., welbeckz@, zhangwm@, ynh@}ustc.edu.cn

{cddlyf, ganghua}@gmail.com, jingliao@cityu.edu.hk, luyuan@microsoft.com

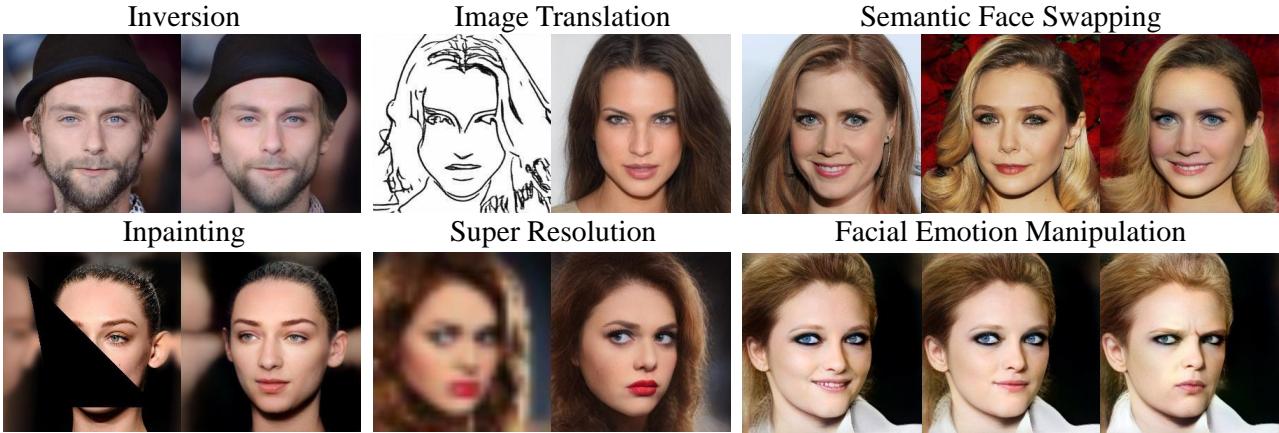


Figure 1: Our StyleGAN inversion approach can be applied to a large number of real image editing tasks.

Abstract

This paper studies the problem of StyleGAN inversion, which plays an essential role in enabling the pretrained StyleGAN to be used for real facial image editing tasks. This problem has the high demand for quality and efficiency. Existing optimization-based methods can produce high quality results, but the optimization often takes a long time. On the contrary, forward-based methods are usually faster but the quality of their results is inferior. In this paper, we present a new feed-forward network for StyleGAN inversion, with significant improvement in terms of efficiency and quality. In our inversion network, we introduce: 1) a shallower backbone with multiple efficient heads across scales; 2) multi-layer identity loss and multi-layer face parsing loss to the loss function; and 3) multi-stage refinement. Combining these designs together forms a simple and efficient baseline method which exploits all benefits of optimization-based and forward-based methods. Quantitative and qualitative results show that our method performs better than existing forward-based methods and comparably to state-of-the-art optimization-based methods, while maintaining the high efficiency as well as forward-based methods. More-

over, a number of real image editing applications demonstrate the efficacy of our method. Our project page is <https://wty-ustc.github.io/inversion>.

1. Introduction

GAN inversion aims to invert a real image back into the latent space of a pretrained GAN model, such as StyleGAN [24, 25], for the image to be faithfully reconstructed from the inverted code by the generator. It not only provides an alternative flexible image editing framework but also helps reveal the mechanism underneath deep generative models. As an emerging technique to bridge the pretrained GAN model and real image editing tasks [37, 1, 2], GAN inversion has the high demand for quality and efficiency.

Recently, numerous GAN inversion methods [1, 2, 33, 16] have been proposed and shown strong competence in performing meaningful manipulations of human faces in the latent space. They can be mainly categorized as optimization-based [1, 2, 34] and forward-based [35, 33, 16]. The optimization-based approach optimizes the latent code directly for a given single image by back-propagation. It is capable of producing high quality inversion but the op-

timization process is too time-consuming, thus greatly limiting its real-time applications. The forward-based method uses an encoder network to learn the mapping from the image space to the latent space, where only one feed-forward pass is required in the inference, providing higher efficiency for real-time applications. However, it suffers from lower reconstruction quality, and its network structure is usually large and complex. Besides, the hybrid approach [43, 8, 7] that incorporates the optimization on top of the forward network mitigates the problem of quality, but the time cost is greatly increasing.

In this paper, we propose a new feed-forward network for StyleGAN inversion, which significantly improves existing approaches in terms of both efficiency and quality. Such a baseline is quite simple but surprisingly effective.

Specifically, we introduce two core sections to the network architecture. First, our encoder network considers a *hierarchical* structure for the latent code prediction, rather than the solely last layer of the encoder for the prediction as performed in existing forward networks [43]. In this way, feature vectors extracted from various spatial levels of the encoder can correspond to different semantic levels of details from the pretrained StyleGAN generator. Meanwhile, we find it is not that the deeper the encoder, the better the inversion will be. And a *shallow* encoder is sufficient for the latent code prediction. Second, we adopt a shared *efficient prediction head* at each level, which only consists of a global average pooling layer with varied sizes to levels and a full-connected layer. It is more lightweight and efficient than the complex and independent heads used in [35] that consists of a series of convolutional layers.

Besides the architecture, we introduce two new losses into the loss function for better quality. One is *multi-layer identity loss*, which provides stronger semantic alignment supervision compared to single-layer identity loss [35] and thus greatly improves the identity consistency between the reconstructed image and the input real image. The other is *multi-layer face parsing loss*, which helps capture local facial details (*e.g.*, eyes, mouse) for a more fine-grained reconstruction.

In order to further reduce the quality gap between the single stage prediction of latent code and the ideal prediction, we propose a *multi-stage refinement* learning approach to progressively predict the residual of the latent codes through multiple passes of the encoder to achieve better inversion quality.

Qualitative and quantitative experiments show that our GAN inversion method substantially outperforms existing forward-based methods and even achieves performance comparable to the state-of-the-art optimization-based methods. A number of applications including secure deep hiding, image manipulation, image restoration, and image translation evidence the generalization of our GAN inver-

sion method for real-time image editing tasks.

2. Related Work

Generative Adversarial Networks (GANs). Since GANs were first proposed by Goodfellow *et al.* [14] in 2014, it has evolved considerably in terms of training strategies [25, 39, 18], loss functions [3, 4], regularizations [32, 9], and network structures [36, 17]. Today’s popular GANs have demonstrated amazing capabilities for image synthesis tasks. BigGAN [9] can successfully generate high-fidelity, diverse samples for complex and large-scale datasets like ImageNet [10] by only feeding the class condition. ProGAN [23] and StyleGAN [24, 25] can synthesize high-resolution images up to 1024×1024 with a progressive upsample network. Built upon these pretrained GAN models, a lot of real image editing tasks can be conducted in the latent space, such as super resolution, face attribute manipulation. To enable these tasks, GAN inversion acts as the intermediate bridge. In this paper, we follow existing methods [35, 1, 2, 16, 43] and choose StyleGAN as the inversion target, but the proposed feed-forward GAN inversion network and the accompanying design principles can be easily adapted to other GAN models.

GAN Inversion. Existing GAN inversion methods can be subdivided into optimization-based [1, 2, 15, 34], forward-based [35, 33, 6, 16] and hybrid [43, 8, 7] approaches. The optimization-based approach directly optimizes the latent code to minimize the loss. Although optimization-based methods can achieve good reconstruction results, the long optimization time limits their application. The forward-based method uses an encoder to learn the mapping from image space to the latent space. Nitzan *et al.* [33] uses two encoders to encode identity information and attribute information separately. Guan *et al.* [16] proposes a novel collaborative learning framework to train the encoder in an unsupervised manner. The pSp [35] designs a feature pyramid network with independent inversion heads for each latent code, and introduces the identity loss as an additional constraint. Different from these methods, we introduce a simple feed-forward network with improved efficiency and quality. Moreover, the newly proposed multi-layer identity loss, local parsing loss, and multi-stage refinement should also generalize to these methods.

The hybrid approach is a combination of forward-based and optimization-based methods. Zhu *et al.* [43] uses an encoder to generate an initial latent code, and the following optimizer uses it as the starting point to further refine the latent code. Our multi-stage refinement is inspired by this method, but improves the inversion quality with purely feed-forward based refinements, thus our speed is much faster than the above hybrid approach.

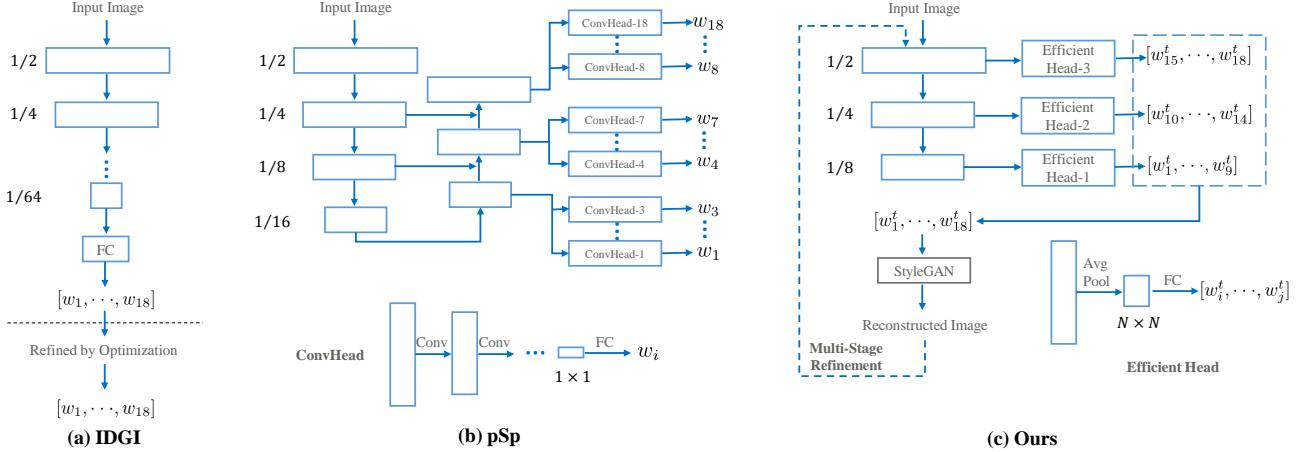


Figure 2: Network structure comparison with other methods. IDGI [43] uses a deep backbone network and pSp [35] employs 18 high-overhead ConvHeads to predict the single layer latent code separately. Compared with them, we design a shallower backbone and a more efficient prediction head. We also introduce the purely feed-forward based multi-stage refinement.

3. Proposed Method

In this section, we will first briefly introduce some representative feed-forward networks designed for GAN inversion. Then we will elaborate on the details of our improved GAN inversion network for better efficiency and quality from three aspects: network design, improved loss functions and multi-stage refinement.

3.1. Network Structure Design

In Figure 2, we show two representative network structures designed in IDGI [43] and pSp [35] for StyleGAN inversion. All these methods choose the $\mathcal{W}+$ space [1, 2, 35, 16, 43] as the target latent space to invert, which is demonstrated to be more suitable for GAN inversion than the original \mathcal{W} space. It is defined by the cascade of 18 different 512-dimensional vectors $[w_1, \dots, w_{18}], w_i \in \mathcal{W}$.

For the feed-forward network designed in IDGI, it uses a deep backbone network consisting of a series of residual blocks to encode the input image into high-level semantic feature maps (1/64 of the original input resolution) and then uses an extra fully connected (FC) layer to regress the latent vectors based on the last encoded features. Similarly, pSp also designs a deep backbone network but leverages the feature pyramid idea [29] to fuse the high-level feature with the low-level feature, then split each latent code into different feature pyramid levels. And for each latent code w_i regression, an independent convolution based head (ConvHead) is used. Depending on the input feature resolution, the ConvHead consists of different numbers of convolutional layers that progressively downsample the features into 1×1 resolution, followed by one FC layer. Therefore, there are a total of 18 ConvHeads in pSp, thus introducing a lot of overheads. For more efficient network design, we ask three questions: “*Is there any guideline for the efficient backbone design?*”, “*How to split the latent code into different feature levels?*”, and “*Can we use cheaper regression heads?*”.

Shallower Backbone. To answer the first question, we propose a simple “semantic level alignment” principle, *i.e.*, *the maximum semantic level of the backbone network should match that of the target GAN model*. If the backbone network is too shallow, its encoded feature will not be able to predict the high-level latent codes well. On the other hand, too deep backbone network will not only introduce much overhead but also possibly bring even worse inversion results. To find the best match point, we follow the recognition network design convention [38, 19] and roughly regard the features of the same downsampled resolution as the same semantic level. Guided by this principle, we conduct a simple ablation on the backbone of pSp by using a single FC head like IDGI, and find the features corresponding to 1/8 of the input resolution are sufficient to match the maximum semantic level of StyleGAN’s latent codes and can even get better inversion results than the 1/16 version. This demonstrates that *a deeper backbone network does not mean better inversion results*. Moreover, we empirically find the feature pyramid does not help, we guess it is because low semantic-level latent codes are independent of high-level semantic information. Detailed results will be given in the ablation part.

Hierarchical Structure. Given the above backbone network, we get three different semantic levels of features (*i.e.*, 1/8, 1/4, 1/2). Inspired by pSp, we then split the features from different semantic levels to predict different parts of the latent codes. Considering the feature dimension of higher semantic levels are also higher, assigning more latent codes to higher semantic-level features will have a large model. To reduce the model size while maintaining a good

inversion performance, we consider the above assignment problem as a constrained optimization problem. Denote the latent code number assigned to each semantic level to be n_1, n_2, n_3 , it can be formally formulated as:

$$\begin{aligned} n_1^*, n_2^*, n_3^* &= \arg \max_{n_1, n_2, n_3} Q(n_1, n_2, n_3) + \lambda \mathcal{P}(n_1, n_2, n_3), \\ \text{s.t., } \quad n_1 + n_2 + n_3 &= 18, \end{aligned} \quad (1)$$

where Q, \mathcal{P} denotes the inversion quality and model size function with respect to n_1, n_2, n_3 . Since it is hard to have an explicit function of Q due to its dependency on the training process, we simply adopt a simple binary search algorithm to find the rough optimal values. Specifically, we first regard n_2, n_3 as a whole and find the smallest n_1 with acceptable performance reduction, then we continue to find the smallest value for n_2 in a similar way. After the above search process, n_1, n_2, n_3 equal to 9, 5, 4 respectively.

Efficient Prediction Head. As mentioned above, there are 18 complex ConvHeads in pSp, each of which compresses the input feature into 1×1 resolution. We find such a complex ConvHead is not only unnecessary but also introduces heavy overhead and information loss. To keep the regression head lightweight while keeping important information for regression, we design a very simple and efficient head, which just consists of an average pooling layer and a fully connected layer. For the deep ($1/8$), medium($1/4$), and shallow ($1/2$) features from the hierarchical structure, they will have such a simple head respectively. Besides, to balance performance and overhead, the average pooling layer in the three heads downsamples the input features to the resolution of 7×7 , 5×5 , and 3×3 respectively.

3.2. Loss Functions

In order to train a better GAN inversion network, besides the commonly used losses, we improve the existing GAN inversion loss by introducing two new losses, *i.e.*, multi-layer identity loss and multi-layer face parsing loss.

Common Losses. The common losses consist of two types of constraints from the pixel level and the feature level, respectively. First, we use ℓ_2 loss to provide pixel-level supervision:

$$\mathcal{L}_2 = \|\mathbf{x} - G(E(\mathbf{x}))\|_2, \quad (2)$$

where \mathbf{x} represents the input image, $E(\cdot)$ represents the GAN inversion network, $G(\cdot)$ represents the target GAN network (StyleGAN), and $G(E(\mathbf{x}))$ represents the reconstructed image of the input image. However, only using the ℓ_2 loss will result in blurring reconstruction results, so we choose to use LPIPS [42] as our feature-level loss, which is demonstrated [16] to yield clearer reconstruction results compared to the perceptual loss [22].

$$\mathcal{L}_{LPIPS} = \|F(\mathbf{x}) - F(G(E(\mathbf{x})))\|_2, \quad (3)$$

where $F(\cdot)$ represents the AlexNet [26] feature extractor.

Multi-Layer Identity Loss. As the key face image attribute, keeping the original identity information is extremely important for GAN inversion. Therefore, we leverage the multi-layer features from one pre-trained face recognition network (ArcFace [11]) to impose semantic constraints on the identity information. According to the resolution size, we choose 5 different levels of features as supervision to better supervise the semantic alignment of the identity information between the reconstructed image and the input image:

$$\mathcal{L}_{m_id} = \sum_{i=1}^5 (1 - \cos(R_i(\mathbf{x}), R_i(G(E(\mathbf{x}))))), \quad (4)$$

where \cos means the cosine similarity and $R_i(\mathbf{x})$ denotes the feature corresponding to the i -th semantic level from the face recognition network R of the input image \mathbf{x} .

Multi-Layer Face Parsing Loss. Because features from the face recognition network often focus on capturing the global identity characteristics, relying on multi-layer identity loss alone cannot achieve accurate reconstructions of local details (*e.g.*, glasses). To provide better local supervision, we introduce multiple layers of features from one pre-trained facial parsing network P [31] to provide a more localized knowledge:

$$\mathcal{L}_{m_par} = \sum_{i=1}^5 (1 - \cos(P_i(\mathbf{x}), P_i(G(E(\mathbf{x}))))), \quad (5)$$

To summarize, the overall loss function is defined as:

$$\mathcal{L} = \lambda_1 \mathcal{L}_2 + \lambda_2 \mathcal{L}_{LPIPS} + \lambda_3 \mathcal{L}_{m_id} + \lambda_4 \mathcal{L}_{m_par}, \quad (6)$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are set to 1, 0.8, 0.5, 1 respectively by default.

3.3. Multi-Stage Refinement

As shown in [43], GAN inversion is a very difficult problem and often difficult to achieve satisfactory inversion results with a single pass. Therefore, they first use a feed-forward network to get an initial inversion result, then refine it by using the optimization based method as a post-processing step. Despite better inversion results, such optimization based refinement is very time-consuming. In this paper, thanks to our better inversion network design, we proposed the multi-stage refinement by purely using the feed-forward inversion networks, thus making the whole process still run in a feed-forward way. The overall framework diagram is shown in Figure 2 (c) and the whole process is formalized as follows:

$$W_t = \begin{cases} E_t(\mathbf{x}) & t = 1 \\ E_t(\mathbf{x}, G(W_{t-1})) + W_{t-1} & t > 1 \end{cases} \quad (7)$$

where E_t represents the network of stage t and $W_t = [w_1^t, \dots, w_{18}^t]$ is the inverted latent code. All the refinement stages ($t > 1$) adopt the same GAN inversion network structure as E_1 , but take the concatenation of original input image and the previous stage inverted image as input. Besides, the learning objective of the refinement stages is changed to predicting the residuals, so as to find a better latent code around the previous stage prediction result.

4. Experiments

Implementation Details. For the backbone structure, we follow the SE-ResNet50[20] backbone but only keep the stages before the 1/16 downsample stage. And we use StyleGAN2 pre-trained on the FFHQ dataset [24] as the target GAN model to inverse. In order to improve the generalization ability of the inversion network, horizontal flip is used during training. Regarding the training strategy, we first train the first-stage network alone. After its convergence, we freeze it and then train the refinement stage network in a similar way. For all the network training, the base learning rate is set to 0.0001. By default, all network is trained for 25 epochs. Following pSp [35], the Ranger optimizer is used, which is a combination of Rectified Adam [30] with the Lookahead technique [41].

Datasets and Metrics. To better evaluate the cross-dataset inversion performance of existing methods, both the baseline methods and our method use the FFHQ dataset [24] with 70,000 faces as the training set, while the qualitative and quantitative comparisons are performed on the CelebA-HQ dataset [23]. Since the optimization-based method I2S [1] is too time-consuming, we randomly selected 2,000 images from the CelebA-HQ dataset and resized them to the resolution of 256×256 for the evaluation of all methods. For quantitative evaluation, five metrics are adopted: PSNR, SSIM, IDentity Similarity (IDS), runtime (2080 Ti GPU), and model size. For SSIM, PSNR, and IDS, higher indicates better. The method we used here to calculate the identity similarity is Curricularface [21], instead of ArcFace [11] which we used for training.

4.1. Quantitative and Qualitative Comparison

We compare our method with the current state-of-the-art GAN inversion approaches, including forward-based methods: pSp [35], IDGI-Encoder [43], optimization-based method: I2S [1], and hybrid method: IDGI [43]. For IDGI, we followed the procedure in their paper: the output of the network was used as the initialization point and the optimization is iterated 100 times. I2S used the mean latent code as initialization and then iterated 1,000 times for each input image.

As shown in Table 1, our single-stage inversion network already outperforms all existing forward-based methods

| Methods | SSIM | PSNR | IDS | RunTime(s) | Param(M) |
|--------------|-------------|-------------|-------------|-------------|-----------|
| pSp | 0.58 | 20.7 | 0.57 | 0.07 | 262 |
| IDGI-Encoder | 0.51 | 18.9 | 0.19 | 0.02 | 165 |
| IDGI | 0.62 | 22.3 | 0.37 | 6.51 | 165 |
| I2S | 0.67 | 23.9 | 0.60 | 47.9 | - |
| Ours-1Stage | 0.63 | 21.3 | 0.69 | 0.04 | 85 |
| Ours-2Stage | 0.66 | 22.5 | 0.74 | 0.08 | 170 |
| Ours-3Stage | 0.67 | 23.0 | 0.75 | 0.11 | 255 |
| Ours-2Stage* | 0.63 | 21.3 | 0.69 | 0.08 | 170 |

Table 1: Quantitative comparison on the CelebA-HQ dataset, IDS means identity similarity, * means without residual learning. Our method achieves comparable results to optimization based methods and better results than feed-forward based methods, while maintaining high efficiency.

| | Size | Head | Loss | SSIM | PSNR | IDS | Param(M) |
|-----|-----------------|------|---------|------|------|------|----------|
| (a) | $\downarrow 4$ | 1E | C+SI | 0.57 | 20.3 | 0.62 | 59 |
| (b) | $\downarrow 16$ | 1E | C+SI | 0.57 | 20.5 | 0.66 | 262 |
| (c) | $\downarrow 8$ | 1E | C+SI | 0.59 | 20.9 | 0.67 | 133 |
| (d) | $\downarrow 8$ | 3E | C+SI | 0.59 | 21.0 | 0.67 | 85 |
| (e) | $\downarrow 8$ | 18C | C+SI | 0.55 | 20.0 | 0.61 | 233 |
| (f) | $\downarrow 8$ | 3E | C | 0.59 | 21.0 | 0.27 | 85 |
| (g) | $\downarrow 8$ | 3E | C+MI | 0.60 | 20.9 | 0.69 | 85 |
| (h) | $\downarrow 8$ | 3E | C+MI+MP | 0.63 | 21.3 | 0.69 | 85 |

Table 2: Quantitative ablation study. Size represents the maximum factor to downsample the input image. For the prediction head, 1E, 3E, and 18C represent the single Efficient Head, the three-level Efficient Heads, and 18 independent ConvHeads, respectively. Regarding training loss, C, SI, MI, and MP stand for common, single-layer identity, multi-layer identity, and multi-layer parsing loss, respectively.

with the smaller model size. With multi-stage refinement, our 2-stage network outperforms the hybrid method IDGI and has comparable performance with the optimization-based method I2S in terms of SSIM and PSNR metrics. Moreover, our identity consistency and speed both have obvious advantages over them. We further provide some qualitative comparison examples in Figure 3, whose quality rank is consistent with the quantitative results.

4.2. Ablation Study

Effectiveness of Network Structure Design. In the top part of Table 3, we first validate the effectiveness of the proposed network structure design. (a), (b), and (c) indicate that we used the single efficient prediction head to directly predict the latent code after downsampling the input image to 1/4, 1/16, and 1/8 of the original resolution, respectively. It clearly shows that deeper networks are not better. For the inversion task, there is a point where the semantic level of StyleGAN’s latent code is best matched, which may explain the unsatisfactory performance of the IDGI encoder.

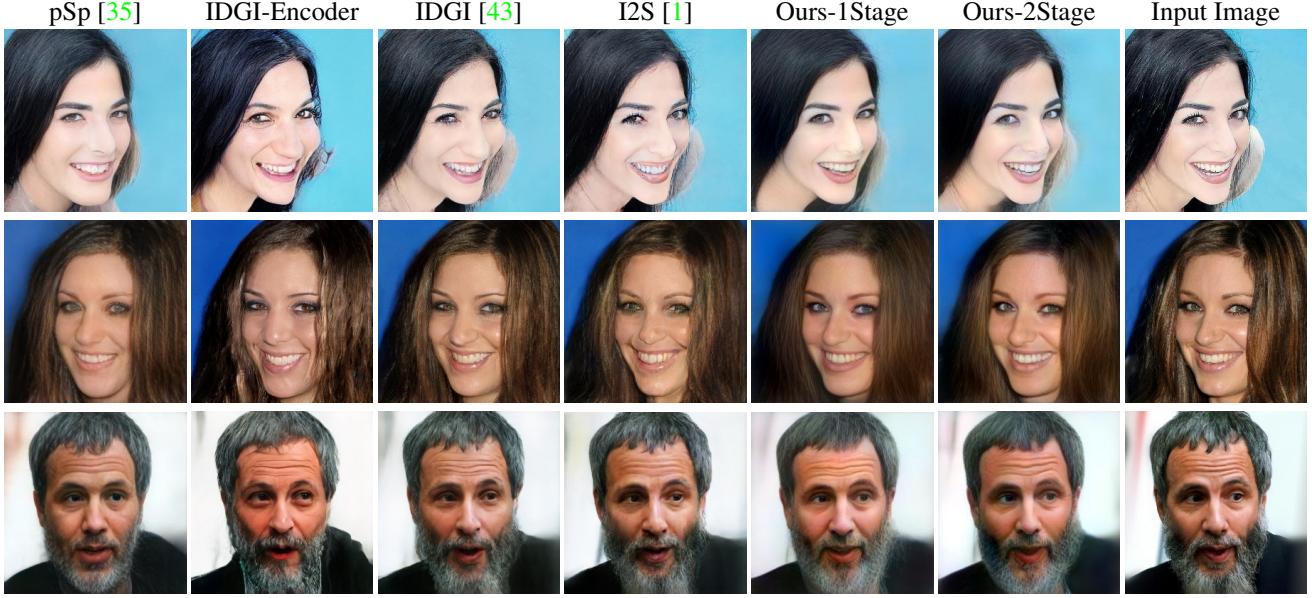


Figure 3: Visual comparison of the GAN inversion quality on the CelebA-HQ dataset.

In order to achieve a more efficient model without degrading the performance, we adopted a three-layer hierarchical structure to predict the latent code as (d). For (e), we took the same scheme as pSp [35], *i.e.*, we used 18 Convheads to predict the latent code of each layer. Compared with (d), (e) brings a larger number of parameters but worse performance. We guess this may be because continuous downsampling to the feature resolution of 1×1 will lead to excessive information missing, while using average pooling can effectively aggregate useful information with a shorter path.

Importance of Multi-Layer Identity and Parsing Losses. The bottom part of Table 3 shows our quantitative results for removing identity loss, introducing multi-layer identity loss, and introducing both multi-layer identity loss and multi-layer face parsing loss on top of (d), respectively. It well demonstrates the effectiveness of multi-layer identity and face parsing loss. We further provide some qualitative visual comparisons in Figure 4 and Figure 5. With the multi-layer identity loss, the identity consistency between the reconstructed image and the input image is significantly improved. With the multi-layer face parsing loss, the network is able to pay attention to the local face regions, thus more accurately reconstructing the local details, *e.g.*, the glasses and tongue of the two cases.

Significance of the Refinement Stage. As shown in the bottom part of Table 1, our 2-stage inversion network significantly improves all the metrics compared to the 1-stage counterpart, but the gain starts to saturate by adding more stages. Considering both speed and performance, we adopt the two-stage inversion network as the default setting. In

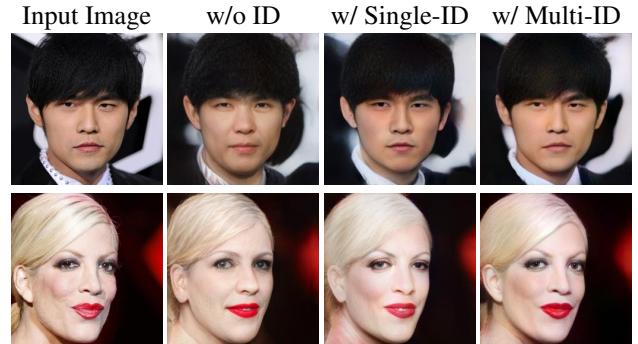


Figure 4: The effect of multi-layer identity loss.

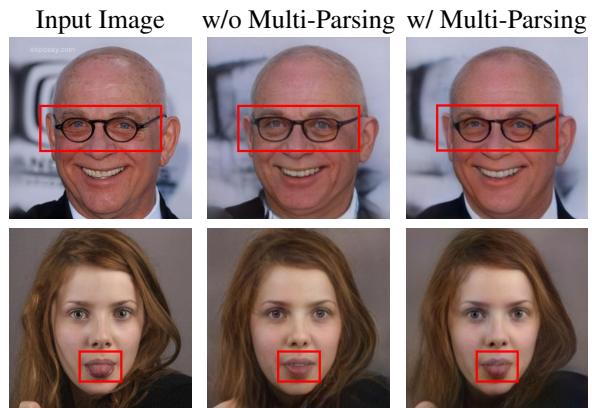


Figure 5: The effect of multi-layer face parsing loss.

the last row of Table 1, we further change the second stage learning objective to directly predict the latent code instead of the residuals of the first stage latent code. The experimental results show that learning the residuals in the re-

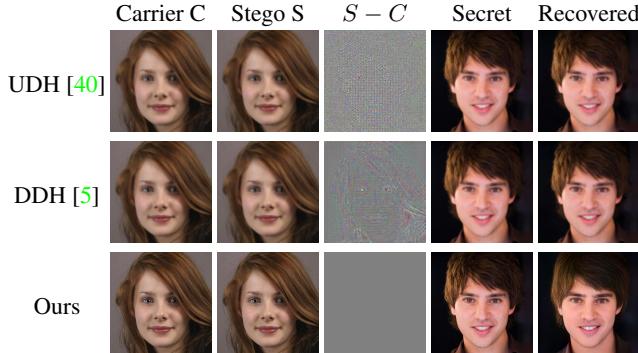


Figure 6: Qualitative comparison of deep hiding. Please zoom in to get a better view of the residuals between the stego image S and the original carrier image C .

finement stages is extremely important. In contrast, if we continue to learn the absolution latent code, the refinement stage is still difficult to bring better performance.

5. Applications

In this section, we show some interesting and practical applications to demonstrate the potentials of our method. More results can be found in the supplementary materials.

5.1. Secure Deep Hiding

In this application, the goal is to hide one secret image into one carrier image in an imperceptible way. Existing deep hiding schemes [5, 40] either directly concatenate two images together and feed them into the network, or feed only the secret image into the network for the carrier-agnostic purpose, which have weak imperceptibility and undetectability, therefore the security cannot be guaranteed. Empowered by our excellent inversion quality, we can combine the proposed method with traditional steganography to propose a novel application: secure deep hiding, which overcomes the aforementioned drawbacks.

Specifically, the whole hiding process is divided into four steps: 1) The sender uses the proposed method to obtain the latent code W_s of the secret image; 2) The sender hides W_s into the carrier image using the traditional steganography method [13] to get the stego image, and then sends the stego image out; 3) After receiving the stego image, the recipient recovers W_s accurately using the corresponding extraction algorithm [13]; 4) The recipient feeds W_s to the same pre-trained StyleGAN for recovering the secret image. In this process, the pre-trained StyleGAN indeed acts as the encryption and decryption codebook.

Figure 12 shows the visual comparison of our method with universal deep hiding (UDH) [40] and dependent deep hiding (DDH) [5]. With the benefit of the proposed high-quality GAN inversion method, our deep hiding scheme achieves comparable reconstruction quality with UDH and DDH. But as shown in the third column of Figure 12,

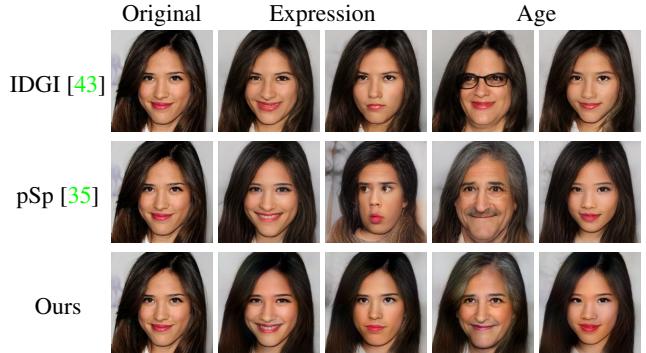


Figure 7: Comparison of our approach with IDGI [43] and pSp [35] on semantic face manipulation.

our scheme has better imperceptibility and undetectability. To summarize, by combining the proposed GAN inversion method with traditional steganography, we can easily implement the application of securely hiding an image with 1024×1024 resolution to another image.

5.2. Image Manipulation

Semantic Face Manipulation. To accomplish semantic face manipulation, given a real face image, we first invert the latent code by using the proposed method, then we follow the InterFaceGAN [37] to find the specific semantic direction in the latent space and modify the latent code along that direction. As shown in Figure 13, compared to IDGI [43] and pSp [35], our results are more desirable. For example, when manipulating the expression, our method significantly better preserves the identity consistency of the image. In terms of age manipulation, our method manipulates other facial semantics less while changing the age of the target face.

Semantic Face Swapping. Semantic face swapping is intended to replace the identity of the target image with that of the source image, however it does not guarantee strict alignment of facial attributes such as: expression, lighting, head pose, etc. With the pretrained StyleGAN2 as a strong face reconstruction prior, we find our method can also be used to create more natural face swapping results easily. Specifically, we crop out the central area of the source face image and paste it directly into the same position of the target face image to create a pasted image. To eliminate the obvious artefacts present in the pasted image, we use the proposed GAN inversion technique to reconstruct the pasted image. As shown in Figure 14, the reconstructed image perfectly blends the pasted face with its surrounding area. Compared to traditional geometric transformation based FaceSwap [12] and learning-based FaceShifter [28], our results have higher identity consistency with the source image and better image quality.

Face Interpolation. Given two real-world images A and B,

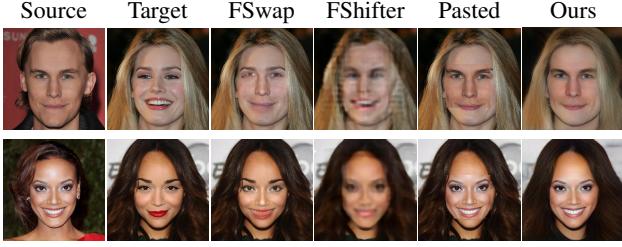


Figure 8: Comparison of our approach with FSswap [12] and FShifter [28] on semantic face swapping.



Figure 9: Face interpolation results.



Figure 10: Style mixing results.

we use the proposed GAN inversion method to obtain their respective latent codes $W_A, W_B \in \mathcal{W}^+$. Then, we combine the two latent codes by linear weighting to generate the intermediate latent code $W_I = \lambda W_B + (1 - \lambda)W_A$. Finally, the image corresponding to the intermediate latent code is generated. By gradually increasing the blending parameter λ from 0 to 1, we can achieve the gradual morphing effect from input A to input B, as shown in Figure 15.

Style Mixing. Style mixing aims to transfer the low-level appearance characteristics from the style image to the content image. In detail, after obtaining the respective latent codes of the style image and the content image, we replace the last 11 layers of the latent codes corresponding to the content image with those of the style image. A specific example is shown in Figure 16, where all the content images are coated with the same color lipstick as the style image.

5.3. Image Restoration

By introducing corresponding data augmentations to the input images during the training phase, our GAN inversion method can be easily extended to perform a number of image restoration tasks, such as: colorization, inpainting, super resolution. Here we qualitatively compare our method with pSp [35] on these tasks, both of which use the same data augmentations. Figure 17 shows the visual comparison of our method with pSp on these three tasks.

Colorization. For colorization task, we use the color image as the target and the corresponding grayscale image

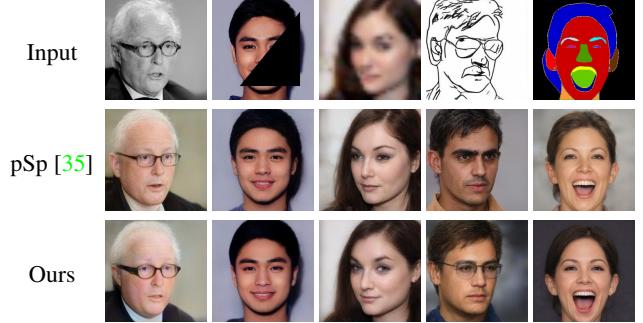


Figure 11: Comparison of our approach with pSp [35] on image restoration and image translation tasks.

as input when training the network. Compared with pSp [35], our approach accomplishes reasonable coloring for the grayscale image while ensuring consistency in identity and local details (*e.g.*, glasses).

Inpainting. To perform inpainting using the GAN inversion framework, we employ a degraded transformation to the input image, in which a region is randomly selected and the pixel value of that region is set to 0 during the training process. As illustrated in Figure 17, our method achieves reasonable filling of unknown regions while keeping the known pixel values unchanged as much as possible.

Super Resolution. Following the data augmentation method of pSp, we randomly downsample the input image by $\times 1, \times 2, \times 4, \times 8$, and $\times 16$ and use the original resolution image as the target during training. Compared to pSp, our results are more realistic and with better facial details, *e.g.*, the eyes in the shown example are more faithfully aligned with the low-resolution image.

5.4. Image Translation

By replacing the input with the corresponding sketch or segmentation label map of the image during the network training phase, our framework can also perform image translation tasks. Given that there is no relevant dataset for the FFHQ, we perform this task on the CelebA-HQ, of which 3,000 images are reserved for testing purposes. Specifically, we follow [35] to generate the sketch dataset of CelebA-HQ, while the segmentation label maps are from the CelebAMask-HQ dataset [27]. The visual comparison results are shown in Figure 17. Compared to pSp [35], it is clear that our results are more faithfully aligned to the respective semantics of the input, *e.g.* the glasses in the sketch-to-image example.

6. Conclusion

In this paper, we propose a new simple feed-forward GAN inversion network, with significant improvements in terms of efficiency and quality. Such improvements come

from three aspects: 1) designing a more efficient GAN inversion network with shallow backbone, hierarchical latent code regression, and efficient prediction heads; 2) introducing multi-layer identity loss and multi-layer parsing loss; and 3) purely feed-forward based multi-stage refinement. Extensive evaluation and applications demonstrate that our method performs much better than existing feed-forward based methods and comparably to state-of-the-art optimization based methods with higher efficiency.

References

- [1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4431–4440, 2019. [1](#), [2](#), [3](#), [5](#), [6](#)
- [2] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8293–8302, 2020. [1](#), [2](#), [3](#)
- [3] Abdul Fatir Ansari, J. Scarlett, and Harold Soh. A characteristic function approach to deep implicit generative modeling. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7476–7484, 2020. [2](#)
- [4] Martín Arjovsky, Soumith Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *ICML*, 2017. [2](#)
- [5] S. Baluja. Hiding images in plain sight: Deep steganography. In *NIPS*, 2017. [7](#), [12](#)
- [6] C. Bartz, Joseph Bethge, Haojin Yang, and C. Meinel. One model to reconstruct them all: A novel way to use the stochastic noise in stylegan. *ArXiv*, abs/2010.11113, 2020. [2](#)
- [7] David Bau, Hendrik Strobelt, W. Peebles, J. Wulff, B. Zhou, Jun-Yan Zhu, and A. Torralba. Semantic photo manipulation with a generative image prior. *ACM Transactions on Graphics (TOG)*, 38:1 – 11, 2019. [2](#)
- [8] David Bau, Jun-Yan Zhu, J. Wulff, W. Peebles, Hendrik Strobelt, B. Zhou, and A. Torralba. Seeing what a gan cannot generate. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4501–4510, 2019. [2](#)
- [9] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. *ArXiv*, abs/1809.11096, 2019. [2](#)
- [10] Jia Deng, W. Dong, R. Socher, L. Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. [2](#)
- [11] Jiankang Deng, J. Guo, and S. Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4685–4694, 2019. [4](#), [5](#)
- [12] FaceSwap. <https://github.com/wuhukai/FaceSwap>. Accessed: Feb. 2021. [Online]. [7](#), [8](#), [13](#)
- [13] T. Filler, Jan Judas, and J. Fridrich. Minimizing additive distortion in steganography using syndrome-trellis codes. *IEEE Transactions on Information Forensics and Security*, 6:920–935, 2011. [7](#)
- [14] Ian J. Goodfellow, Jean Pouget-Abadie, M. Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014. [2](#)
- [15] Jinjin Gu, Yujun Shen, and B. Zhou. Image processing using multi-code gan prior. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3009–3018, 2020. [2](#)
- [16] Shanyan Guan, Ying Tai, B. Ni, Feida Zhu, Feiyue Huang, and Xiaokang Yang. Collaborative learning for faster stylegan embedding. *ArXiv*, abs/2007.01758, 2020. [1](#), [2](#), [3](#), [4](#)
- [17] Ishaan Gulrajani, F. Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. In *NIPS*, 2017. [2](#)
- [18] Tianyu Guo, C. Xu, Jiajun Huang, Yunhe Wang, Boxin Shi, Chao Xu, and Dacheng Tao. On positive-unlabeled classification in gan. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8382–8390, 2020. [2](#)
- [19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. [3](#)
- [20] Jie Hu, L. Shen, Samuel Albanie, G. Sun, and Enhua Wu. Squeeze-and-excitation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:2011–2023, 2020. [5](#)
- [21] Y. Huang, Yuhang Wang, Ying Tai, Xiaoming Liu, Pengcheng Shen, Shao xin Li, Jilin Li, and Feiyue Huang. Curricularface: Adaptive curriculum learning loss for deep face recognition. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5900–5909, 2020. [5](#)
- [22] J. Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. [4](#)
- [23] Tero Karras, Timo Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *ArXiv*, abs/1710.10196, 2018. [2](#), [5](#)
- [24] Tero Karras, S. Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019. [1](#), [2](#), [5](#)
- [25] Tero Karras, S. Laine, Miika Aittala, Janne Hellsten, J. Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8107–8116, 2020. [1](#), [2](#)
- [26] A. Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84 – 90, 2012. [4](#)
- [27] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. [8](#)
- [28] Lingzhi Li, Jianmin Bao, Hao Yang, Dong Chen, and Fang Wen. Faceshifter: Towards high fidelity and occlusion aware face swapping. *ArXiv*, abs/1912.13457, 2019. [7](#), [8](#), [13](#)

- [29] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 3
- [30] Liyuan Liu, Haoming Jiang, Pengcheng He, W. Chen, Xiaodong Liu, Jianfeng Gao, and J. Han. On the variance of the adaptive learning rate and beyond. *ArXiv*, abs/1908.03265, 2020. 5
- [31] Ziwei Liu. https://github.com/switchablenorms/CelebAMask-HQ/tree/master/face_parsing. Accessed: Mar. 2021. [Online]. 4
- [32] Takeru Miyato, T. Kataoka, Masanori Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *ArXiv*, abs/1802.05957, 2018. 2
- [33] Yotam Nitzan, A. Bermano, Yangyan Li, and D. Cohen-Or. Face identity disentanglement via latent space mapping. *ACM Transactions on Graphics (TOG)*, 39:1 – 14, 2020. 1, 2
- [34] Xingang Pan, Xiaohang Zhan, Bo Dai, D. Lin, Chen Change Loy, and Ping Luo. Exploiting deep generative prior for versatile image restoration and manipulation. *ArXiv*, abs/2003.13659, 2020. 1, 2
- [35] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and D. Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. *ArXiv*, abs/2008.00951, 2020. 1, 2, 3, 5, 6, 7, 8, 11, 13, 15
- [36] Edgar Schönfeld, B. Schiele, and A. Khoreva. A u-net based discriminator for generative adversarial networks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8204–8213, 2020. 2
- [37] Yujun Shen, Jinjin Gu, X. Tang, and B. Zhou. Interpreting the latent space of gans for semantic face editing. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9240–9249, 2020. 1, 7
- [38] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 3
- [39] Song Tao and J. Wang. Alleviation of gradient exploding in gans: Fake can be real. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1188–1197, 2020. 2
- [40] C. Zhang, Philipp Benz, Adil Karjauv, Geng Sun, and In-So Kweon. Udh: Universal deep hiding for steganography, watermarking, and light field messaging. In *NeurIPS*, 2020. 7, 12
- [41] Michael Ruogu Zhang, James Lucas, Geoffrey E. Hinton, and Jimmy Ba. Lookahead optimizer: k steps forward, 1 step back. In *NeurIPS*, 2019. 5
- [42] Richard Zhang, Phillip Isola, Alexei A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018. 4
- [43] Jiapeng Zhu, Yujun Shen, De li Zhao, and B. Zhou. In-domain gan inversion for real image editing. In *ECCV*, 2020. 2, 3, 4, 5, 6, 7, 13

Appendix

A. Video Inversion

In this section, we demonstrate the potential of the proposed method for video inversion tasks. In <https://youtu.be/gJwFgdRHK0M>, we compare our method with pSp [35] on two continuous real-world videos, each of which is 10 seconds in length and 300 frames in total. Obviously, our results have better identity consistency and motion coherence.

B. Ablation of FPN

In Table 3, we give the quantitative ablation study about the feature pyramid network (FPN), which serves as a supplement to the ablation experiment in the main paper. Compared with baseline (d), we add the same feature pyramid structure as pSp [35] to (d), however, quantitative results prove that this structure does not lead to performance improvement. We guess it is because low semantic-level latent codes are independent of high-level semantic information.

| Method | SSIM | PSNR | IDS | RunTime(s) | Param(M) |
|---------|------|------|------|------------|----------|
| (d) | 0.59 | 21.0 | 0.67 | 0.04 | 85 |
| (d)+FPN | 0.59 | 21.0 | 0.67 | 0.05 | 98 |

Table 3: Quantitative ablation study of FPN.

C. More Application Results

In Figure 12, Figure 13, Figure 14, Figure 15, Figure 16, and Figure 17, we provide more application results. The high-quality results demonstrate the great potential of our approach for many real-world image editing tasks.

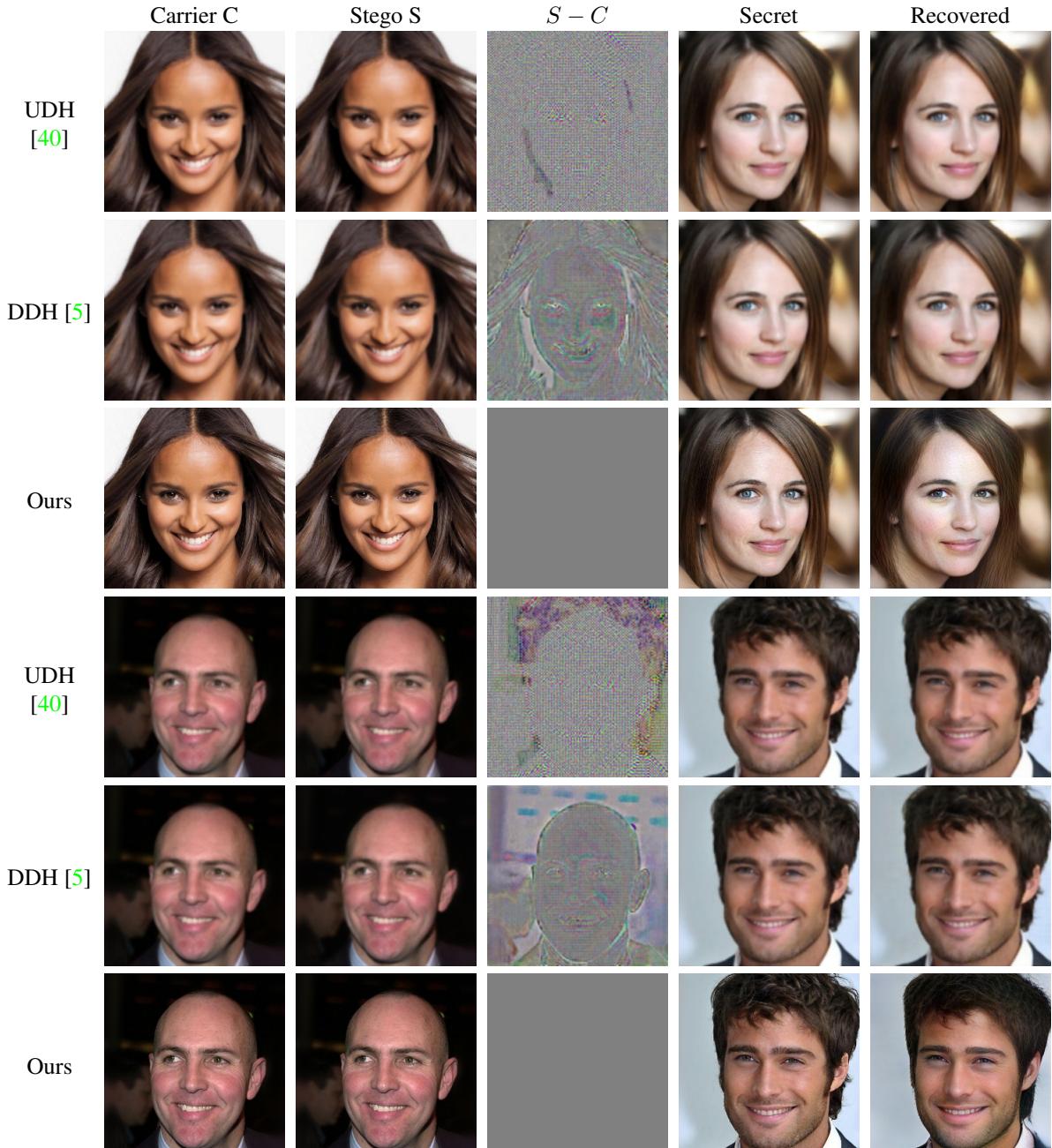


Figure 12: Qualitative comparison of deep hiding. Stego is obtained by hiding Secret into the Carrier. Please zoom in to get a better view of the residuals between the stego image S and the original carrier image C .

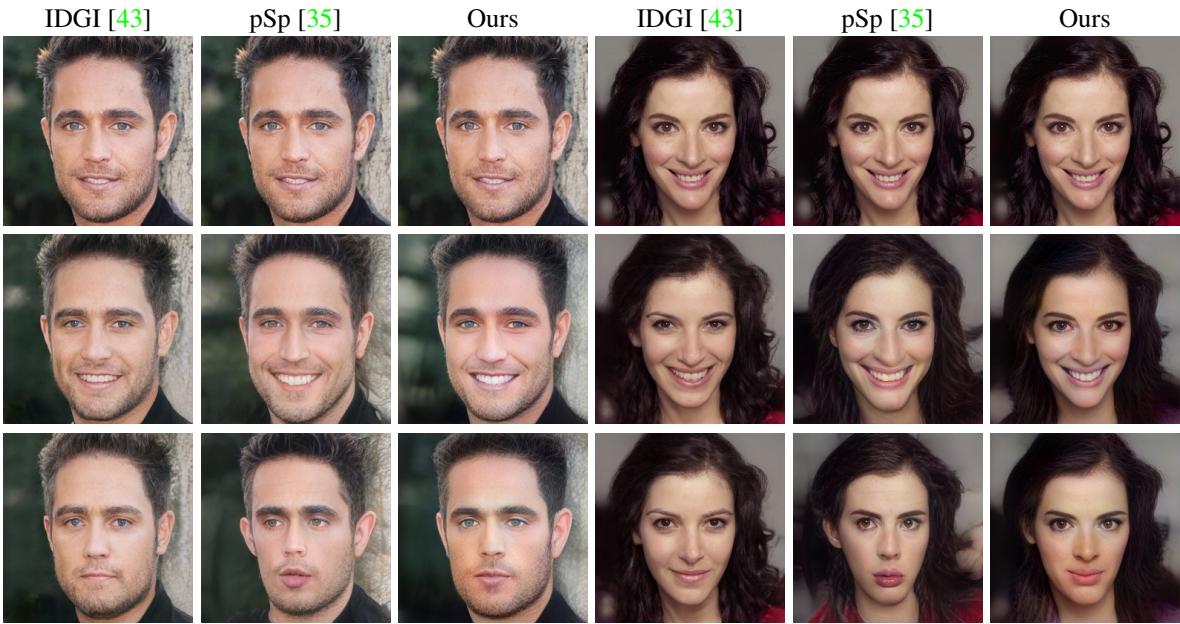


Figure 13: Comparison of our approach with IDGI [43] and pSp [35] on semantic expression manipulation. The first row is the original input and the rest are the manipulated results of each method.

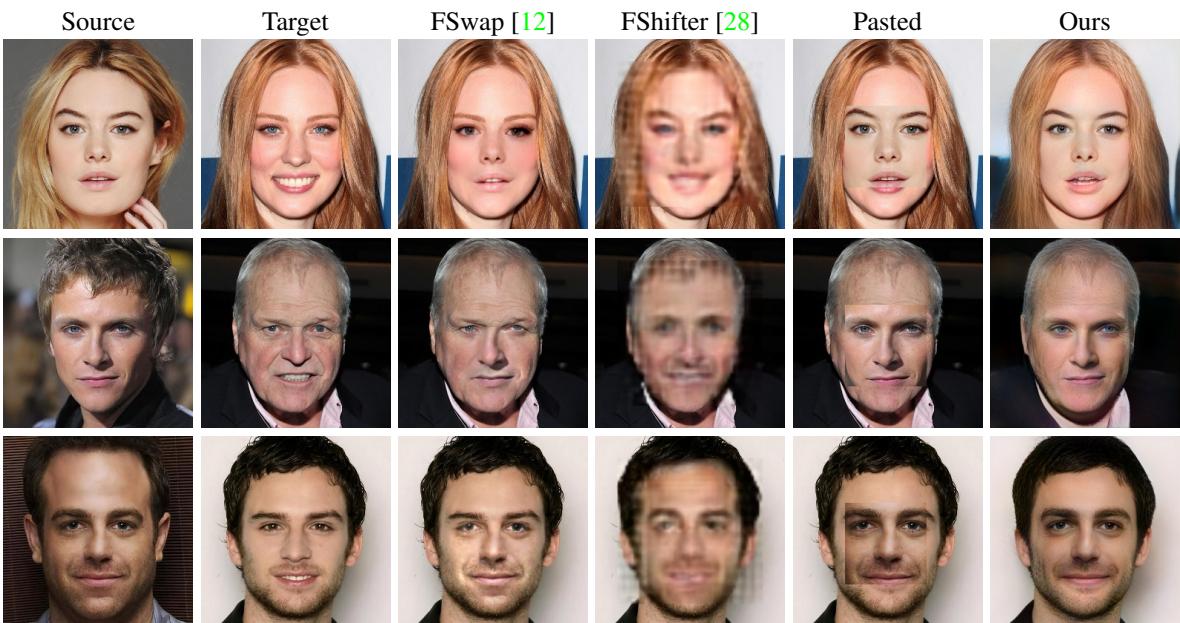


Figure 14: Comparison of our approach with FSwap [12] and FShifter [28] on semantic face swapping.

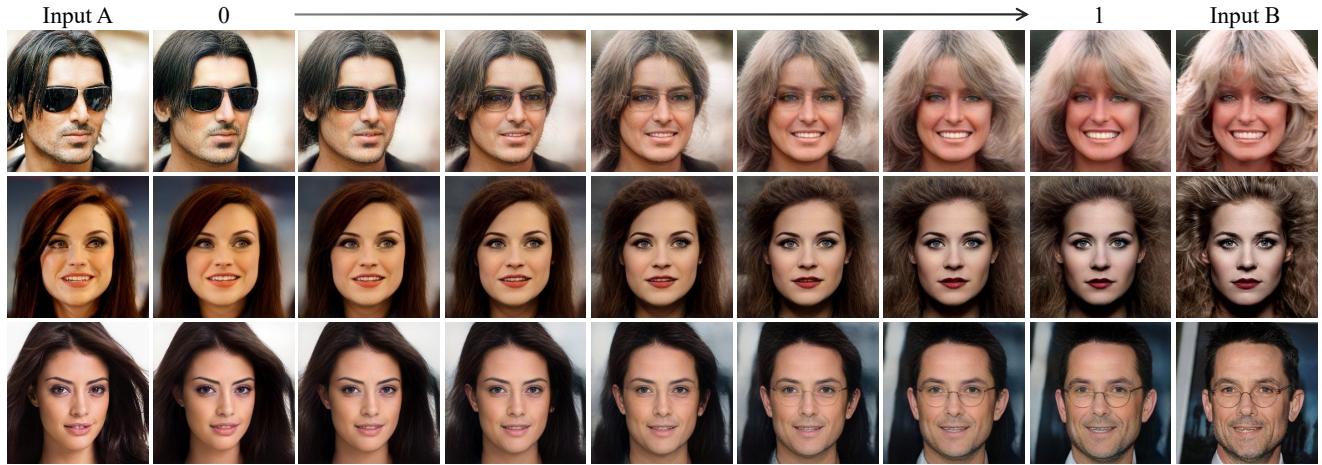


Figure 15: Face interpolation results.

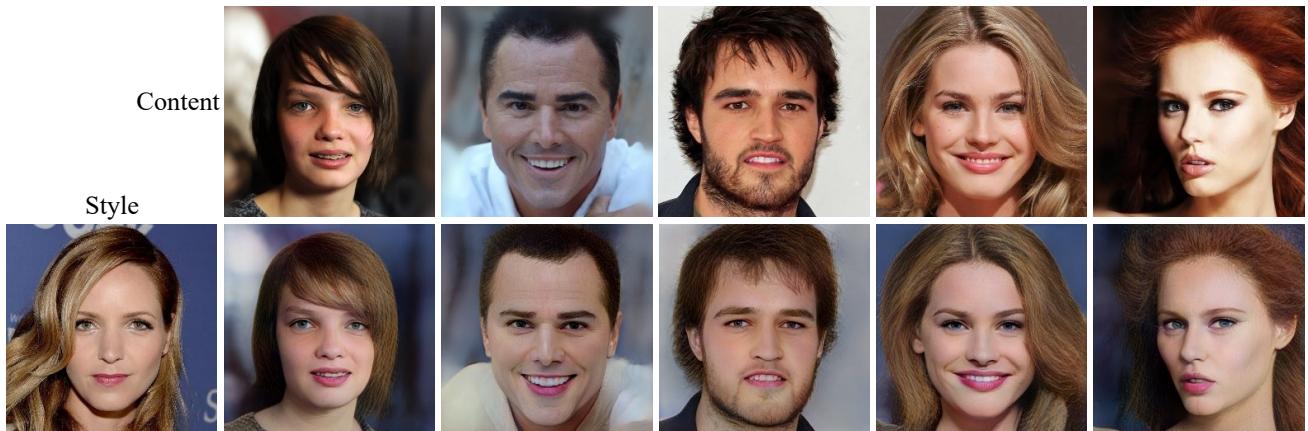


Figure 16: Style mixing results.

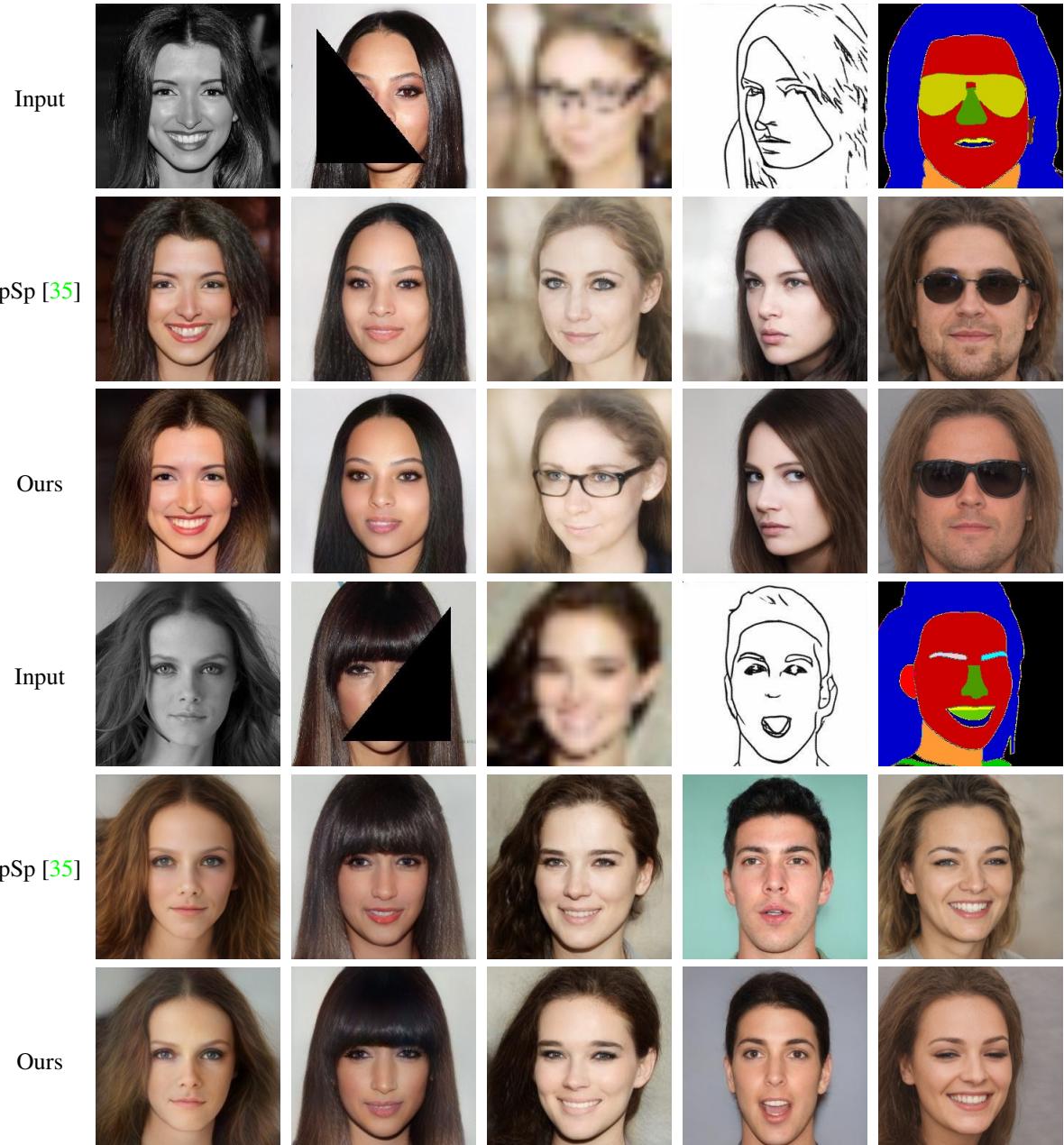


Figure 17: Comparison of our approach with pSp [35] on image restoration and image translation tasks.