

Designing an Encoder for StyleGAN Image Manipulation

Omer Tov

Tel-Aviv University

Yuval Alaluf

Tel-Aviv University

Yotam Nitzan

Tel-Aviv University

Or Patashnik

Tel-Aviv University

Daniel Cohen-Or

Tel-Aviv University

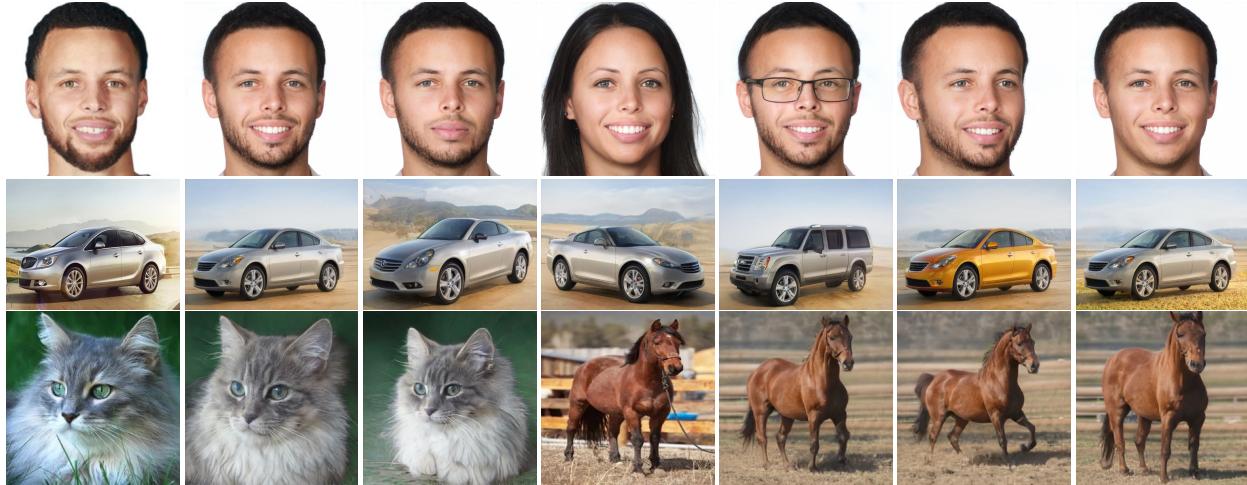


Figure 1: Real image editing via StyleGAN inversion using our e4e method. For each domain we show from left to right: the original real image, the inverted image, and multiple manipulations performed using various editing techniques.

Abstract

Recently, there has been a surge of diverse methods for performing image editing by employing pre-trained unconditional generators. Applying these methods on real images, however, remains a challenge, as it necessarily requires the inversion of the images into their latent space. To successfully invert a real image, one needs to find a latent code that reconstructs the input image accurately, and more importantly, allows for its meaningful manipulation. In this paper, we carefully study the latent space of StyleGAN, the state-of-the-art unconditional generator. We identify and analyze the existence of a distortion-editability tradeoff and a distortion-perception tradeoff within the StyleGAN latent space. We then suggest two principles for designing encoders in a manner that allows one to control the proximity of the inversions to regions that StyleGAN was originally trained on. We present an encoder based on our two principles that is specifically designed for facilitating editing on real images by balancing these tradeoffs. By evaluating its performance qualitatively and quantitatively on numerous challenging domains, including cars and horses, we show that our inversion method, followed by common editing techniques, achieves superior real-image editing quality, with only a small reconstruction accuracy drop.

1. Introduction

In recent years, Generative Adversarial Networks (GANs) [12] have made significant progress in unconditional image synthesis. In particular, StyleGAN [19, 20, 21] has achieved unprecedented visual quality and fidelity in various domains. Beyond its phenomenal realism, StyleGAN uses a learnt intermediate latent space, \mathcal{W} , which more faithfully reflects the distribution of the training data compared to the standard Gaussian latent space. Furthermore, numerous works have demonstrated that \mathcal{W} has intriguing disentangled properties [7, 14, 34, 37, 40], which allow one to perform extensive image manipulations by leveraging a pretrained StyleGAN.

To apply such manipulations on real images, one must first *invert* the given image into the latent space, i.e. retrieve a latent code, a so called style code, $w \in \mathcal{W}$, such that feeding the obtained style code as input to the pretrained StyleGAN returns the original image. Hence, a high-quality inversion scheme is vital for such editing techniques. High-quality inversion is characterized by two aspects. First, the generator should properly reconstruct the given image with the style code obtained from the inversion. Second, it should be possible to best leverage the editing capabilities of the latent space to obtain meaningful and realistic edits of the

given image. In short, we call the latter aspect *editability*. To define a proper reconstruction, we distinguish between two properties: (i) distortion and (ii) perceptual quality [5]. As will be elaborately explained in the paper, *distortion* measures per-image input-output similarity, whereas *perceptual quality* measures how realistic the reconstructed image is. Note that ideally the distortion should be low and the perceptual quality high.

There is an intimate relation between distortion, perceptual quality, and editability. The expressiveness of the \mathcal{W} latent space has been shown to be limited [1, 31], in that not every image can be accurately mapped into \mathcal{W} . To alleviate this limitation, Abdal *et al.* [1] demonstrate that any image can be inverted into an extension of \mathcal{W} , denoted \mathcal{W}^+ , where a style code consists of a number of style vectors. The space \mathcal{W}^+ has more degrees of freedom, and is thus significantly more expressive than \mathcal{W} . Although this extension is expressive enough to represent real images, as we shall show, inverting images away from the original \mathcal{W} space reaches regions of the latent space that are less editable and in which the perceptual quality is lower. This tradeoff between distortion, perceptual quality, and editability is presented and extensively analyzed in the paper.

Recognizing that the main motivation for inverting an image is the downstream editing task, we focus on understanding the editability of inverted real images. Our key insight is that editability and perceptual quality are best achieved by inverting an image *close* to \mathcal{W} . In the following, the term “close” will be characterized by two key properties. First, low *variance* between the different style vectors. Second, each style vector should lie within the distribution \mathcal{W} .

We design a new encoder, which is explicitly encouraged to invert images close to \mathcal{W} . This encoder serves two purposes: (i) it allows demonstrating that the distortion-editability and distortion-perception tradeoffs are controlled by the proximity of a latent code to \mathcal{W} , and (ii) it constitutes an effective encoder for editing real images. Thus, we name our encoder *e4e* — *Encoder for Editing*.

Specifically, we design the encoder to map a given real image to a style code that consists of a series of style vectors with low variance, each close to the distribution of \mathcal{W} . Since the distribution of \mathcal{W} cannot be explicitly modeled, we extend the adversarial training of the style code introduced in Nitzan *et al.* [27], and apply it to multiple codes to encourage the proper encoding of each into \mathcal{W} . To further assist in staying in editable regions, we additionally present a progressive training scheme where the variance between the style vectors is gradually increased during training.

We present quantitative and qualitative results that demonstrate the distortion-editability and distortion-perception tradeoffs, and the benefit of inverting “close” to \mathcal{W} . We evaluate our encoder, showing the generalization of our approach and its applicability for a variety of challenging do-

mains which, unlike the facial domain, have no common structure and may contain numerous modes. In Figure 1, we show inversions obtained by our encoder in multiple domains, followed by several manipulations performed using various editing methods [3, 14, 34, 35]. As can be seen, with only a slight degradation in distortion, we are able to achieve plausibly edited images, while preserving the content and quality of the original images.

To summarize, we present four main contributions:

- We analyze the complex latent space of StyleGAN and suggest a novel view of its structure.
- We present the innate tradeoffs among distortion, perception, and editability.
- We characterize the tradeoffs and design two means for an encoder to control them.
- We present e4e, a novel encoder that is specifically designed to allow for the subsequent editing of inverted real images.

Source code and pretrained models can be found at our project page: <https://github.com/omertov/encoder4editing>.

2. Background and Related Work

2.1. Latent Space Manipulation

Recently, understanding and controlling the latent representation of pretrained GANs has attracted considerable attention. Notably, it has been shown that StyleGAN [19, 20, 21], with its novel style-based architecture, contains a semantically rich latent space that can be used to perform diverse image manipulations. Motivated by this, many works have presented diverse approaches for discovering and performing semantically meaningful manipulations on images. A commonly-used approach involves finding “walking” directions that control a specific attribute of interest. Early works [10, 11, 34] use a fully-supervised approach and find latent directions for binary-labeled attributes such as young \leftrightarrow old or smile \leftrightarrow no-smile. Others have proposed self-supervised approaches that find latent space directions corresponding to a specific image transformation, such as zoom or rotation [17, 29, 36]. Finally, several methods [14, 38, 39] find latent directions in an unsupervised manner and require manual annotations to determine the semantic meaning of each direction post hoc.

Extending the aforementioned works, there are other techniques that go beyond walking along linear directions. Tewari *et al.* [37] use a pretrained 3DMM to edit expression, pose and illumination of faces by borrowing them from other latent codes. Abdal *et al.* [3] infer a modified latent code using an auxiliary pretrained face attributes classifier. Shen

	Individual style codes are limited to \mathcal{W}	Same style code in all layers
\mathcal{W}	✓	✓
\mathcal{W}^k	✓	
\mathcal{W}_*		✓
\mathcal{W}_*^k		

Table 1: A concise summary outlining the differences between various latent spaces. Note that, due to its ambiguity, \mathcal{W}^+ is omitted from the table and is instead defined more explicitly using \mathcal{W}^k and \mathcal{W}_*^k .

et al. [35] perform eigenvector decomposition of the weights of the generator’s first layer to find edit directions. Collins *et al.* [7] perform local semantic editing by borrowing a subset of the latent code from other samples.

Some recent works have also considered other latent spaces. Sendik *et al.* [33] suggest a modified style-based architecture that learns multiple input constants and achieves better disentanglement between data modes. Last, [24, 40] have considered the so called *Style Space*, and are able to identify specific components that have a clear effect on spatial regions or on a single attribute. By modifying these components, they are able to achieve disentangled editing.

2.2. Latent Space Embedding

To perform such manipulations on real images, one must first obtain the latent code from which the pretrained GAN can most accurately reconstruct the original input image. This task has been commonly referred to as *GAN Inversion* [31, 42, 45]. Generally, inversion methods either (i) directly optimize the latent vector to minimize the error for the given image [1, 2, 8, 21, 23], (ii) train an encoder to map the given image to the latent space [13, 28, 31], or (iii) use a hybrid approach combining both [4, 45]. Typically, methods performing optimization are superior in achieving low distortion. However, they require a substantially longer time to invert an image and are less editable.

Given the high dimensionality of the latent space, finding meaningful directions is extremely challenging. Therefore, recent methods propose an end-to-end approach for performing latent manipulations using a well-trained generator. Specifically, Nitzan *et al.* [27] train an encoder to obtain a latent vector representing the identity of one image and the pose, expression, and illumination of another. Menon *et al.* [25] solve single-image super resolution by taking a low-resolution image and searching the latent space for a high-resolution version of the image using direct optimization. Finally, Richardson *et al.* [31] perform image-to-image translation by directly encoding input images into the latent codes representing the desired transformation.

In contrast to previous works, our encoder is specifically designed to output latent codes that ensure further editing

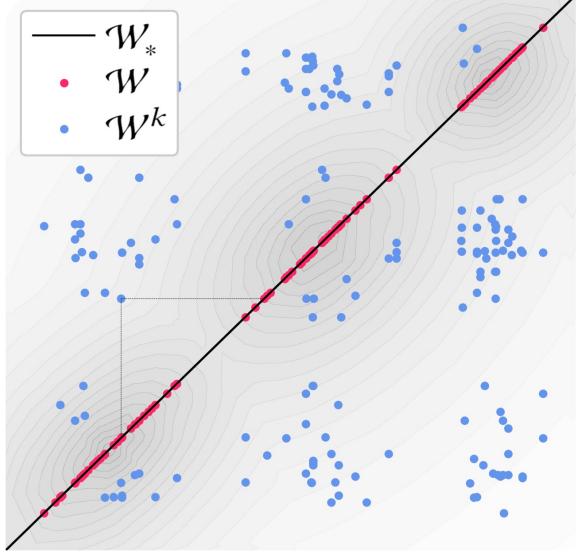


Figure 2: An illustration of our definitions for a 1-dimensional \mathcal{W} and for $k = 2$ (i.e., \mathcal{W}^2). The diagonal line represents \mathcal{W}_* since the two coordinates of each point on it are equal. The pink points are sampled from \mathcal{W} , which here is represented as a mixture of 1-dimensional Gaussians. The entire gray region represents \mathcal{W}_*^2 , corresponding to \mathbb{R}^2 . The blue points are sampled from \mathcal{W}^2 since each coordinate of such a point is sampled from \mathcal{W} .

capabilities. As we shall see, there is an explicit trade-off between distortion and editability where one can obtain better editability by allowing a small degradation in distortion.

3. Terminology

StyleGAN [19, 20, 21] consists of two key components: (i) a mapping function that maps a latent code $z \in \mathcal{Z} = \mathcal{N}(\mu, \sigma^2)$ into a *style code* $w \in \mathcal{W} \subsetneq \mathbb{R}^{512}$, and (ii) a generator that takes in the style code, replicated several times (according to the desired resolution), and generates an image. Note that while the distribution \mathcal{Z} is known to be Gaussian, there is no known explicit model for the distribution of \mathcal{W} [41]. In the following, we will refer to this distribution as the *range* of the mapping function.

It has been shown [1] that not every real, in-domain image can be inverted into StyleGAN’s latent space. To alleviate this limitation, one can increase the expressiveness of the StyleGAN generator by inputting k *different* style codes instead of a single vector. We denote this extended space by $\mathcal{W}^k \subsetneq \mathbb{R}^{k \times 512}$ where k is the number of style inputs of the generator. For example, a generator capable of synthesizing images at a resolution of 1024×1024 operates in the extended \mathcal{W}^{18} space corresponding to the 18 different style inputs.

Even more expressive power can be achieved by inputting style codes which are not necessarily from the true distribution of \mathcal{W} , i.e. outside the range of StyleGAN’s mapping function. Observe that this extension can be applied by taking a single style code and replicating it, or by taking k different style codes. We denote these extensions by \mathcal{W}_* and \mathcal{W}_*^k , respectively. Note, that here, we depart from the commonly used \mathcal{W}^+ notation due to its ambiguity with various works referring to it as both \mathcal{W}^k and \mathcal{W}_*^k . However, note, that for simplicity and convenience, we refer to \mathcal{W} both as the 512-dimensional distribution, and as a subset of \mathcal{W}^k where all the k style codes are equal and in \mathcal{W} . In Table 1 we present a summary describing the differences between the latent spaces. We provide an illustration of the differences between \mathcal{W} and \mathcal{W}^k in Figure 2.

4. The GAN Inversion tradeoffs

4.1. Preliminaries

As discussed in Section 2, most of the research on StyleGAN’s latent space revolves around two separate tasks: *GAN inversion* and *latent space manipulation*. Previous works have considered the two tasks as follows. First, in the task of inversion, given an image x , we infer a latent code w , which is used to reconstruct x as accurately as possible when forwarded through the generator G . In the task of latent space manipulation, for a given latent code w , we infer a new latent code, w' , such that the synthesized image $G(w')$ portrays a semantically meaningful edit of $G(w)$.

Inspecting the GAN inversion task, we stress that reconstruction alone has limited purpose since reconstructing an inverted image can at best return the original image. As previous works have mentioned [1, 3, 45], the central motivation for inversion is to allow for further latent editing operations. That is, a successful reconstruction should enable extensive and diverse manipulation of *real* images with greater ease. This objective, however, is not trivial as some latent codes are more editable than others.

Following the above observations, we suggest a broader perspective on how to evaluate GAN inversion methods. Specifically, they should be evaluated based on *both* reconstruction and editability.

The term reconstruction is ill-defined as an evaluation metric. Blau and Michaeli [5] have distinguished between two separate properties of reconstruction — *distortion* and *perceptual quality*. Formally, distortion is defined as $\mathbb{E}_{x \sim p_X} [\Delta(x, G(w))]$ where p_X is the distribution of the real images, and $\Delta(x, G(w))$ is an image-space difference measure between images x and $G(w)$. Perceptual quality measures how realistic the reconstructed images are, with no relation to any reference image.

Most previous works have considered only distortion [1, 27, 31] for evaluating the reconstruction. Distortion

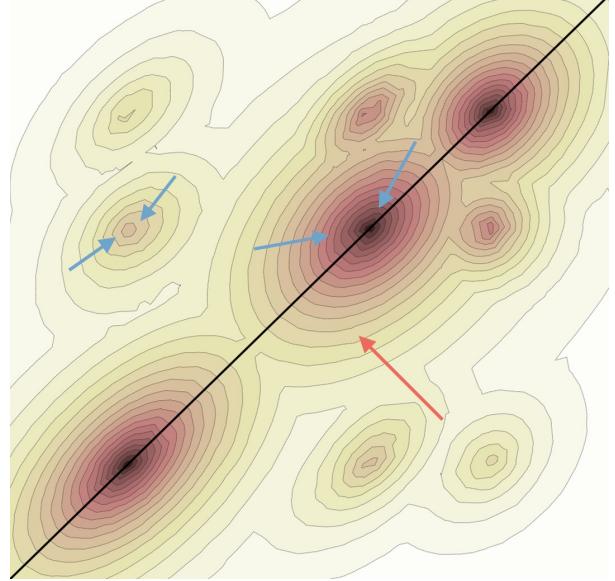


Figure 3: As in Figure 2, \mathcal{W} is 1-dimensional and $k = 2$, that is $\mathcal{W}^k = \mathcal{W}^2$. The main diagonal represents \mathcal{W}_* , and warmer colors correspond to higher densities of \mathcal{W}^k . As one traverses the space in parallel to the red arrow and approaches the diagonal, the latent codes become closer to \mathcal{W}_* since the variation between the coordinates of the latent code is decreased. Likewise, traversing along a blue arrow results in latent codes that get closer to \mathcal{W}^k since each coordinate, independently, gets closer to the distribution \mathcal{W} .

alone, however, does not capture the quality of the reconstruction. In fact, Blau and Michaeli [5] proved that not only is the perceptual quality different than distortion, there exists an explicit tradeoff between the two. Therefore, both the distortion and the perceptual quality of the reconstructed images must be evaluated to provide a complete evaluation of a reconstruction method or, in our case, a GAN Inversion method.

For editability, it is expected that given the inverted latent code, one can find many latent space directions corresponding to disentangled semantic edits in the image-space. Moreover, it is important to maintain a high *perceptual quality* of the edited images. In addition to the above, in Section 7 we present a novel measure that is specifically designed to evaluate the success of an encoder for the editability of real images.

4.2. Distortion-Editability & Distortion-Perception Tradeoffs

We now turn to study the distortion, perceptual quality, and editability of different regions in StyleGAN’s latent space. As discussed thoroughly in Section 3, \mathcal{W}_* differs



Figure 4: The editability gap between \mathcal{W} and \mathcal{W}_*^k . The top row depicts several edits performed on an image generated from a latent code sampled from \mathcal{W} . In the bottom row, the same image is inverted back into \mathcal{W}_*^k and passed through the generator to obtain a visually-similar image. The same edits as those performed in the top row are then applied on the inverted \mathcal{W}_*^k code. Note that although the two source images are similar in the image-space, the edits in \mathcal{W} are visually better than the corresponding ones in \mathcal{W}_*^k , as can be observed by the warped shapes of the cars in the bottom row.

from \mathcal{W} in two ways. First, \mathcal{W}_*^k may contain different style codes at different style-modulation layers. Second, each style code, independently, is not bound to the true distribution of \mathcal{W} , but can instead take any value from \mathbb{R}^{512} .

It is well known that \mathcal{W}_*^k achieves lower, i.e. better, distortion than \mathcal{W} [1, 4, 31, 45]. Additionally, we find that \mathcal{W} is more editable, see Figure 4. This provides the first evidence of the inherent tradeoff between distortion and editability. Observe that since StyleGAN is originally trained in the \mathcal{W} space, it is not surprising that \mathcal{W} is more well-behaved and has better perceptual quality compared to its \mathcal{W}_*^k counterpart. On the other hand, observe that due the significantly higher dimensionality of \mathcal{W}_*^k and the architecture of StyleGAN, \mathcal{W}_*^k has far greater expressive power.

We claim that the distortion-editability and the distortion-perception tradeoffs exist not only between \mathcal{W} and \mathcal{W}_*^k , but also within the \mathcal{W}_*^k space itself. Moreover, the tradeoffs are controlled by the proximity to \mathcal{W} . More specifically, as we approach \mathcal{W} , the distortion worsens while the editability and perceptual quality improve, see Figures 4 and 5. To validate this claim, it is necessary to develop a means for allowing one to control the proximity of an encoded image to \mathcal{W} . In Section 5, we introduce two principles and mechanisms for controlling this proximity. Then, in Section 8.2, we perform extensive experimentation to support our claims.

Note that the concept of the distortion-editability tradeoff is not new. Previous works [46], and [45] also observed the fact that different regions in the latent space have different editing properties and presented inversion methods that explicitly addressed this. However, here we identify the editable regions more explicitly by differentiating between the different extensions of \mathcal{W} .



Figure 5: An example of the distortion-perception tradeoff. Inverting the source image using two different encoders yields significantly different results. While the middle image achieves low distortion (notice the overall similarity in shape and posture) and low perceptual quality (notice the warped head), the rightmost image achieves higher distortion but better perceptual quality.

5. Designing an encoder

Building upon the observations from the previous section, we now present principles for designing an encoder and novel training scheme to explicitly address the proximity to \mathcal{W} . Note that we make the important distinction between the two main inversion methodologies: (i) latent code optimization, and (ii) encoder-based methods. In this work, we concentrate on encoder-based methods for several key reasons. First, they are significantly faster as they infer a latent code with a single forward pass. This is in contrast to the costly, per-image optimization methodology. Second, due to the fact that CNNs are piece-wise smooth, the output of an encoder lies in a tight space which is more suitable for editing. Conversely, an optimization-based inversion may converge to an arbitrary point in the latent space. In the following, we consider a generic encoder which infers latent codes in the space of \mathcal{W}_*^k .

We now present the two principles for controlling the proximity to \mathcal{W} , where each is defined using a dedicated training paradigm to encourage the encoder to map into regions in \mathcal{W}_*^k that lie close to \mathcal{W} (see Figure 3). We find these principles to be most effective when applied jointly.

5.1. Minimize Variation

The first approach for getting closer to \mathcal{W} is to encourage the inferred \mathcal{W}_*^k latent codes to lie closer to \mathcal{W}_* , i.e. minimize the variance between the different style codes, or equivalently, encourage the style codes to be identical. To this end, we propose a novel “progressive” training scheme.

Let $E(x) = (w_0, w_1, \dots, w_{N-1})$ denote the output of the encoder, where N is the number of style-modulation layers. Common encoders are trained directly into \mathcal{W}_*^k , i.e., learn each w_i separately and simultaneously. Conversely, we infer a *single* latent code, w , and a set of *offsets* from w . More formally, we learn an output of the form $E(x) = (w, w + \Delta_1, \dots, w + \Delta_{N-1})$.

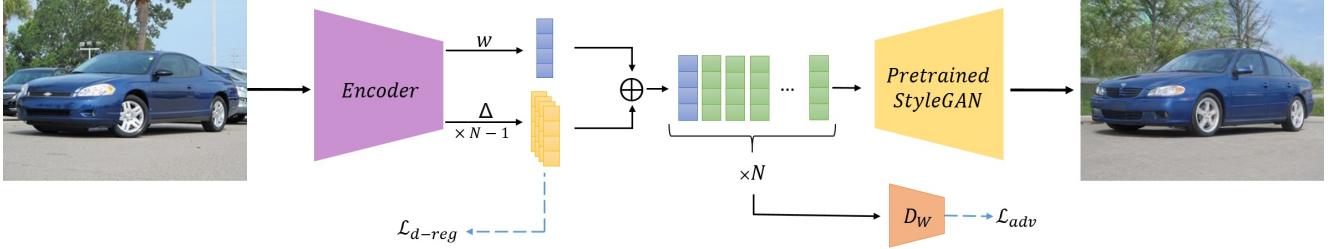


Figure 6: Our e4e network architecture. The encoder receives an input image and outputs a single style code w together with a set of offsets $\Delta_1.. \Delta_{N-1}$, where N denotes the number of StyleGAN’s style modulation layers. We obtain our final latent representation by replicating the w vector N times and adding each Δ_i to its corresponding entry. During training, the $\mathcal{L}_{d\text{-reg}}$ regularization encourages small variance between different entries of the final representation, thereby remaining close to \mathcal{W}_* . \mathcal{L}_{adv} guides each latent code towards the range of StyleGAN’s mapping network, resulting in a final representation closer to \mathcal{W}^k . As a result of applying both regularization terms, the encoder’s final learned representation lies close to \mathcal{W} .

Specifically, at the start of training we set $\forall i : \Delta_i = 0$ and the encoder is trained to infer a single \mathcal{W}_* code. We then gradually allow the network to learn a *different* Δ_i for each i sequentially. We find that doing so is useful in allowing the encoder to gradually expand from \mathcal{W}_* towards \mathcal{W}^k . This scheme, together with the semantic meaning of specific input layers of StyleGAN, allows one to first learn a coarse reconstruction of the input image and then gradually add finer details while still remaining close to \mathcal{W}_* .

We note that low frequency details greatly control the distortion quality. Thus, our progressive training scheme first focuses on improving the low frequency distortion by tuning the coarse-level offsets. Then, the encoder gradually complements these offsets with higher frequency details introduced by the finer-level offsets.

In terms of Figure 3, this process is equivalent to starting from a style code that lies on the main diagonal, and then allowing small perturbations, thereby slightly diverging from it.

In order to explicitly enforce a proximity to \mathcal{W}_* , we add an L_2 delta-regularization loss:

$$\mathcal{L}_{\text{d-reg}}(w) = \sum_{i=1}^{N-1} \|\Delta_i\|_2. \quad (1)$$

5.2. Minimize Deviation From \mathcal{W}^k

The second approach for getting closer to \mathcal{W} is to encourage the \mathcal{W}_* latent codes obtained by the encoder to lie closer to \mathcal{W}^k . That is, encourage the individual style codes to lie within the actual distribution of \mathcal{W} . To do so, we adopt a latent discriminator [27] which is trained in an adversarial manner to discriminate between real samples from the \mathcal{W} space (generated by StyleGAN’s mapping function) and the encoder’s learned latent codes.

Observe that such a latent discriminator addresses the challenge of learning to infer latent codes that belong to a distribution which cannot be explicitly modeled. In doing

so, the discriminator encourages the encoder to infer latent codes that lie within \mathcal{W} as opposed to \mathcal{W}_* . We use a single latent discriminator, denoted $D_{\mathcal{W}}$, that operates on each latent code entry separately. In every iteration, we calculate the GAN loss for every $E(x)_i$ and average over all i -s.

We use the non-saturating GAN loss [12] with R_1 regularization [26],

$$\begin{aligned} \mathcal{L}_{\text{adv}}^D &= -\mathbb{E}_{w \sim \mathcal{W}} [\log D_{\mathcal{W}}(w)] - \mathbb{E}_{x \sim p_X} [\log(1 - D_{\mathcal{W}}(E(x)_i))] + \\ &\quad \frac{\gamma}{2} \mathbb{E}_{w \sim \mathcal{W}} \left[\|\nabla_w D_{\mathcal{W}}(w)\|_2^2 \right], \end{aligned} \quad (2)$$

$$\mathcal{L}_{\text{adv}}^E = -\mathbb{E}_{x \sim p_X} [\log D_{\mathcal{W}}(E(x)_i)]. \quad (3)$$

6. e4e: Encoder for Editing

We now turn to implement our encoder and novel training scheme. A high-level description of our encoder is illustrated in Figure 6. Our encoder builds upon the Pixel2Style2Pixel (pSp) encoder [31], but here, we design the encoder specifically for editing. Therefore, we name our new encoder *e4e* — “Encoder for Editing”.

Unlike the original pSp encoder which generates N style-codes in parallel, here, we follow the principles described in Section 5.1 and generate a single base style code, denoted by w , and a series of $N - 1$ offset vectors (illustrated in yellow in Figure 6). The offsets are then summed up with the base style code w to yield the final N style codes which are then fed into a fixed, pretrained StyleGAN2 generator to obtain the reconstructed image.

6.1. Losses

To train our encoder, we employ losses that ensure low distortion, as commonly done when training an encoder, and losses that explicitly encourage the generated style codes to remain close to \mathcal{W} , thereby increasing the perceptual quality and editability of the generated images.

Distortion One of the key ideas presented in pSp is the identity loss, which is specifically designed to assist in the accurate inversion of real images in the facial domain. Motivated by recent strong self-supervised learning techniques, we generalize this identity loss and introduce a novel \mathcal{L}_{sim} loss defined by,

$$\mathcal{L}_{\text{sim}}(x) = 1 - \langle C(x), C(G(e4e(x))) \rangle, \quad (4)$$

where C is a ResNet-50 [15] network trained with MO-Cov2 [6] and G is the pretrained StyleGAN2 generator. Here, \mathcal{L}_{sim} explicitly encourages the encoder to minimize the cosine similarity between the feature embeddings of the reconstructed image and its source image. Observe that \mathcal{L}_{sim} can be applied in any *arbitrary* domain due to the general nature of the extracted features. Note that in the facial domain, we adopt the original identity loss used in pSp, and employ a pretrained ArcFace [9] facial recognition network for extracting the feature embeddings.

In addition, we employ the commonly used \mathcal{L}_2 and $\mathcal{L}_{\text{LPIPS}}$ [44] losses to learn both pixel-wise and perceptual similarities. That is, our distortion loss is defined as:

$$\mathcal{L}_{\text{dist}}(x) = \lambda_{l2}\mathcal{L}_2(x) + \lambda_{lpips}\mathcal{L}_{\text{LPIPS}}(x) + \lambda_{sim}\mathcal{L}_{\text{sim}}(x). \quad (5)$$

Perceptual quality and editability To increase the perceptual quality and editability, we employ the two losses introduced in Section 5. First, we apply a delta-regularization loss to ensure proximity to \mathcal{W}_* when learning the offsets Δ_i . Second, we use an adversarial loss using our latent discriminator, which encourages each learned style code to lie within the distribution \mathcal{W} . More formally, the editability loss is given by

$$\mathcal{L}_{\text{edit}}(x) = \lambda_{d-reg}\mathcal{L}_{d-reg}(x) + \lambda_{adv}\mathcal{L}_{adv}(x), \quad (6)$$

where \mathcal{L}_{d-reg} and \mathcal{L}_{adv} are defined in Section 5.1 and Section 5.2.

Total loss Our overall loss objective is defined as a weighted combination of the distortion and editability losses:

$$\mathcal{L}(x) = \mathcal{L}_{\text{dist}}(x) + \lambda_{edit}\mathcal{L}_{\text{edit}}(x). \quad (7)$$

7. Evaluation

Evaluating our approach is particularly challenging. Our goal is to evaluate a tradeoff between distortion and two qualities — perceptual quality and editability. Each alone is difficult to evaluate as they are perceptual in essence, and hard to objectively measure numerically. Perceptual quality is commonly quantitatively evaluated by measuring the discrepancy between the real and generated *distributions* using algorithms, such as FID [16], SWD [30] or IS [32].

However, these methods do not always agree with human judgement, which is their true goal. Further, they are also affected by distortion, which makes them less suitable for evaluating the tradeoffs. As an example, let us consider a face reconstruction algorithm that perfectly reconstructs a given face, except that it adds to all men a realistic-looking beard. This is only a distortion problem, as the images all remain realistic to the human eye. However, the discrepancy between the distributions, which is measured only as a *proxy of realism* will be affected. Editability is even more difficult to evaluate because the quality of an edited image should be evaluated as a function of the magnitude of the edit change. Additionally, qualitative measures are often susceptible to be biased.

To evaluate the results in a rigorous and fair manner, we provide numerous visual examples in the next section and in the supplementary materials to provide a large scale gallery for the reader’s impression. It should be noted, that we take special care to avoid subjective bias in selecting the presented images. We do so by selecting the images according to their given order in their relevant test sets (avoiding the suspicion of cherry picking). We additionally conduct a user study to assess human opinion on the visual results. Last, we complete the evaluation by including all popular quantitative metrics. However, as we shall demonstrate below, such metrics often not only contradict the human opinion, but each other as well.

It should be emphasized that in the following examples, we present various editing results executed by different techniques. However, we do not evaluate their performance. Instead, we evaluate only the distortion-perception and distortion-editability tradeoffs which exist in all techniques. Another important point to stress is that results presented by editing methods are often demonstrated on synthetic images generated by StyleGAN, with only a few editing examples on real images. Here, however, since the focus of the paper is the inversion of real images, *all* results are necessarily on real images.

To properly evaluate our proposed method, one needs to evaluate the performance of distortion, perceptual quality, and editability. Following the definitions introduced in Section 4.1, we now elaborate the evaluation protocols used.

Distortion We provide numerous qualitative results for the reader’s impression. To quantify these results, we apply the common metrics of L_2 and LPIPS [44] on pairs of input and reconstructed images.

Perceptual quality In addition to the large gallery shown, we provide the results of a user study to evaluate human subjective opinion. We quantitatively evaluate the results by additionally measuring the FID [16] and SWD [30] between the distributions of the real and reconstructed images.

Editability We define editability as the ability to perform latent-space editing using any arbitrary technique while maintaining high-visual quality of the image obtained using the edited latent code. To this end, we follow our inversion method with several existing editing techniques: StyleFlow [3], InterFaceGAN [34], GANSpace [14], and SeFa [35]. After performing inversion, we apply these techniques to edit the code in semantic manners such as pose, gender, and age for the human facial domain. We then generate images from the edited code and evaluate the perceptual quality of the generated images. We again provide numerous visual samples for the reader’s impression and perform a user study. To further quantify these results, we also adopt the FID and SWD measures to compare the distributions of the original and the edited images. Note that FID and SWD measure perceptual quality for both reconstruction and editability. The difference lies with the distributions used to measure them.

Latent Editing Consistency Here we present a new evaluation measure, which we call *latent editing consistency* (*LEC*), that combines two key components of GAN inversion methods meant for latent space editing. One captures the extent for which the inversion matches the true inverse of the generator, and the second captures how well-behaved the edits of the inversion outputs are. The protocol of this measure is illustrated in Figure 7. We visually study the difference between the input and output images and quantitatively define the distance in the latent space by

$$LEC(f_\theta) = \mathbb{E}_x \|E(x) - f_\theta^{-1}(E(G(f_\theta(E(x)))))\|_2, \quad (8)$$

where E is an encoder and $f_\theta(w)$ is an invertible semantic latent editing function parameterized by θ . For example, if the editing is performed by traversing along a linear direction v in the latent space, then $f_\theta(w) = w + \alpha \cdot v$ and θ is composed of a scalar α and latent direction v . Note that, in the optimal case, where E is the perfect inverse of G , then $LEC = 0$. In practice, however, the encoder is imperfect and thus $f_\theta(E(x))$ and $E(G(f_\theta(E(x))))$ are not exactly equal. By performing the inverse editing, LEC captures the extent for which the inherent inversion errors translate to errors in the subsequent editing. A well-behaved encoder suitable for latent editing should yield a small LEC difference in the latent space.

Note that edits with f_θ translate to visual results in the image-space after applying G . Therefore, there exists a corresponding function F that performs the same edits in the image-space (see Figure 7). If F was tractable to compute, one could evaluate how far E is from editing equivariance:

$$\mathbb{E}_x \|f_\theta(E(x)) - E(F(x))\|_2. \quad (9)$$

Since F is usually intractable, however, LEC can be considered as a proxy for equivariance.

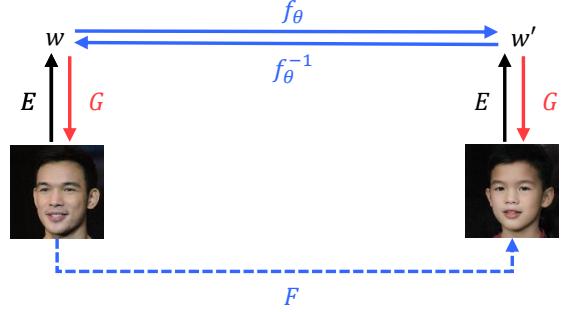


Figure 7: An Illustration of the Latent Editing Consistency (*LEC*) protocol. The color of the arrows correspond to their purpose, where black \leftrightarrow invert, blue \leftrightarrow edit and red \leftrightarrow generate. We begin by inverting a real image, on the left, into the latent space of a pre-trained generator. Then, the latent code is edited in some semantic manner by f_θ . Using the obtained edited code, an image is synthesized using the generator. This resulting image, on the right, is then inverted back into the latent space. Finally, the obtained latent code is edited using the inverse of the original edit, f_θ^{-1} , and an image is synthesized. Ideally, the first and last images, as well as the latent codes, should be equal.

	d-reg	$D_{\mathcal{W}}$
A		
B	✓	
C		✓
D	✓	✓

Table 2: Our encoder and training configurations. Recall, the delta-regularization (d-reg) refers to the variation minimization described in 5.1. D_w refers to the latent discriminator as explained in 5.2.

8. Experiments

8.1. Settings

Configurations Throughout this section, we explore various configurations of our e4e method. We concisely summarize these configurations in Table 2. Note that the two editing losses listed in the table are used in conjunction with the distortion loss $\mathcal{L}_{\text{dist}}$ defined in Equation 5.

Datasets To illustrate the generalization of our approach, we perform extensive experimentation on a diverse set of challenging domains. For the facial domain, we use the FFHQ [20] dataset for training and the CelebA-HQ [18] test dataset for evaluation. We use the Stanford Cars [22] dataset for training and evaluation on the cars domain. We additionally provide results on the LSUN [43] Horse, Cats, and Churches datasets. The diversity of the LSUN domains is significantly greater than that of human faces and cars. We therefore find them to be more challenging. Note that all results are obtained using official StyleGAN2 generators trained on images from that domain.

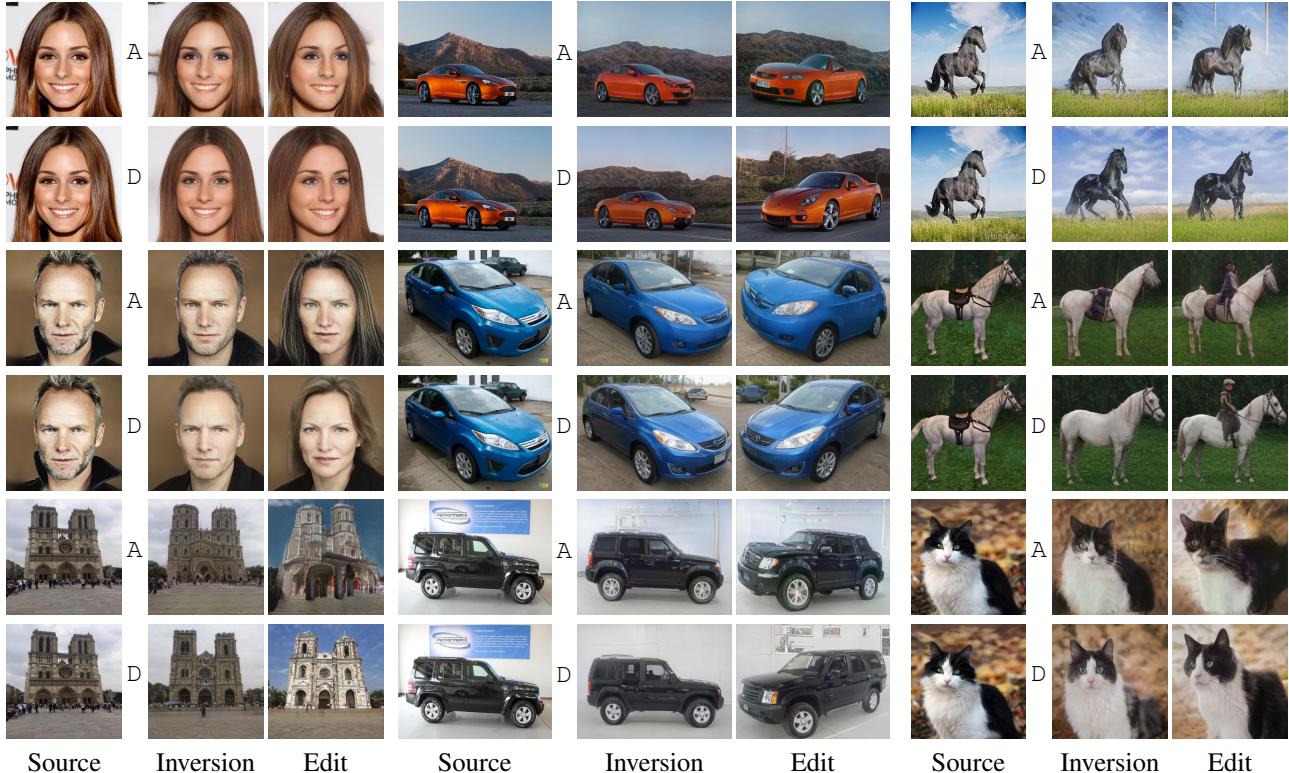


Figure 8: We show triplets of a source image, its inversion, and an edit applied on the inverted image for multiple domains. In the odd rows, the inversions are obtained by our baseline encoder (A). In the even rows, we use configuration D, which encodes images closer to \mathcal{W} . Observe the tradeoff between the distortion and perceptual quality of the inverted images. For example, in the image of the white horse, observe the lower distortion of the inverted image using configuration A (e.g. the saddle is preserved). However, the perceptual quality is lower than that obtained by D (e.g. the horse’s head is not realistic). With respect to editability, notice how in the top left image of the female, the pose edit does not faithfully change the hair in A. Conversely, D obtains a realistic and visually-pleasing editing result at the cost of a subtle degradation in distortion. From top to bottom and left to right the edits are: head pose, gender, daylight, viewpoint (x3), horse pose, horse rider, cat pose.

8.2. Tradeoff

Here we show qualitative and quantitative results to support the existence of both the distortion-perception and distortion-editability tradeoffs. More specifically, we show that approaching \mathcal{W} tends to increase the distortion, perceptual quality, and editability. We start by showing that inversions obtained by our complete method (denoted by configuration D) are much closer to \mathcal{W} than inversions obtained by configuration A, which reduces to pSp with the addition of the generic similarity loss defined in Section 6.1. Note that for faces, configuration A is equivalent to pSp. By comparing the distortion and editability of the two variants, we derive insights about the tradeoff of the latent space.

Which latent vectors are close to \mathcal{W} ? We begin by inspecting the variation of the latent codes produced by configurations A and D on the CelebA-HQ [18] test set. Specif-

ically, given a latent code $w = (w_1, \dots, w_k) \in \mathcal{W}_*^k$ obtained from the encoder, we calculate $\|w - \bar{w}_\mu\|_2$ where $w_\mu = \frac{1}{k} \sum_{i=1}^k w_i$ and $\bar{w}_\mu = (w_\mu, \dots, w_\mu)$. We then take the average over all embeddings obtained from the test set. We obtain a variation of 324.76 for codes generated by configuration A and 20.18 for codes generated by configuration D, respectively. Therefore, latent codes obtained from D are closer to \mathcal{W} in the sense of having a lower variation.

The deviation from the distribution \mathcal{W} is more difficult to measure as there is no known explicit model for it. We therefore perform the following test. We sample a single $w \in \mathcal{W}$ by using StyleGAN’s mapping function, and generate the corresponding image. We then encode the resulting image back into StyleGAN’s latent space using both configurations A and D. We repeat this procedure on 5,000 randomly sampled $w \in \mathcal{W}$ vectors, and measure the average Euclidean distance between the source w and the resulting latent code $w' := E(G(w))$. That is, we compute $\mathbb{E}_{w \in \mathcal{W}} [\|w' - w\|_2]$.

Domain	Conf.	Distortion		Perception		Editability	
		L_2	LPIPS	FID	SWD	FID	SWD
Faces	A	0.03	0.17	25.17	48.72	62.46	48.75
	D	0.05	0.23	30.96	40.54	81.08	43.63
Cars	A	0.10	0.32	10.56	22.08	12.92	24.30
	D	0.10	0.32	12.18	22.71	15.44	26.83
Horse	A	0.11	0.36	34.13	35.79	40.48	31.84
	D	0.16	0.45	37.07	31.91	35.31	32.18
Cats	A	0.10	0.41	42.60	32.83	217.44	45.13
	D	0.14	0.48	49.90	40.59	222.30	51.07
Church	A	0.09	0.32	25.96	38.90	26.91	29.63
	D	0.13	0.40	27.09	31.18	21.87	23.32

Table 3: Perceptual quality metrics computed on inversions and edited images obtained by configurations A and D. For all computed metrics, lower is better. Recall that latent codes obtained by D are encouraged to be close to \mathcal{W} , and therefore the distortion is higher. Note that FID and SWD contradict each other and do not reflect human judgments.

For the facial domain, we obtain an average distance of 364.51 for configuration A and 57.67 for configuration D. For cars, we obtain 316.14 for configuration A and 39.85 for D. Note that while Euclidean distance does not necessarily reflect the distance within the \mathcal{W} space, the distance obtained by encoding with configuration A is an order of magnitude greater than that of configuration D, suggesting that the encoding of configuration D is indeed closer to \mathcal{W} , even within the subspace. Therefore, latent codes obtained by configuration D are closer to \mathcal{W} in both the sense of their deviation from the distribution \mathcal{W} and in the sense of the variation between their different style codes.

Showing the tradeoff Having measured the proximity of each method to \mathcal{W} , we now turn to showing the existence of the distortion-editability and distortion-perception tradeoffs. Recall that our claim is that latents that are closer to \mathcal{W} have higher distortion, but better perceptual quality and editability. As we have shown, latents obtained by configuration D are closer to \mathcal{W} than those obtained by A. Therefore, to support our claim, we use their respective latent codes.

In Figure 8, we provide a visual comparison of the inversions obtained by configurations A and D. For each domain, we also perform an edit for each inverted image. For cars, we use directions obtained by GANSpace [14]; for faces we use StyleFlow [3]; and for horses, cats, and churches we use SeFa [35]. As can be observed, while the distortion of images inverted by configuration A is lower, the visual quality of both the reconstructed and edited images inverted by configuration D is much higher.

Next, we perform a quantitative evaluation by measuring metrics detailed in Section 7. The results are provided in Table 3. We expect the distortion achieved by configuration

Domain	Conf.	Perception	Editability
Faces	A	43.67%	14.33%
	D	56.33%	85.67%
Cars	A	25.67%	18.33%
	D	74.33%	81.67%
Horse	A	6.50%	13.50%
	D	93.50%	86.50%

Table 4: User study results. Each cell displays the percentage of respondents that favored the corresponding configuration for each of the two evaluated aspects. Observe, that both the edited and reconstructed images produced by D were chosen as being more realistic across all evaluated domains.

A to be lower, as it encodes images further away from \mathcal{W} . This can indeed be seen in the results. Note that for both the perception and editability, FID and SWD mostly *contradict* each other. This provides additional evidence for the “leakage” of distortion into the perceptual quality measurements. As shown in Figure 8, reconstructions obtained by A and the subsequent edits applied contain many artifacts which are not faithfully captured by the FID and SWD measures.

Finally, we conduct a qualitative human evaluation to evaluate the perceptual quality and editability of the two configurations. As editability is measured by the perceptual quality of the edited images, all questions were with regard to perceptual quality. The user study consisted of two sections, each including several questions from the facial, cars, and horses domains. A total of 87 respondents participated in the study. In each question, the respondents were given a side-by-side comparison of images generated by configurations A and D, and were asked to choose the more realistic image. In the first section, the images shown were reconstructions of real images sampled from the test set. In the second section, the shown images were of results obtained after performing editing on inverted latent codes. The results of the user study are shown in Table 4. As can be seen, the perceptual quality of images obtained by configuration D is higher both on the reconstructed and edited images. This further shows that the common perceptual metrics (e.g. FID and SWD) do not accurately reflect human perceptual judgement.

Controlling the tradeoff The above analysis shows the existence of the distortion-editability and distortion-perception tradeoffs. We now show that it is a continuous tradeoff that can be controlled. We demonstrate this by considering inversions obtained by pSp and by our e4e method. We interpolate between their inverted latent codes and obtain additional latent codes with a monotonically increasing proximity to \mathcal{W} . Then, we manipulate the codes by applying the same edits using StyleFlow. In Figure 10 we show an example of the above process.

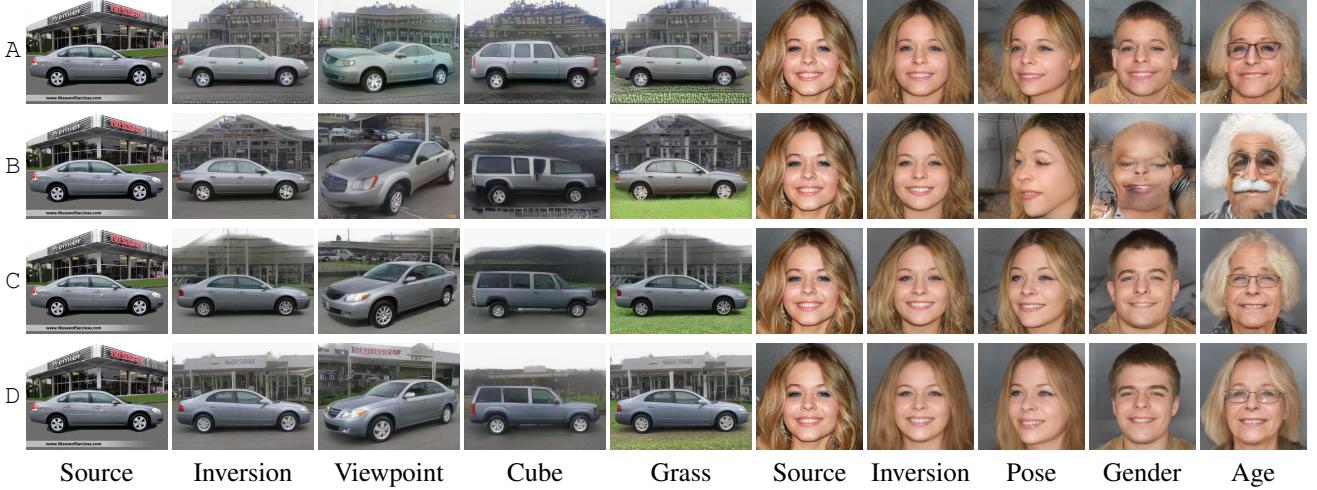


Figure 9: Inversion followed by a series of edits for each of our configurations where configuration D is our complete e4e method. As can be seen, the distortion in the first row is the lowest while the perceptual quality of both the inverted and edited images in the last row is the highest.



Figure 10: The distortion-perception and distortion-editability tradeoffs. Zoom-in for details. The image on the left is the source image. In the top row, we show a series of images, where the leftmost image is the reconstruction obtained by pSp, and the rightmost is obtained by e4e. As we move to the right, the inversion approaches \mathcal{W} , the distortion becomes worse, and the perceptual quality becomes better. Then, for each of the inverted and interpolated images we perform a gender edit using StyleFlow. Notice that as the latent code used for editing approaches \mathcal{W} , the perceptual quality becomes significantly better. For example, observe the non-realistic hair in the leftmost edited image.

8.3. Latent Editing Consistency

We now turn to evaluating the e4e encoder using the newly proposed LEC method presented in Section 7. Qualitative and quantitative results are displayed in Figure 11 and Table 5 with more results provided in the supplementary materials. The evaluation settings are described in Table 5. As can be observed, configuration D is superior to A.

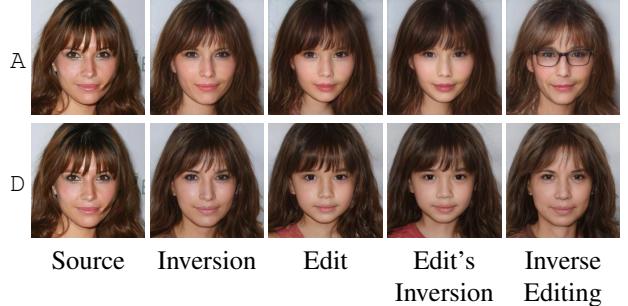


Figure 11: Comparing the LEC measure of different configurations. The desired result is for the “Inverse Editing” image to be an exact reconstruction of the “Inversion” image. As can be seen, configuration D achieves superior results.

		Faces				
Conf.	f_θ	Young	Old	Smile	No Smile	Avg
A		63.03	59.42	48.08	48.15	54.67
D		24.33	24.58	19.92	20.46	22.32
		Cars				
Conf.	f_θ	Pose I	Pose II	Cube	Color	Grass
A		186.95	181.50	133.93	83.93	89.78
D		56.28	56.37	70.46	28.19	33.06
						Avg
						135.22
						48.87

Table 5: The LEC measure computed for the facial and cars domains. For the facial domain, we perform editing using InterFaceGAN and for the cars domain we apply GANSpace. The results are obtained by averaging over the entire test set. Note that lower is better.

Domain Conf.		Distortion		Perception		Editability	
		L_2	LPIPS	FID	SWD	FID	SWD
Faces	A	0.03	0.17	25.17	48.72	62.46	48.75
	B	0.04	0.18	28.09	39.98	149.85	69.86
	C	0.04	0.19	27.36	33.96	143.03	41.94
	D	0.05	0.23	30.96	40.54	81.08	43.63
Cars	A	0.10	0.32	10.56	22.08	12.92	24.30
	B	0.11	0.32	11.16	24.06	17.85	29.78
	C	0.11	0.33	12.47	38.08	16.22	44.73
	D	0.10	0.32	12.18	22.71	15.44	31.45

Table 6: Quantitative comparison of the e4e configurations.

8.4. Ablation Study

In Section 8.2 we demonstrated that configuration D encodes real images into more editable regions comparable to configuration A. Here, we perform an ablation study of the configurations described in Table 2. More specifically, we aim at understanding the importance of each of the principles described in Section 5 for improving editability. In Figure 9, we provide an inversion and a series of edits for each configuration.

In configuration A, we see a low distortion and low perceptual quality in both the inverted and edited images. For example, observe the hair of the woman in the inverted image, and the shape of the car when editing the viewpoint. In configuration B, which uses the delta-regularization, but no latent discriminator, we observe a slight improvement in image sharpness and overall realism of the inverted image compared to A. For example, observe the more realistic hair of the woman. On the other hand, the resulting latent code is highly un-editable (e.g., see the cube and age edits). By using the latent discriminator, but no delta-regularization in configuration C, we observe a significant improvement in perceptual quality of both the inverted and edited images. For example, observe the realistic nature of the grass compared to the grass generated in A. As one can see, by combining the delta-regularization with the latent discriminator in configuration D, we get the best perceptual quality in both the inverted and edited images. For example, observe the realistic head shape when editing the woman’s gender compared to C.

8.5. Comparison to Other Methods

We now compare e4e with other state-of-the-art StyleGAN inversion methods. Specifically, we compare our e4e approach with both optimization-based and encoder-based inversion methodologies.

Optimization-based methods Although there has recently been significant progress in encoder-based inversion, most methods still resort to using a per-image optimization.

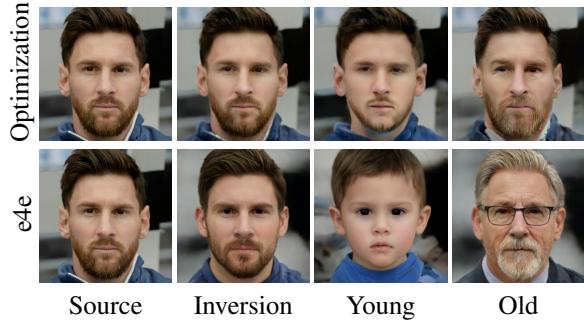


Figure 12: Embedding the source image into \mathcal{W}_*^k using StyleGAN2 optimization yields low distortion and a near perfect reconstruction. However, the obtained latent code behaves poorly when performing latent space editing (here age). On the other hand, latent codes obtained by our method are more suitable for editing, at the cost of a higher distortion.



Figure 13: The applicability of our encoder-based approach for patch-editing. Embedding the patch-edited image into \mathcal{W}_*^k using StyleGAN2 optimization yields a perceptually poor reconstruction, but of low distortion. Although it has a higher distortion, e4e’s inversion is more realistic.

There are several points that should be considered when using direct optimization: (i) it is computationally intensive; (ii) it tends to favor low distortion over editability (see Figure 12); and (iii) it tends to over-fit on the input image, which is undesirable in many applications (see Figure 13).

Encoder-based methods We now compare our method with the current state-of-the-art encoder-based inversion methods: IDInvert [45] and pSp [31]. We focus on showing that previous methods tend to prefer low distortion over high editability. In contrast, we show that e4e achieves a slightly higher distortion, but much better editability. We evaluate the three methods on the facial and cars domains.

For the human facial domain, we use the official pre-trained inversion models for both IDInvert and pSp. For performing editing on the facial domain, we use editing directions from InterFaceGAN [34]. For IDInvert we use their official editing directions. As InterFaceGAN is originally trained using StyleGAN1, we retrain their method on a StyleGAN2 generator and follow the same procedure as described in their paper to obtain the editing directions. These are then used for editing the latents obtained by pSp and e4e.

Domain	Method	Distortion		Perception		Editability	
		L_2	LPIPS	FID	SWD	FID	SWD
Faces	pSp	0.03	0.17	29.57	21.35	73.70	32.30
	IDInvert	0.06	0.22	20.10	31.31	129.00	36.97
	e4e	0.05	0.23	35.47	31.24	96.93	36.02
Cars	pSp	0.10	0.30	17.63	25.95	21.00	24.46
	IDInvert	0.13	0.31	8.26	24.79	18.18	18.94
	e4e	0.10	0.32	12.18	22.71	15.44	26.83

Table 7: Quantitative comparison to IDInvert and pSp. As can be seen, our inversion usually has greater distortion, which is expected. However, the perceptual metrics contradict each other, resulting in no clear winner. As discussed, perceptual quality is best evaluated using human judgement.

For the cars domain, we retrain both IDInvert and pSp with an input resolution of 512×384 using a StyleGAN2 generator. Note that while training IDInvert took over three weeks on two NVIDIA P40 GPUs, our e4e encoder was trained for only three days using a single P40 GPU. For performing editing on the inversions, we use editing directions obtained by GANSpace [14].

In Figure 14, we provide a qualitative comparison of the three methods. As can be seen, when editing the inverted images with e4e, the perceptual quality is significantly higher. For example, for IDInvert observe the grass in the first example (column 4) and the unrealistic colors in the second example (column 3). Also, observe the low quality faces generated when applying the age edit. For pSp, observe the car’s warped shape when applying the pose edit (column 2) and the unrealistic hair obtained on the edited images.

In Table 7 we provide a quantitative evaluation. For distortion we show both L_2 and LPIPS similarities. As can be seen, pSp achieves the lowest distortion of the three methods. For the completeness of the evaluation, we additionally show the FID and SWD metrics. However, as can be seen they contradict each other, again.

9. Discussion and Conclusions

The latent space of StyleGAN and its extensions are powerful, in the sense that they are semantically disentangled, and thus allow for various meaningful edits. However, they are complex spaces. Their quality is not fairly distributed across the whole space, where some regions are more well-behaved than others. Furthermore, the \mathcal{W}_*^k space is huge — much larger than the space of all natural images — and thus, many images have more than a single latent representation. In our work, we make the first steps in analyzing the structure of this complex space, aiming to characterize well-behaved regions to guide the inversion of images into these regions.

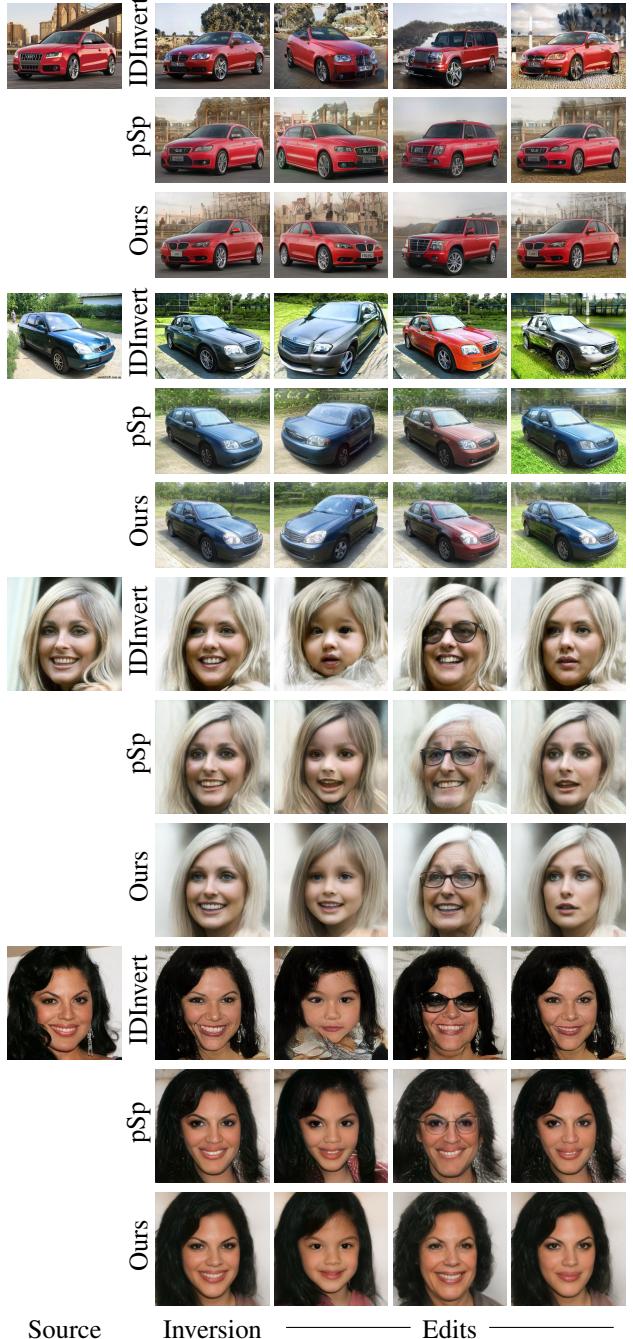


Figure 14: Qualitative comparison to IDInvert and pSp. As can be seen, our inversion results have subtly greater distortion, but the subsequent editing is significantly more realistic and visually-appealing.

Our main contribution is twofold: (i) we propose means for encouraging the encoding of a real image to be mapped into well-behaved regions of \mathcal{W}_*^k , and (ii) we design an encoder and demonstrate its performance, in light of the tradeoff between distortion and editability. We also discuss the difficulties in evaluating reconstruction and editability and propose evaluation protocols built upon commonly used measures. In a sense, our presented method complements image manipulation methods by facilitating higher editing quality on *real* images.

Generally speaking, our encoder encourages mapping close to \mathcal{W} which works well since the space around \mathcal{W} is still surprisingly highly expressive. Moreover, this principle can be adopted for problems beyond image inversion. For example, it can be applied to map latent vectors that represent more than one image, or say a combination of two, like the disentangled representation of identity and pose [27], or a blend of two images, to a proper latent code of the target image which is likely to exist in proximity of \mathcal{W} . We are planning to explore this research direction.

Our inversion scheme is generic, and we have demonstrated its performance on five challenging and diverse domains. Note, however, that some domains are harder than others. Human faces are well-structured, simplifying the training of the encoder. The domain of horses, for example, is much more complex as it is unstructured and it has many modes. Hence, training an encoder for such a domain is much more challenging. In the future, we would like to consider multi-modal generators, like that of Sendik *et al.* [33], and develop an encoder into multi-modal latent spaces.

Finally, here we consider the inversion into a given latent space. In the future, it would be interesting and challenging to consider fine-tuning the generator and train both the encoder and decoder for their common target objective for specific downstream tasks.

References

- [1] R. Abdal, Y. Qin, and P. 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. [2](#), [3](#), [4](#), [5](#)
- [2] R. Abdal, Y. Qin, and P. 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. [3](#)
- [3] R. Abdal, P. Zhu, N. Mitra, and P. Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows, 2020. [2](#), [4](#), [8](#), [10](#), [17](#), [21](#)
- [4] Baylies. [stylegan-encoder](#), 2019. Accessed: January 2021. [3](#), [5](#)
- [5] Y. Blau and T. Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6228–6237, 2018. [2](#), [4](#)
- [6] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning, 2020. [7](#)
- [7] E. Collins, R. Bala, B. Price, and S. 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](#), [3](#)
- [8] A. Creswell and A. A. Bharath. Inverting the generator of a generative adversarial network. *IEEE transactions on neural networks and learning systems*, 30(7):1967–1974, 2018. [3](#)
- [9] J. Deng, J. Guo, N. Xue, and S. 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. [7](#), [16](#)
- [10] E. Denton, B. Hutchinson, M. Mitchell, and T. Gebru. Detecting bias with generative counterfactual face attribute augmentation. *arXiv preprint arXiv:1906.06439*, 2019. [2](#)
- [11] L. Goetschalckx, A. Andonian, A. Oliva, and P. Isola. Ganalyze: Toward visual definitions of cognitive image properties, 2019. [2](#)
- [12] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. 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. [1](#), [6](#)
- [13] S. Guan, Y. Tai, B. Ni, F. Zhu, F. Huang, and X. Yang. Collaborative learning for faster stylegan embedding. *arXiv preprint arXiv:2007.01758*, 2020. [3](#)
- [14] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris. Ganspace: Discovering interpretable gan controls. *arXiv preprint arXiv:2004.02546*, 2020. [1](#), [2](#), [8](#), [10](#), [13](#), [19](#), [23](#), [25](#)
- [15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015. [7](#)
- [16] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018. [7](#)
- [17] A. Jahanian, L. Chai, and P. Isola. On the “steerability” of generative adversarial networks. *arXiv preprint arXiv:1907.07171*, 2019. [2](#)
- [18] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. [8](#), [9](#), [16](#), [21](#), [22](#), [28](#), [29](#), [30](#)
- [19] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila. Training generative adversarial networks with limited data. In *Proc. NeurIPS*, 2020. [1](#), [2](#), [3](#)
- [20] T. Karras, S. Laine, and T. 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](#), [2](#), [3](#), [8](#)
- [21] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. 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](#), [3](#), [19](#)
- [22] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition*

- (3dRR-13), Sydney, Australia, 2013. 8, 16, 19, 23, 24, 31, 32, 33
- [23] Z. C. Lipton and S. Tripathi. Precise recovery of latent vectors from generative adversarial networks. *arXiv preprint arXiv:1702.04782*, 2017. 3
- [24] Y. Liu, Q. Li, Z. Sun, and T. Tan. Style intervention: How to achieve spatial disentanglement with style-based generators?, 2020. 3
- [25] S. Menon, A. Damian, S. Hu, N. Ravi, and C. Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2445, 2020. 3
- [26] L. Mescheder, A. Geiger, and S. Nowozin. Which training methods for gans do actually converge? *arXiv preprint arXiv:1801.04406*, 2018. 6
- [27] Y. Nitzan, A. Bermano, Y. Li, and D. Cohen-Or. Face identity disentanglement via latent space mapping. *ACM Transactions on Graphics (TOG)*, 39:1 – 14, 2020. 2, 3, 4, 6, 14
- [28] G. Perarnau, J. Van De Weijer, B. Raducanu, and J. M. Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016. 3
- [29] A. Plumerault, H. L. Borgne, and C. Hudelot. Controlling generative models with continuous factors of variations. *arXiv preprint arXiv:2001.10238*, 2020. 2
- [30] J. Rabin, G. Peyré, J. Delon, and M. Bernot. Wasserstein barycenter and its application to texture mixing. In A. M. Bruckstein, B. M. ter Haar Romeny, A. M. Bronstein, and M. M. Bronstein, editors, *Scale Space and Variational Methods in Computer Vision*, pages 435–446, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. 7
- [31] E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. *arXiv preprint arXiv:2008.00951*, 2020. 2, 3, 4, 5, 6, 12
- [32] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016. 7
- [33] O. Sendik, D. Lischinski, and D. Cohen-Or. Unsupervised k-modal styled content generation. *ACM Transactions on Graphics (TOG)*, 2020. 3, 14
- [34] Y. Shen, J. Gu, X. Tang, and B. 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, 8, 12, 28, 29, 30
- [35] Y. Shen and B. Zhou. Closed-form factorization of latent semantics in gans. *arXiv preprint arXiv:2007.06600*, 2020. 2, 3, 8, 10, 26, 27
- [36] N. Spingarn-Eliezer, R. Banner, and T. Michaeli. Gan steerability without optimization. *arXiv preprint arXiv:2012.05328*, 2020. 2
- [37] A. Tewari, M. Elgharib, G. Bharaj, F. Bernard, H.-P. Seidel, P. Pérez, M. Zollhöfer, and C. Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. *arXiv preprint arXiv:2004.00121*, 2020. 1, 2
- [38] A. Voynov and A. Babenko. Unsupervised discovery of interpretable directions in the gan latent space. *arXiv preprint arXiv:2002.03754*, 2020. 2
- [39] B. Wang and C. R. Ponce. A geometric analysis of deep generative image models and its applications. In *International Conference on Learning Representations*, 2021. 2
- [40] Z. Wu, D. Lischinski, and E. Shechtman. Stylespace analysis: Disentangled controls for stylegan image generation, 2020. 1, 3
- [41] J. Wulff and A. Torralba. Improving inversion and generation diversity in stylegan using a gaussianized latent space, 2020. 3
- [42] W. Xia, Y. Zhang, Y. Yang, J.-H. Xue, B. Zhou, and M.-H. Yang. Gan inversion: A survey, 2021. 3
- [43] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop, 2016. 8, 26, 27
- [44] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018. 7
- [45] J. Zhu, Y. Shen, D. Zhao, and B. Zhou. In-domain gan inversion for real image editing. *arXiv preprint arXiv:2004.00049*, 2020. 3, 4, 5, 12
- [46] P. Zhu, R. Abdal, Y. Qin, and P. Wonka. Improved stylegan embedding: Where are the good latents?, 2020. 5

Appendix

In this supplemental document we provide implementation details and further experiments. Zooming-in to better observe fine details is recommended in all figures.

A. Implementation Details

Here, we provide additional implementation details to complement the architecture, training scheme, and objective functions as described in the main text.

Losses We set the loss weights as follows: we set $\lambda_{l2} = 1$ and $\lambda_{lpips} = 0.8$. When training with the latent discriminator D_w , we use $\lambda_{adv} = 0.1$. When applying a progressive delta-based training, we use $\lambda_{d-reg} = 2e^{-4}$. For the cars, horses, and cats domains we set $\lambda_{sim} = 0.5$ for our MOCO-based similarity loss. For the facial domain we set $\lambda_{sim} = 0.1$ over a pre-trained ArcFace [9] facial recognition network. Finally, we set $\lambda_{edit} = 1$.

Progressive Training Only the first w style vector is trained in the first 20,000 training steps. After the first 20,000 steps, we gradually add a delta for the next latent code entry every 2,000 training steps.

Discriminator For our latent discriminator, we use a 4-layer MLP network using 0.2 LeakyReLU activations. We train the discriminator using the Adam optimizer with a fixed learning rate of $2e^{-5}$.

B. Comparing Configurations A and D

We provide additional figures in this document. See Figures 15 to 18. Observe, that our method achieves significantly superior editing in terms of perceptual quality and expressiveness.

C. The Continuous Distortion-Editability Tradeoff

We provide additional results, similar to Figure 10 in the main paper, where we demonstrate that the distortion-editability tradeoff and the fact that the location on the trade-off curve can easily be controlled by interpolation in the latent space. To prevent the suspicion of cherry-picking, we provide uncurated results of the first samples in the CelebA-HQ [18] and Stanford Cars [22] datasets. See Figures 19 to 22.

D. LEC

We provide additional results of the LEC measure, similar to Figure 11 in the main paper. See Figure 23. Note that

our method successfully reconstructs the inversion image after the LEC protocol. This serves as additional evidence that our encoder is well-behaved and suitable for successive semantic latent editing.

E. Additional Results

Last, we provide numerous inversion and editing results obtained using our e4e approach in Figures 24 to 31.



Figure 15: Additional comparison of configurations A and D trained on FFHQ. Here, results are displayed on real images of celebrities collected from the internet. To display the versatile edit capability of configuration D, each pair of rows depicts randomly selected edits performed using StyleFlow [3].

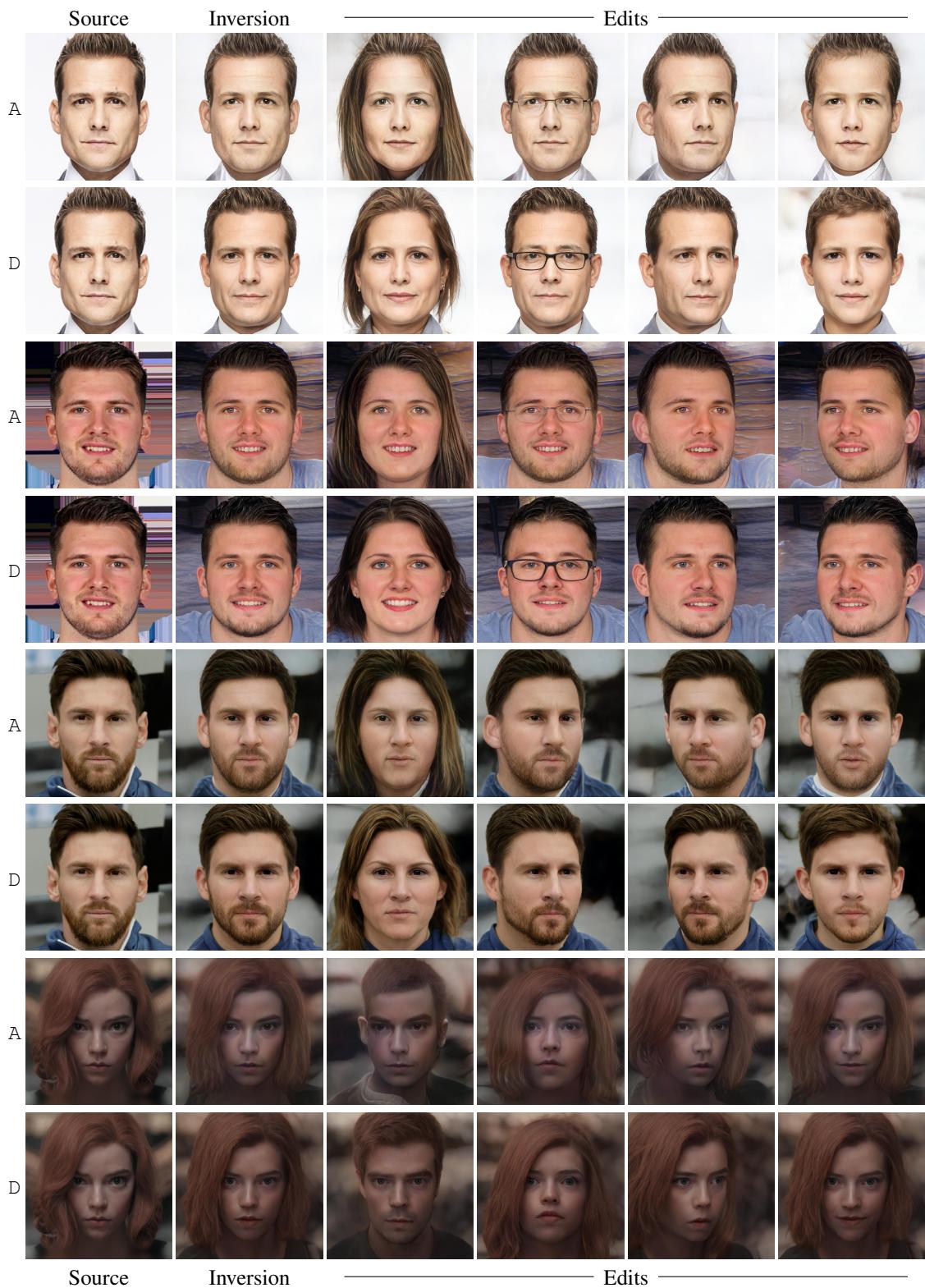


Figure 16: Additional comparison of configurations A and D, following the same format as Figure 15.

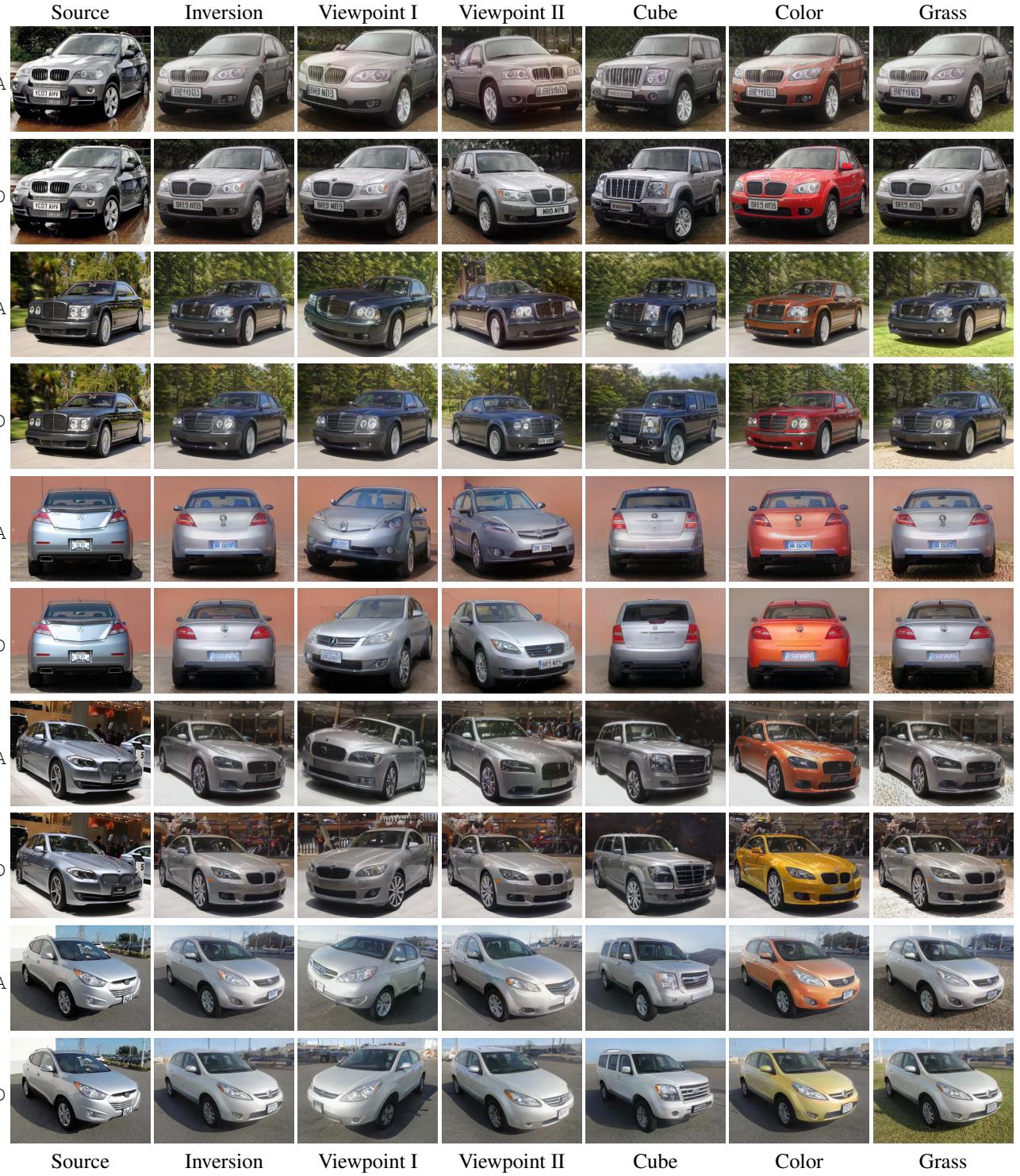


Figure 17: Comparing inversion and editing results of configurations A and D on the Stanford Cars test set [22]. The leftmost column is the real source image, to its right is the reconstruction through StyleGAN2 [21]. Remaining columns are edits along the directions obtained by GANSpace [14]. The column header specifies the edit performed. Note that while configuration A often presents less, i.e. better, distortion, configuration D yields reconstruction and editings with better perceptual quality.

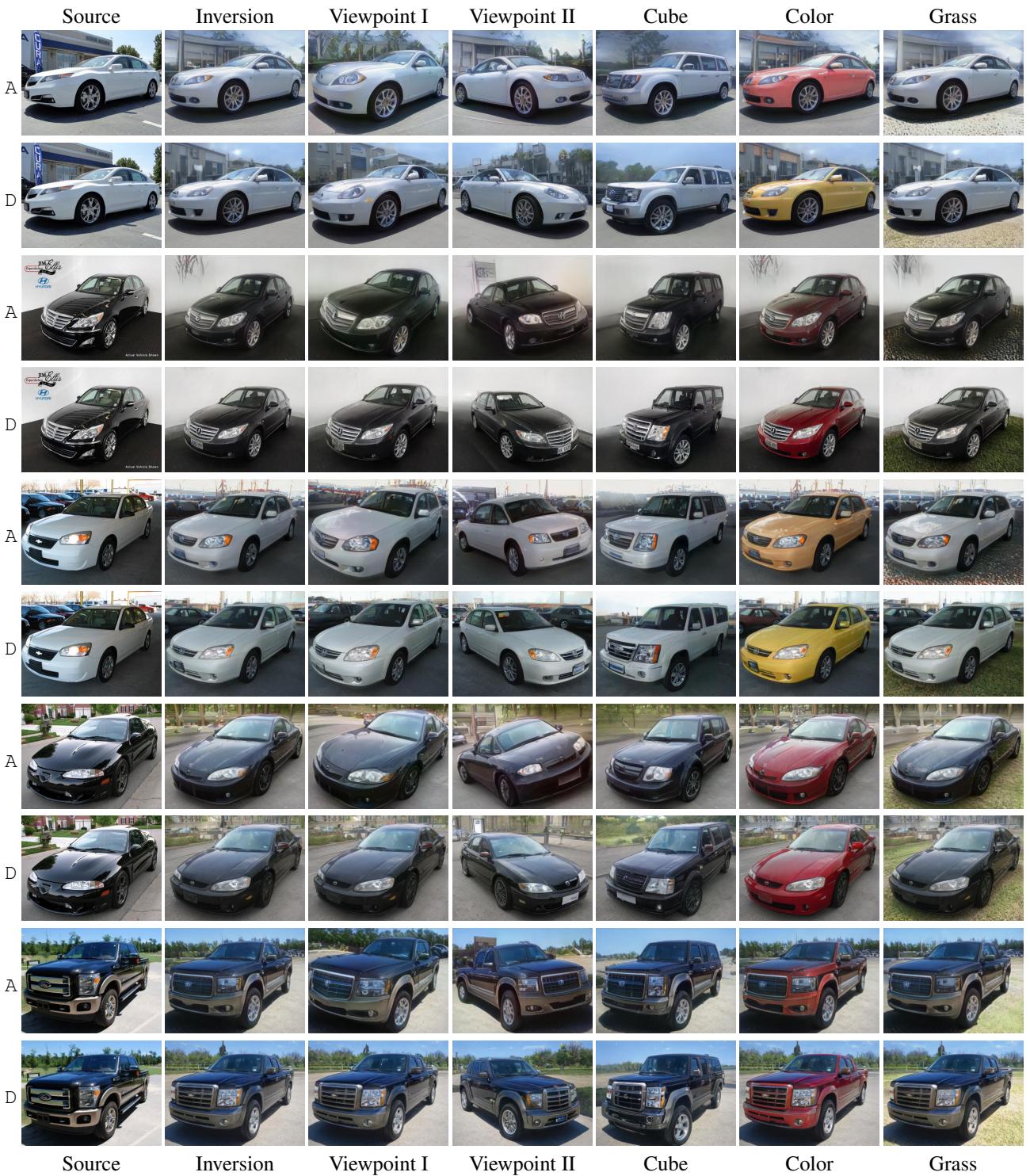


Figure 18: Additional comparison of configurations A and D, following the same format as Figure 17.



Figure 19: Displaying the continuous nature of the distortion-editability tradeoff on the first 8 images of CelebA-HQ [18] test set. The leftmost column is the source, the second and fourth columns are inversions obtained by pSp and e4e, respectively. The third column is the mid-point interpolation between the inversions. The three rightmost columns are pose edits of the inversions and interpolated inversion performed using StyleFlow [3]. Note that, the proximity to the pSp or e4e inversion controls where the latent code lies on the tradeoff curve.



Figure 20: Similar to Figure 19, displaying the next 8 images from the CelebA-HQ [18] test set. The first four pairs of rows perform a gender edit while the last four pairs of rows perform an age edit.

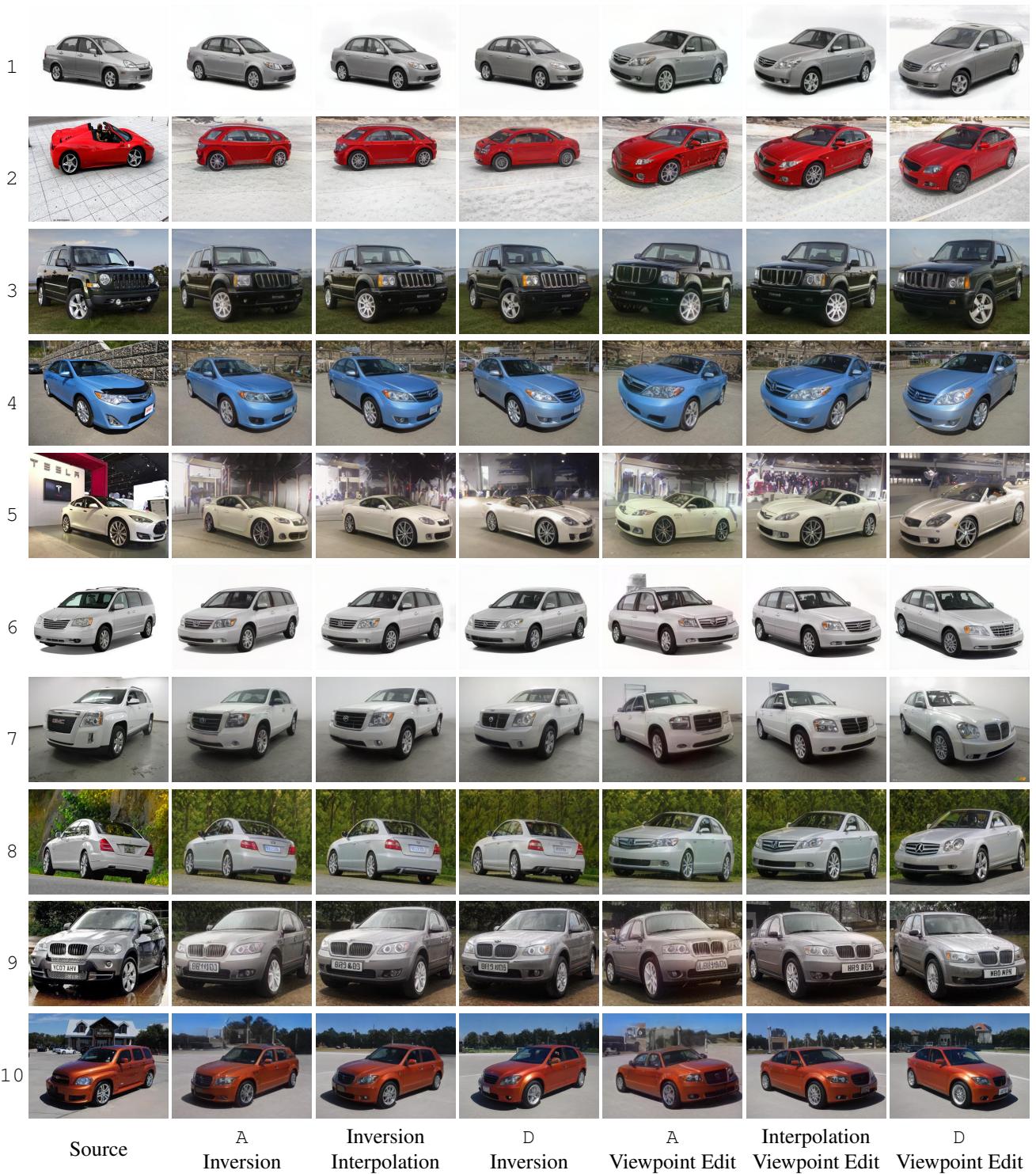


Figure 21: Similar to Figure 19, displaying the first 10 images from the Stanford Cars [22] test set. Here, we perform viewpoint edits using GANSpace [14].

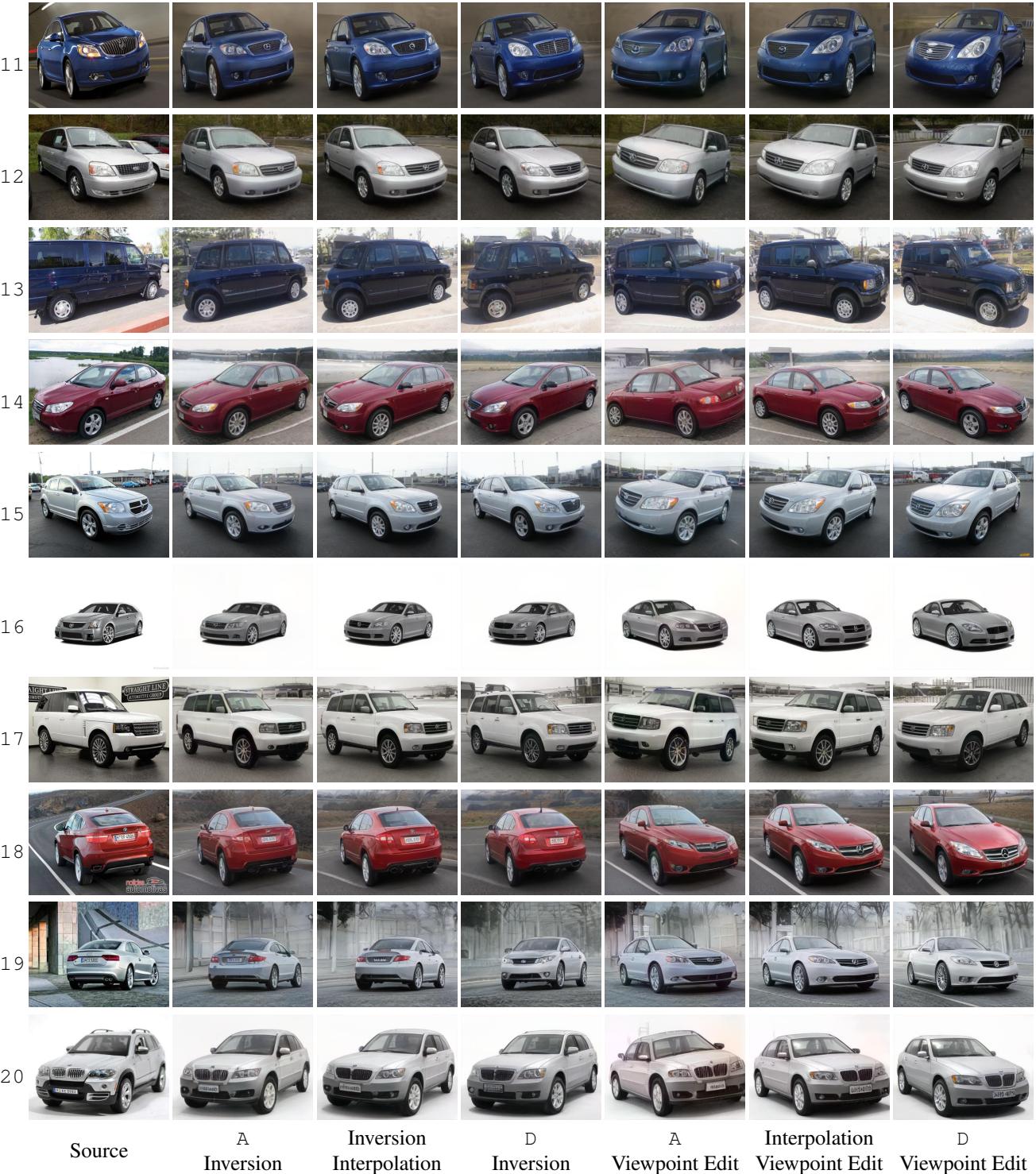


Figure 22: Similar to Figure 19, displaying the next ten images from the Stanford Cars [22] test set.

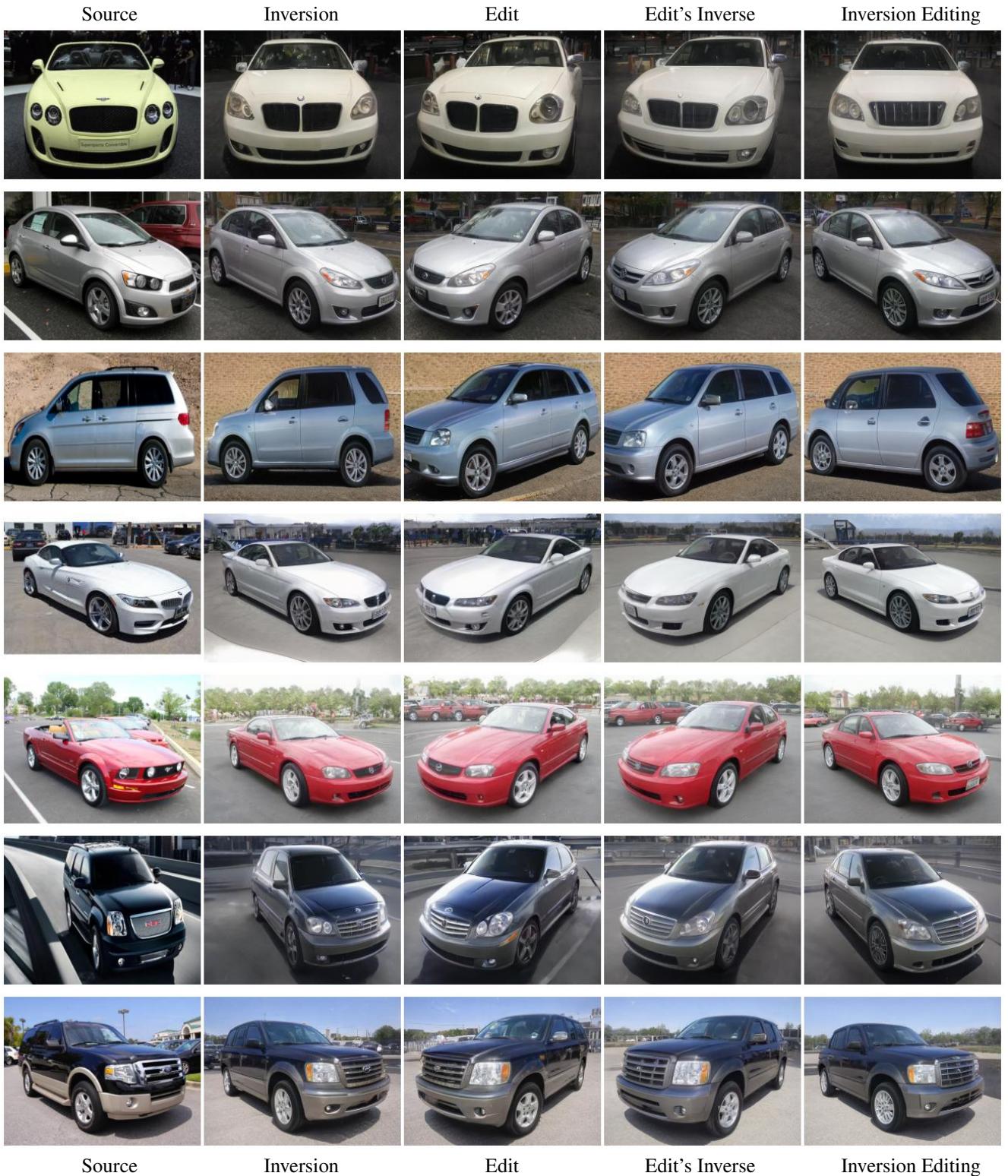


Figure 23: Examples of the LEC protocol over the cars domain using GANSpace [14] to edit the car’s viewpoint. Note that the “Inversion Editing” almost perfectly reconstructs the “Inversion” which serves as evidence that our e4e encoder is well-behaved.



Figure 24: Additional results obtained by e4e over the LSUN horses domain [43]. Various edits, such as pose, head change, and horse rider, are performed using SeFa [35].



Figure 25: Additional results obtained by e4e over the LSUN churches domain [43]. Various edits, such as viewpoint, geometric change, and daylight, are performed using SeFa [35].

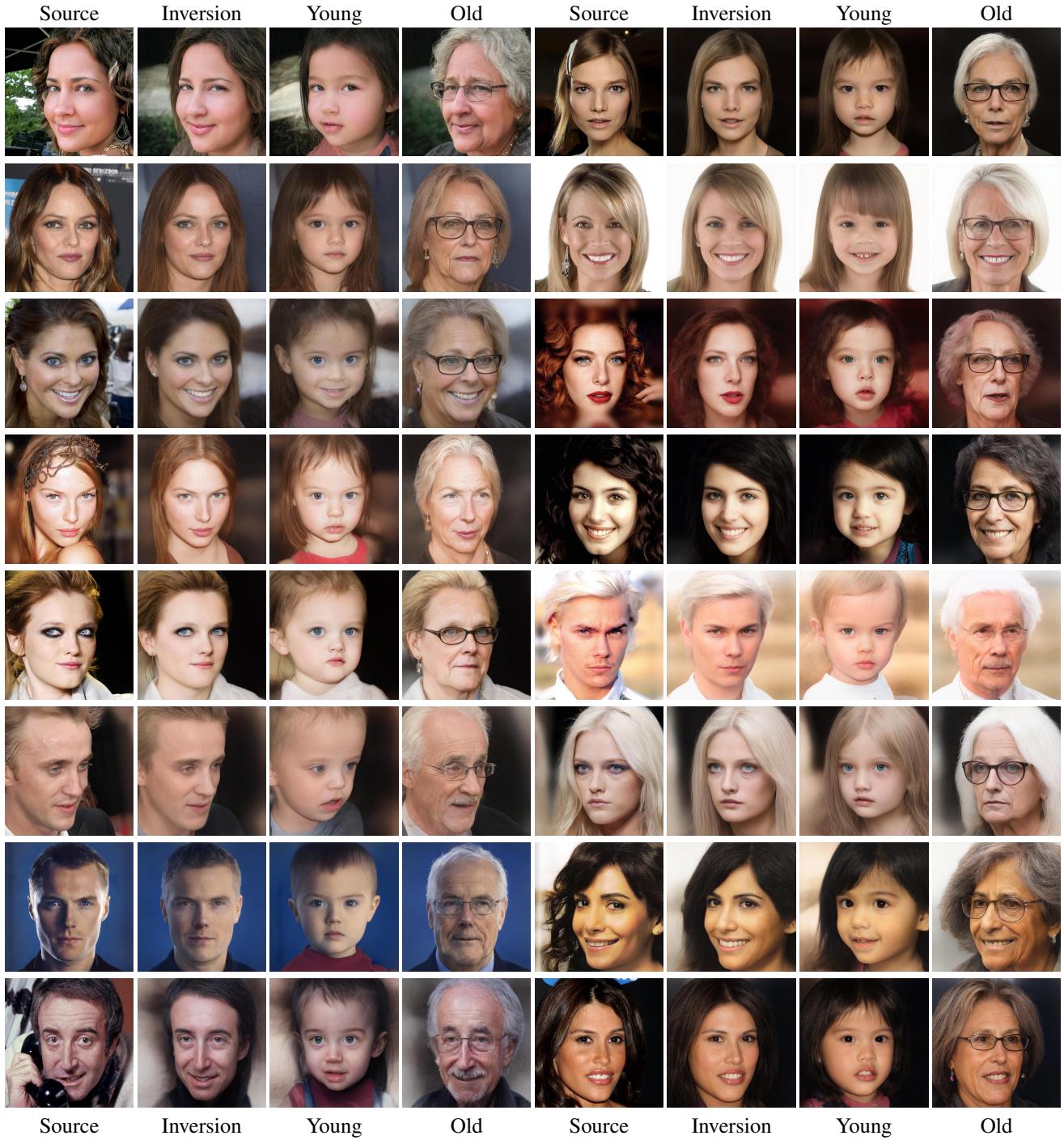
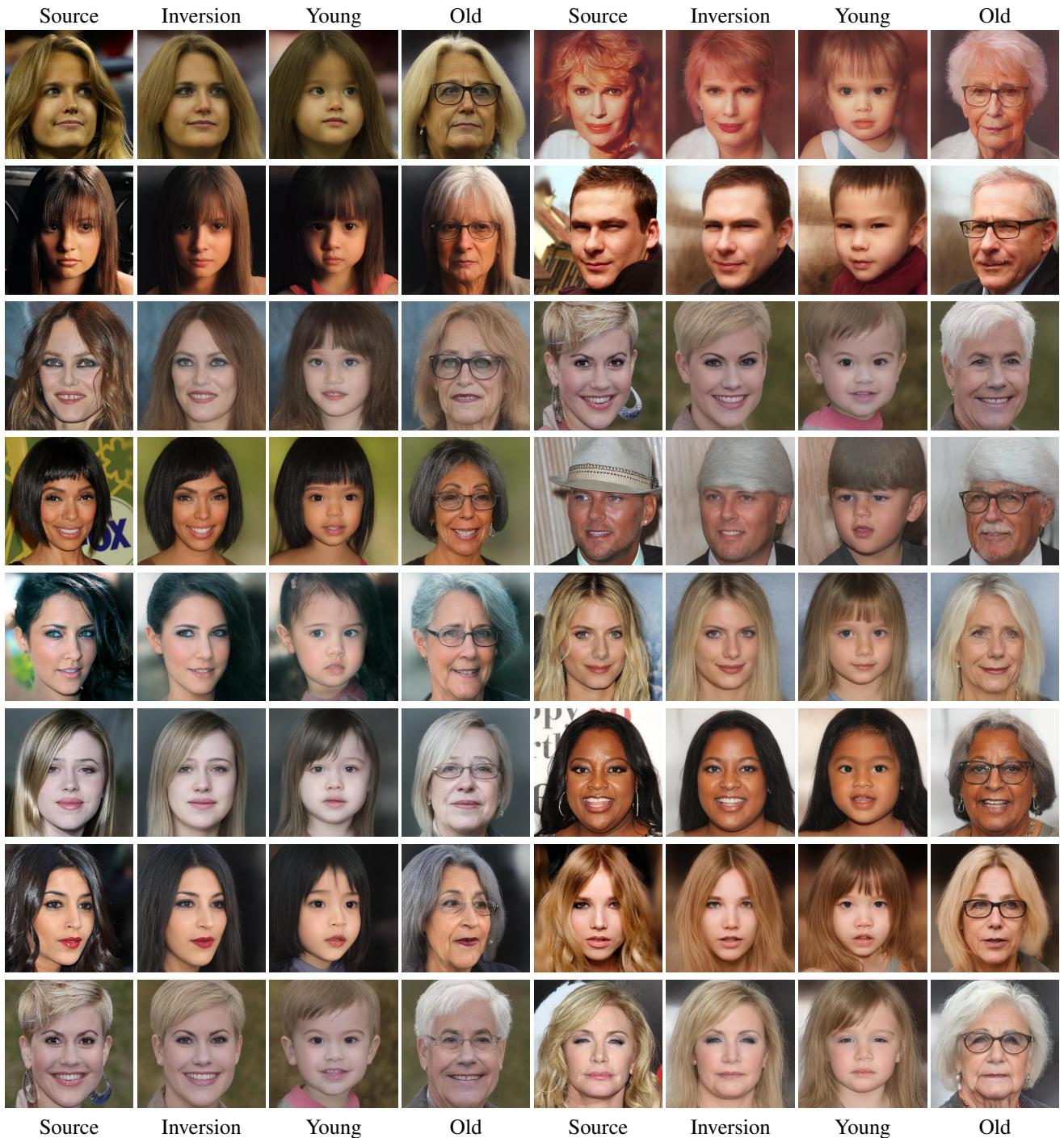




Figure 27: Additional results of our e4e encoder over the first 48 images in the CelebA-HQ [18] test set using InterFaceGAN [34] age editing. In this figure, images 17 – 32 are presented.



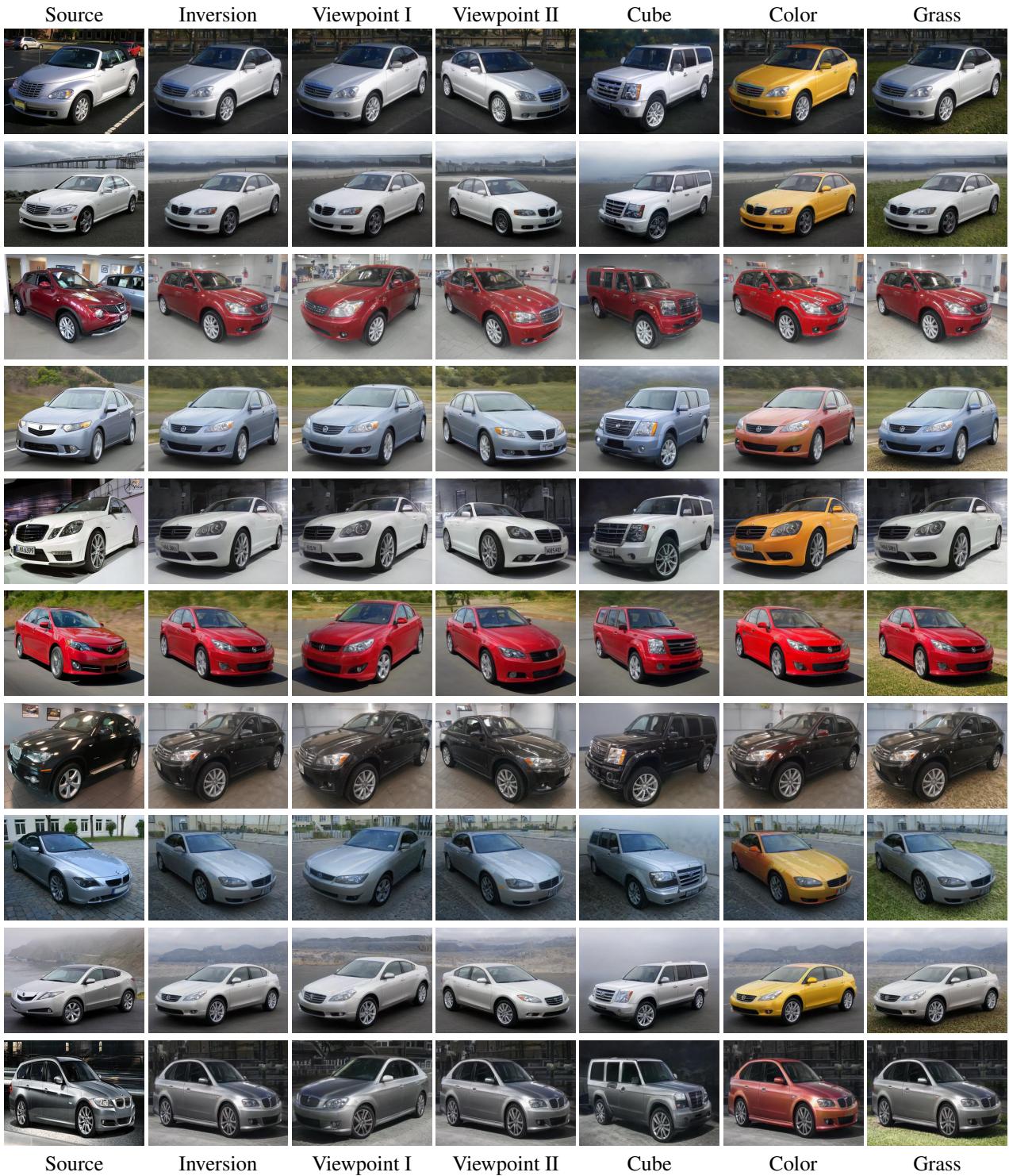


Figure 29: Additional results of our e4e encoder on the cars domain, similar to Figure 15 on the Stanford Cars test set [22].

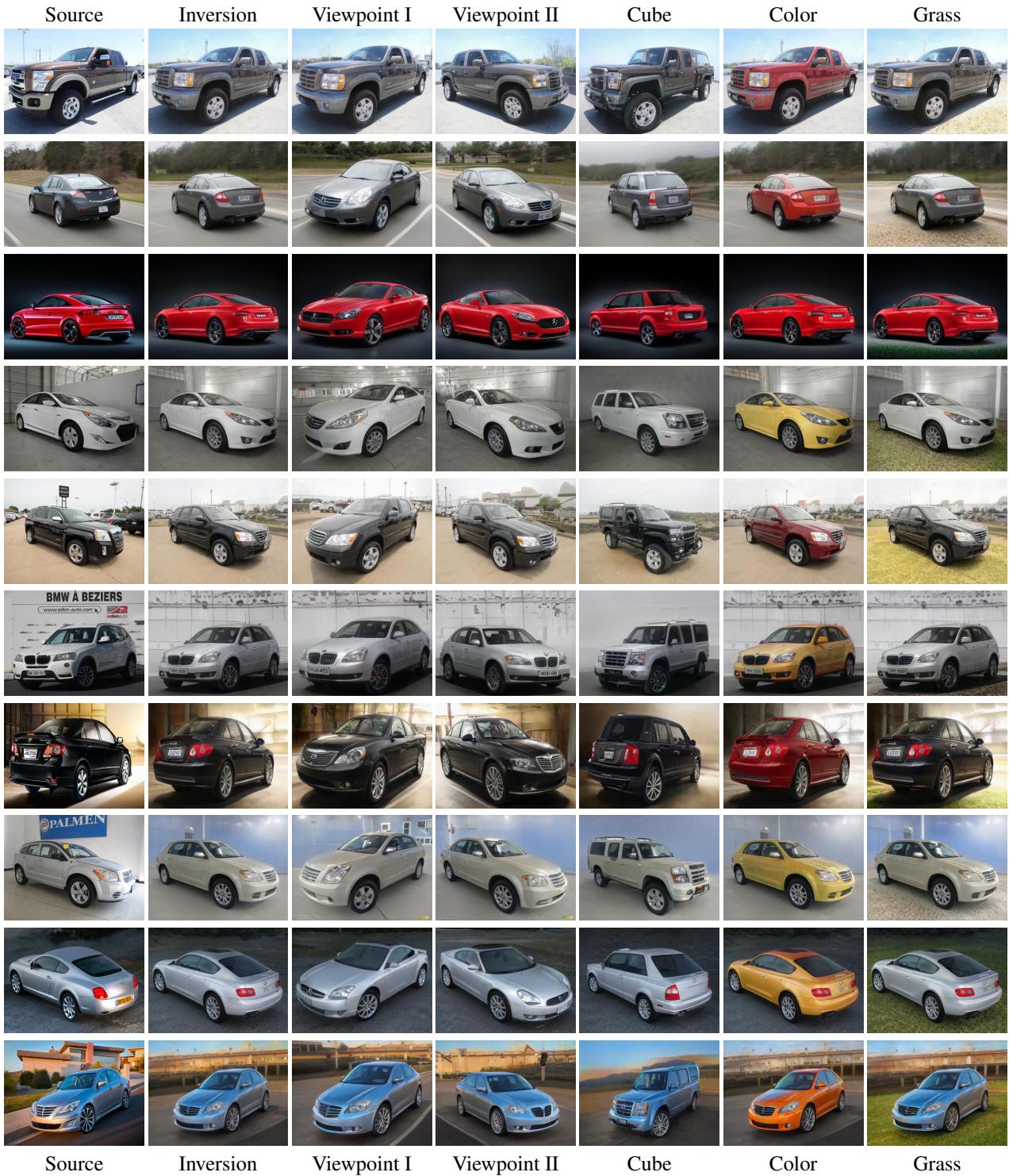


Figure 30: Additional results of our e4e encoder on the cars domain, similar to Figure 15 on the Stanford Cars test set [22].

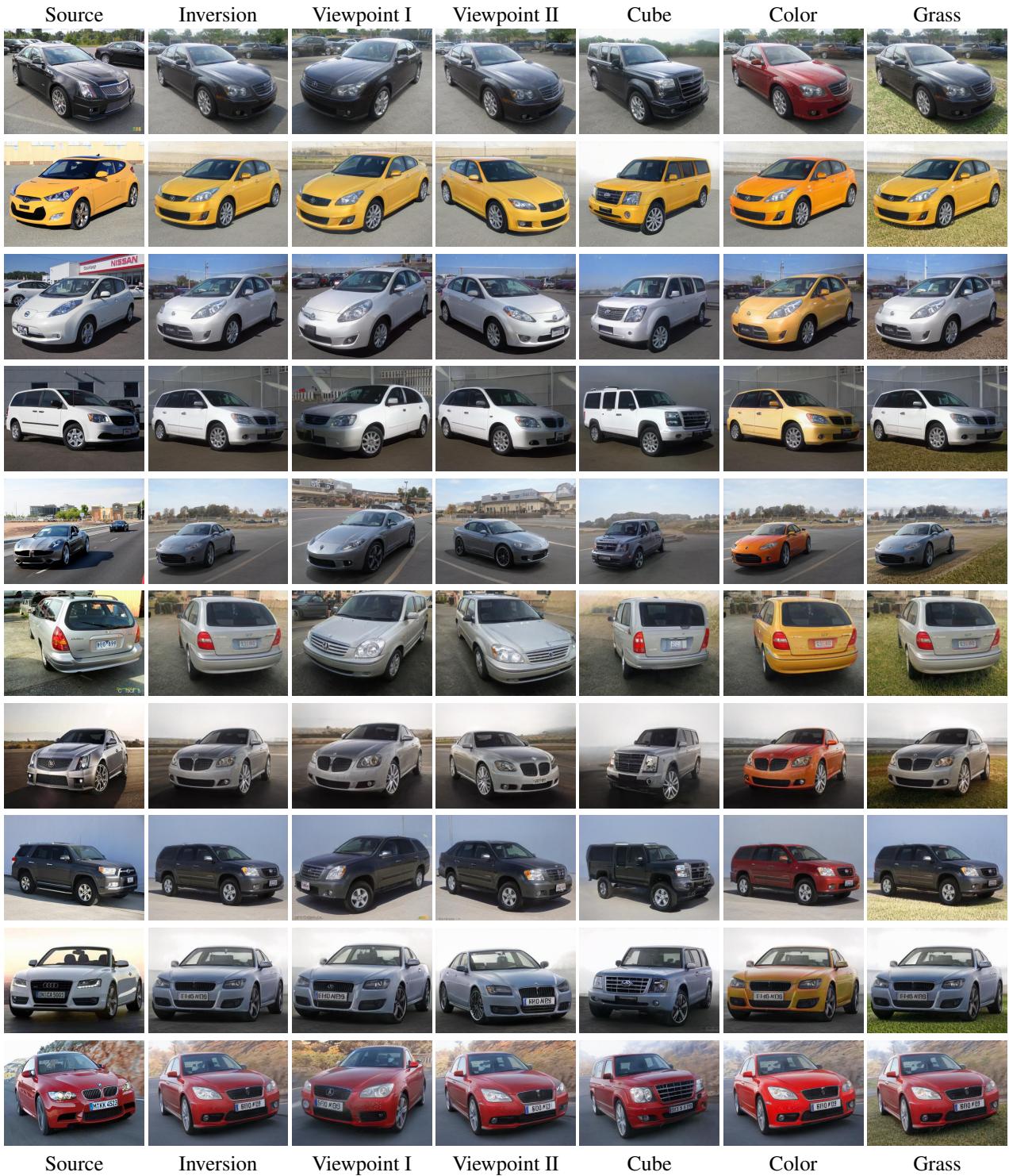


Figure 31: Additional results of our e4e encoder on the cars domain, similar to Figure 15 on the Stanford Cars test set [22].