ReStyle: A Residual-Based StyleGAN Encoder via Iterative Refinement Supplementary Materials

Yuval Alaluf Or Patashnik Daniel Cohen-Or

Blavatnik School of Computer Science, Tel Aviv University

In this supplemental document we provide additional details and analysis to complement those provided in the main manuscript. Along with the additional details, we also perform an ablation study to validate our design choices and provide a large gallery of comparisons and results at full resolution using the proposed ReStyle scheme. Finally, we invite the readers to view the accompanying full-resolution animations.

1. Additional Details

1.1. The ReStyle Encoder Architecture

We begin by providing additional details regarding the ReStyle encoder architecture presented in Section 4.1. Recall that our simplified architecture is derived from the architecture used in Richardson *et al*. [10]. There, the authors employ an FPN-based architecture for encoding real images into the StyleGAN latent space. Specifically, the encoder extracts the style input vectors using three intermediate feature maps of spatial resolutions 64×64 (for inputs $0-2$), 32×32 (for inputs $3-6$), and 16×16 (for inputs $7-17$). Each style vector is extracted from their corresponding feature map using a *map2style* block, which is a small convolutional network containing a series of 2-strided convolutions with LeakyReLU activations. This FPN-based architecture is illustrated in Figure 1.

With ReStyle, we take a simpler approach. Instead of extracting the style vectors from three intermediate feature maps along the encoder, each style input is extracted from the final 16×16 feature map and a map2style block, as illustrated in Figure 3 in the main paper. An ablation study comparing these two architectures is provided below in Section 3.

1.2. Datasets

Here, we provide additional details regarding the datasets used in each of the evaluated domains.

Human Faces. For the human facial domain, we use all $70,000$ images from the FFHQ [5] dataset for training the ReStyle encoders. For evaluations, we use all $2,824$ test images from the CelebA-HQ [8, 4] dataset using the official train-test split.

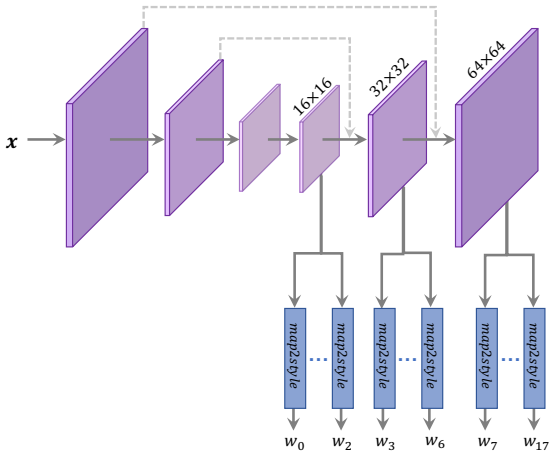

Figure 1. Visualization of the original FPN architecture used in Richardson *et al*. [10] and Tov *et al*. [13]. A visualization of our simplified encoder architecture is provided in the main paper.

Cars. For the cars domain, we use the Stanford Cars [7] dataset with training performed using the $8,144$ training images. For our evaluations, due to the large test set ($8,041$ images), we randomly select $1,000$ images from the test set with all metrics computed using the selected subset.

AFHQ Wild. Here, training and evaluations are performed on $4,738$ and 500 images, respectively, taken from the official AFHQ [1] Wild dataset.

Horses. From the LSUN [14] Horse dataset, we randomly select $10,000$ to be used for training and $2,215$ images used for evaluations.

Churches. For the churches domain we use all $126,227$ training images and 300 testing images from the official LSUN [14] Church dataset.

1.3. Baselines

In our evaluations in Sections 5.2 and 5.4, we compare ReStyle with various encoder-based inversion methods. Below, we provide additional details on the baselines evaluated in the main paper.

IDInvert. We compared our ReStyle approach to the IDInvert encoder from Zhu *et al*. [15] on the human facial and cars domains. For the human facial domain, we use the official pre-trained model, which employs a StyleGAN1 [5] generator. For the cars domain, we re-trained IDInvert using an input resolution of 512×384 and the official StyleGAN2 generator. Note, due to the long training time required by IDInvert (over three weeks on two NVIDIA P40 GPUs), we chose to train IDInvert only on the cars domain.

pSp. For the human facial domain, we use the official pre-trained model from Richardson *et al*. [10]. For all other domains, we trained pSp using StyleGAN2 generators and default pSp hyper-parameters. During training, we also incorporated the MOCO-based similarity loss from [13] which was shown to improve reconstruction quality.

e4e. For our comparison with Tov *et al*. [13], we trained e4e on the AFHQ [1] Wild dataset using the official implementation and default e4e hyper-parameters. All other domains were evaluated using official pre-trained models.

2. ReStyle Analysis

In this section, we provide additional analyses to complement those performed in the main paper (Section 5.3).

Where's the focus? (Part II) In Section 5.3 (Figures 6 and 7), we showed which image regions change the most at each inference step and showed how the magnitude of change decreases over time. There, the resulting Figures were obtained by averaging over all $2,000+$ test images. To complement these Figures, we can examine this behavior while observing each image *independently* rather than averaging over all images. Consider Figure 2. There, we illustrate the intermediate outputs of ReStyle$_{pSp}$ alongside the normalized heat-maps where red denotes a large pixel change and blue denotes a small pixel change. As shown, in the early iterations, ReStyle focuses on adjusting global features such as the head pose or the car shape while in subsequent iterations finer details are adjusted.

How many steps are needed? The ReStyle analysis performed in the main paper and above explore ReStyle's behavior in the image space. Here, we explore the behavior of ReStyle in the latent space. Specifically, we analyze (i) which latent entries change the most across the inference steps and (ii) how many steps are needed for convergence.

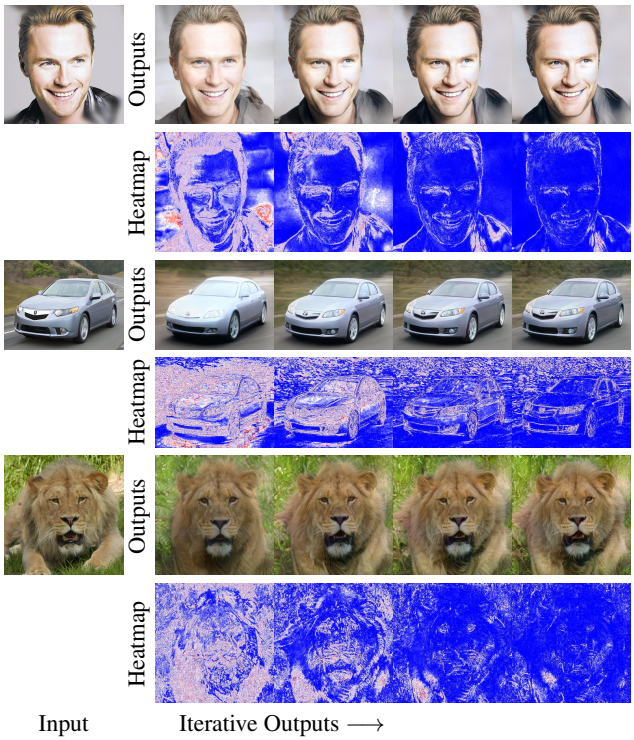

Figure 2. *Visualizing the iterative refinement.* For each image we visualize the iterative outputs produced by ReStyle. Below each output, we show a heatmap illustrating which image regions were altered the most at the corresponding step. Note, all heatmaps are normalized globally with respect to each other. Red regions indicate a large change with blue representing small changes in the pixel-space.

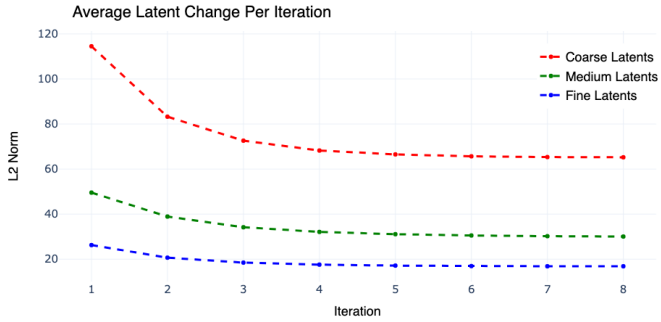

Figure 3. *Which latents change the most?* We plot the average magnitude of change in each group of latent inputs per step during inference. As shown, ReStyle focuses on adjusting the coarse and medium-level inputs and converges after a few steps.

To do so, we focus on the cars domain and perform the following process. Consider some iteration t and some latent entry $\mathbf{w}_l$ where $l \in [1, k]$ and k is the number of style inputs of the generator. To compute how much the values of $\mathbf{w}_l$ change between iterations $t-1$ and t we compute the squared difference between the latent entries averaged across all test samples.

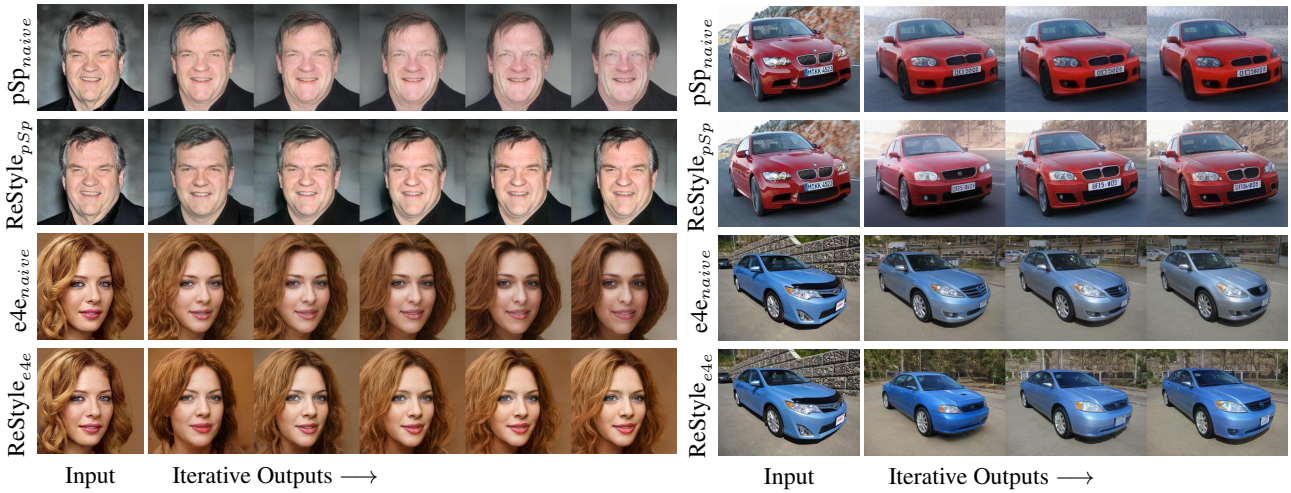

Figure 4. *Ablation study.* As shown, using a pre-trained pSp or e4e encoder and simply feeding the output image multiple times leads to deteriorating reconstruction results. Conversely, our dedicated ReStyle scheme incrementally improves its reconstruction outputs with each additional step.

That is,

$$\mathbf{d}_{l,t} = \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{w}_{l,t}^{(i)} - \mathbf{w}_{l,t-1}^{(i)}\right)^2, \tag{1}$$

$$v_{l,t} = ||\mathbf{d}_{l,t}||_2 \tag{2}$$

where $\mathbf{w}_{l,t}^{(i)}$ is the l-th latent entry of the i-th sample obtained in iteration t and $v_{l,t}$ is the L_2 norm of the average difference $d_{l,t}$. Having computed the average change of each of the k latent entries across all test samples, we group the entries into the coarse, medium, and fine inputs as defined by StyleGAN and compute the average change of each group.

In Figure 3, we plot the change along each step for each of the three groups. As can be seen, the coarse input group attains the most change in the early iterations as the encoder focuses on refining the background and pose of the inversions. Conversely, the fine styles undergo the least change and converge after only a few steps, indicating that refining these aspects (e.g., the color) of the output image may be easier for the encoder. Overall, it can be seen that a few number of steps are needed for the encoder to converge to its final inversion prediction. Another interpretation of the above phenomenon is that the learned residuals decrease at every inference step, as is desired.

3. Ablation Study

In this section, we validate our design choices for our ReStyle training scheme and encoder architecture.

The Iterative Training Scheme. We begin by showing that a dedicated iterative training scheme is truly needed. A natural first attempt for creating an iterative inversion scheme is simply using a pre-trained conventional encoder and passing the output image back as input multiple times. Notice that there are key differences from the above formulation and the ReStyle formulation. First, in the ReStyle scheme, we pass both the input and current output to the encoder. Second, a ReStyle encoder is trained explicitly to output a *residual* with respect to the previous latent at each step.

In Figure 4 we provide a comparison on the human facial and cars domains using both a pSp encoder and e4e encoder using the ReStyle formulation and naive formulation presented above. As can be seen, at each additional step, the naive formulation moves away from the reconstruction of the input image. This shows that our dedicated iterative, residual-based training scheme is indeed needed over readily-available encoders.

The ReStyle Encoder Architecture. Here, we show that our simplified encoder architectural design choice leads to a negligible difference in reconstruction quality while reducing the inference time compared to the FPN-based architecture from Richardson *et al*. [10]. Table 1 summarizes the reconstruction results across 5 domains. For each domain and encoder, we perform 5 steps during inference and display quantitative results computed on the final output. As shown, the two variants attain nearly identical reconstruction quality in terms of both L_2 and LPIPS similarity. However, our simplified architecture is more than 10% faster.

4. Analyzing the Toonify Latent Space

In Section 5.5, we explored a new *encoder bootstrapping* technique for performing image toonification [9]. Rather than initializing ReStyle using the average latent code and corresponding toon image, the iterative process is initialized

Domain	Method	↓ LPIPS	↓ MSE	↓ Runtime
Faces (1024)	ReStyle$_{fpn}$	0.03	0.14	0.538
	ReStyle$_{simple}$	0.03	**0.13**	**0.451**
Cars (512)	ReStyle$_{fpn}$	0.08	0.26	0.411
	ReStyle$_{simple}$	**0.07**	**0.25**	**0.361**
Wild (512)	ReStyle$_{fpn}$	0.06	0.23	0.413
	ReStyle$_{simple}$	0.06	**0.21**	**0.363**
Churches (256)	ReStyle$_{fpn}$	0.09	0.28	0.355
	ReStyle$_{simple}$	0.09	**0.26**	**0.298**
Horses (256)	ReStyle$_{fpn}$	0.09	0.32	0.355
	ReStyle$_{simple}$	0.09	**0.31**	**0.298**

Table 1. Ablation study on applying ReStyle to pSp using our simpler encoder architecture variants compared to the original FPN-based architecture. All results shown are computed on the final reconstruction outputs after 5 inference steps. As shown, our simplified architecture is comparable to the FPN-variant with a reduced inference time.

by first inverting the real image into the FFHQ StyleGAN latent space. The resulting inverted code and reconstructed image are then used to initialize the ReStyle toonify encoder. Thanks to the improved initialization from the inverted latent code, the toonify encoder is able to learn a more faithful translation of the real image into its corresponding toon image. However, as mentioned in the main text, it is not trivial to assume that the real inverted code in the FFHQ latent space corresponds to the same identity characteristics in the toonify latent space.

To show that the above is in fact true we randomly sample $\mathbf{w}$ vectors from the FFHQ latent space. We then pass the same $\mathbf{w}$ vector to both the FFHQ StyleGAN generator and toonify StyleGAN generator to see if the synthesized images are semantically similar. We provide several examples in Figure 5. As shown, the same latent code produces semantically similar images in both latent spaces, indicating that the two latent spaces are indeed well-aligned. As a result of the above, we gain valuable insights as to the effectiveness of the encoder bootstrapping technique in the task of image toonification.

5. Quantitative Results

In Section 5.2, we quantitatively compared various inversion methods by constructing *quality-time graphs* which allowed us to visualize how reconstruction quality changes with respect to inference time. In Figures 6 and 7 we provide the quality-time graphs for both the L_2 and LPIPS loss metrics across all five domains. For the human facial domain we additionally measure the identity similarity of the reconstructed images by using the Curricularface [3] method for facial recognition.

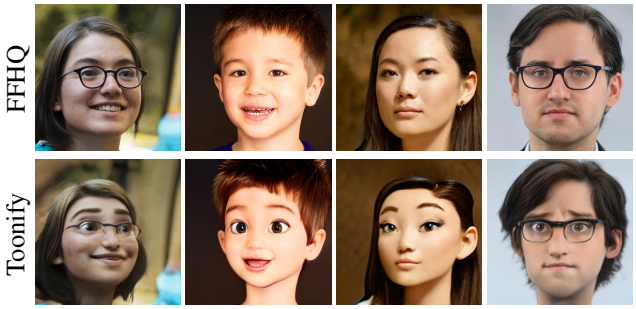

Figure 5. Synthesized images generated by passing the same randomly sampled latent code throug the FFHQ StyleGAN generator and Toonify StyleGAN generator. As shown, the two latent spaces are well-aligned.

6. Additional Results

The remainder of this document contains additional comparisons and results, as follows:

1. Figures 8 and 9 contain additional comparisons on the human facial domain. Additionally, Figure 10 provides a comparison on more challenging input poses and expressions.

2. Figures 11 and 12 contain additional comparisons on the cars domain.

3. Figures 13 and 14 contain additional comparisons on the churches domain.

4. Figure 15 contains additional comparisons on the wild animals domain.

5. Figure 16 contains additional comparisons on the horses domain.

6. Figure 17 contains comparisons to the IDInvert encoder from Zhu *et al*. [15] on the human facial and cars domains.

7. Figure 18 shows iterative outputs generated by ReStyle applied over pSp [10] on the human facial domain.

8. Figure 19 shows iterative outputs generated by ReStyle applied over e4e [13] on the cars domain.

9. Figure 20 shows iterative outputs generated by ReStyle applied over pSp [13] on the churches domain.

10. Figure 21 shows iterative outputs generated by ReStyle applied over e4e [13] on the horses domain.

11. Figures 22 contains editing comparisons with the optimization technique from Karras *et al*. [6] on the human facial domain obtained using InterFaceGAN [11].

12. Figures 23 contains editing comparisons with the optimization technique from Karras *et al*. [6] on the cars domain obtained using GANSpace [2].

13. Figures 24 contains editing comparisons with the optimization technique from Karras *et al*. [6] on the horses domain obtained using SeFa [12].

14. Figures 25 and 26 contain additional results on the image toonification task using the encoder bootstrapping method presented in Section 5.5.

Note, all results are shown at full-resolution: 1024×1024 for human faces, 512×512 for cars and wild animal faces, and 256×256 for churches and horses.

References

[1] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains, 2020. 1, 2, 15

[2] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *arXiv preprint arXiv:2004.02546*, 2020. 5, 23

[3] Yuge Huang, Yuhan Wang, Ying Tai, Xiaoming Liu, Pengcheng Shen, Shaoxin Li, Jilin Li, and Feiyue Huang. Curricularface: adaptive curriculum learning loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5901–5910, 2020. 4

[4] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 1, 8, 9, 10, 17

[5] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 1, 2

[6] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020. 4, 5, 22, 23, 24

[7] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 1, 11, 12, 17

[8] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild, 2015. 1, 8, 9, 10, 17

[9] Justin N. M. Pinkney and Doron Adler. Resolution dependent gan interpolation for controllable image synthesis between domains, 2020. 3

[10] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2, 3, 4, 8, 10, 11, 13, 15

[11] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9243–9252, 2020. 4, 22, 24

[12] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. *arXiv preprint arXiv:2007.06600*, 2020. 5

[13] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation, 2021. 1, 2, 4, 9, 12, 14, 16

[14] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop, 2016. 1, 7, 13, 14, 16

[15] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. *arXiv preprint arXiv:2004.00049*, 2020. 2, 4, 6, 17

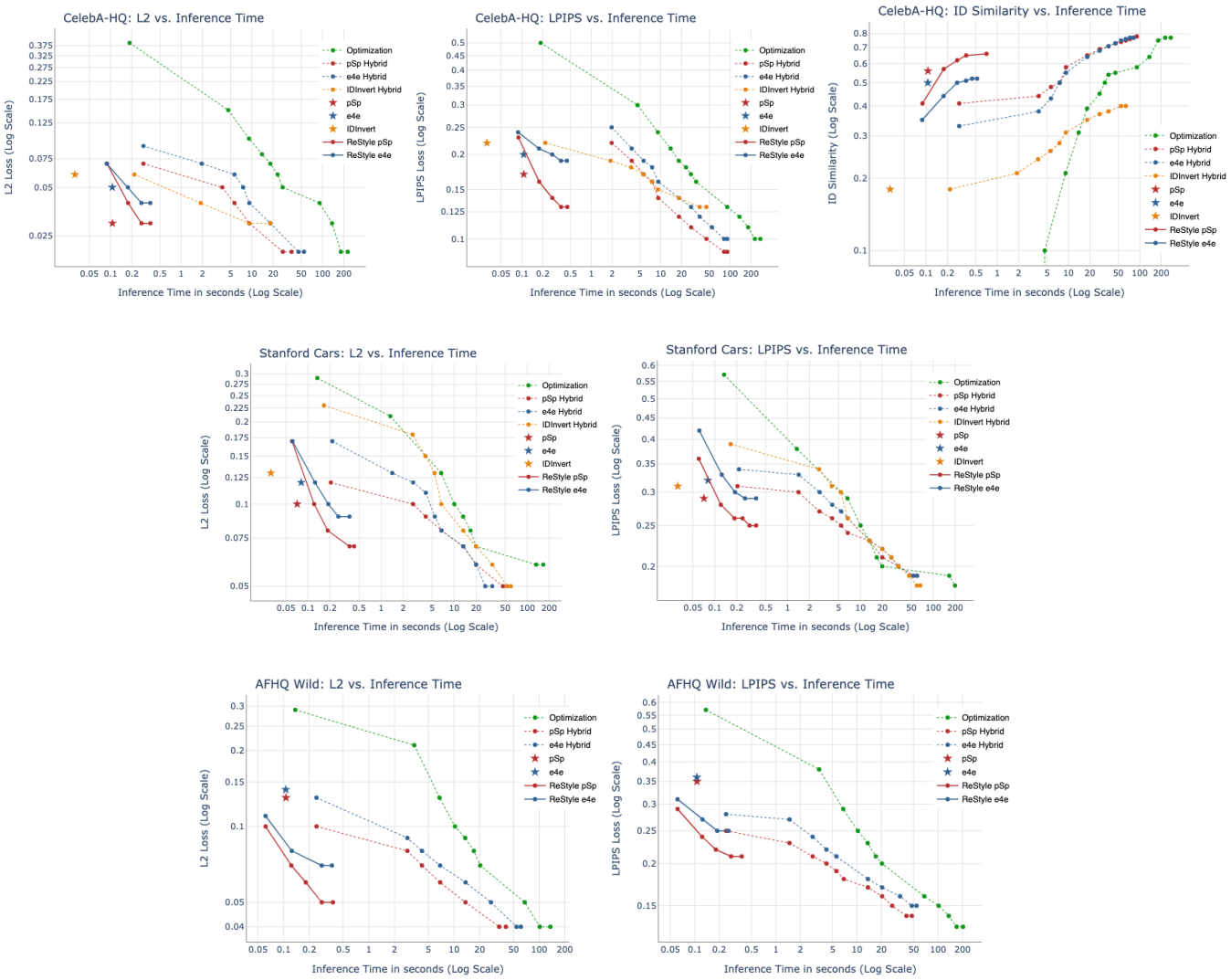

Figure 6. *Quantitative comparison.* We compare ReStyle with current state-of-the-art optimization-based and encoder-based methods by analyzing reconstruction via multiple evaluation metrics while measuring each method's run-time during inference. Each encoder-based method is represented using a $\star$ symbol. The corresponding hybrid method is marked using a dashed line of the same color with the ReStyle applied over the base method shown using a solid line of the same color. Optimization results are shown using a dashed green line. Methods based on pSp are shown in red with methods based on e4e shown in blue. Finally, results obtained using IDInvert [15] are shown in orange. Note that both axes are shown in log-scale.

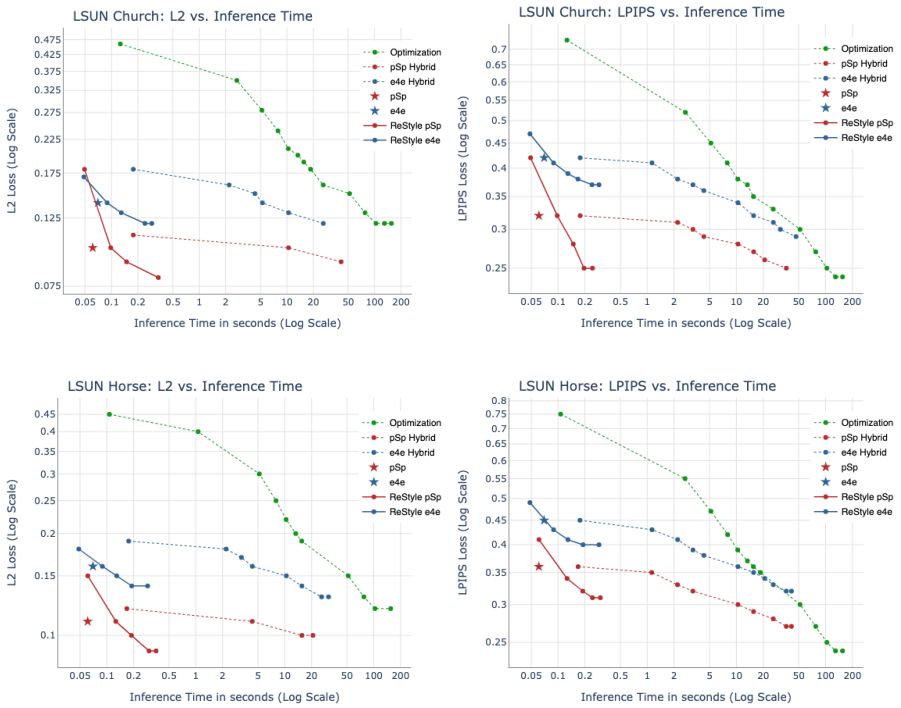

Figure 7. *Quantitative comparison.* Same setting as Figure 6 for the LSUN [14] Church and LSUN Horse domains.

Figure 8. Additional qualitative comparisons on the CelebA-HQ [8, 4] test set. Here, we apply ReStyle over the pSp encoder [10].

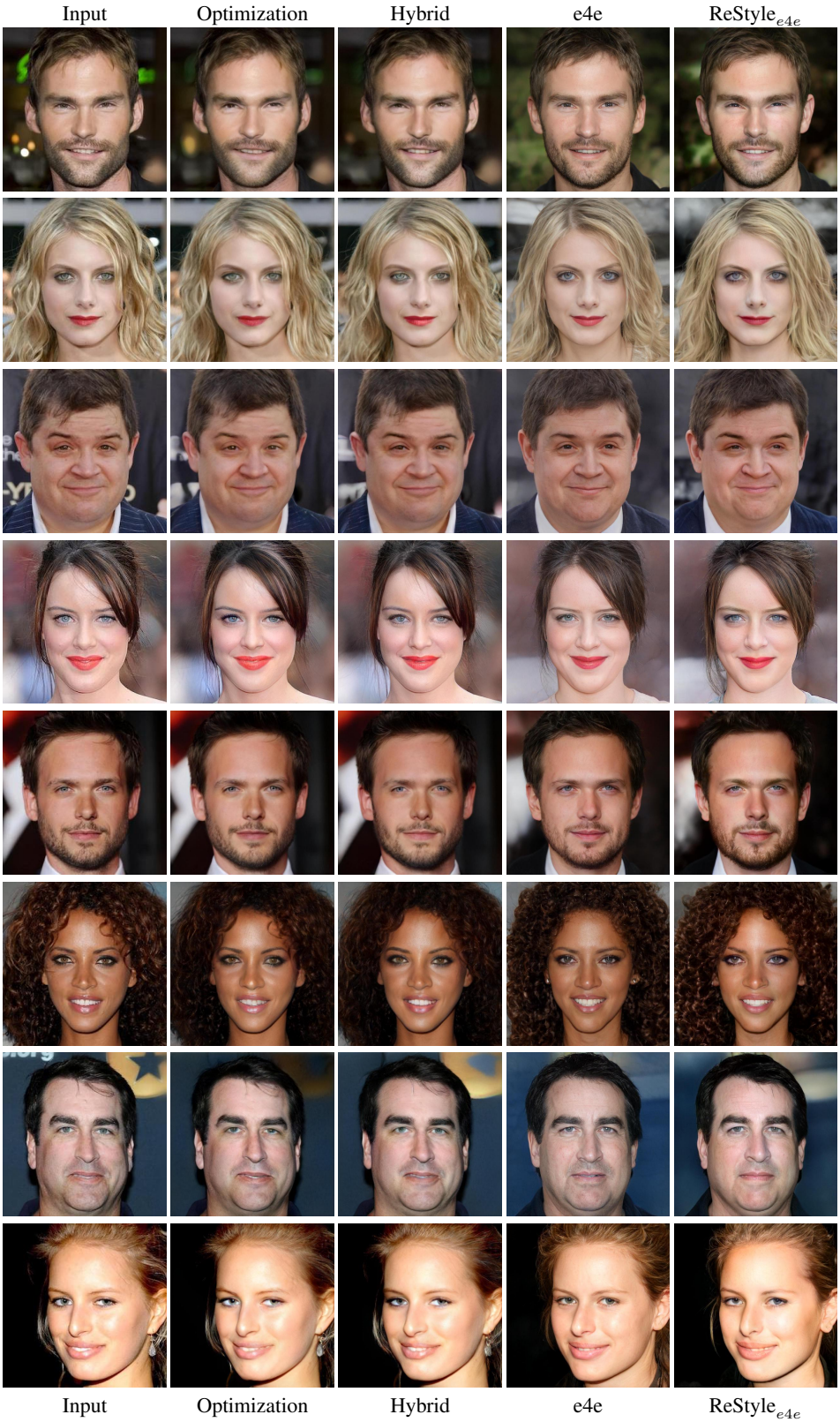

Figure 9. Additional qualitative comparisons on the CelebA-HQ [8, 4] test set. Here, we apply ReStyle over the e4e encoder [13].

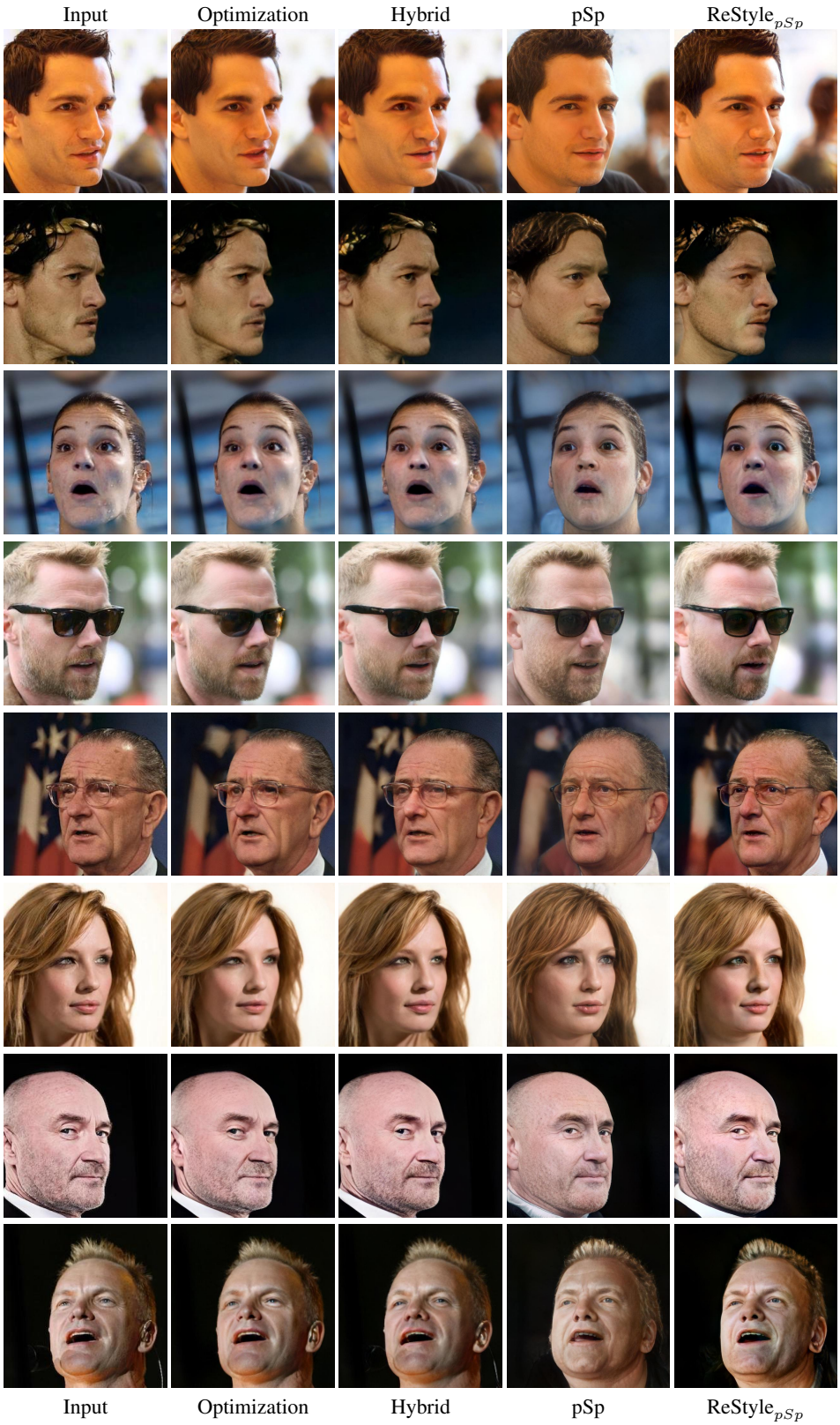

Figure 10. Additional qualitative comparisons on the CelebA-HQ [8, 4] test set on more challenging input poses and expressions. Here, we apply ReStyle over the pSp encoder [10].

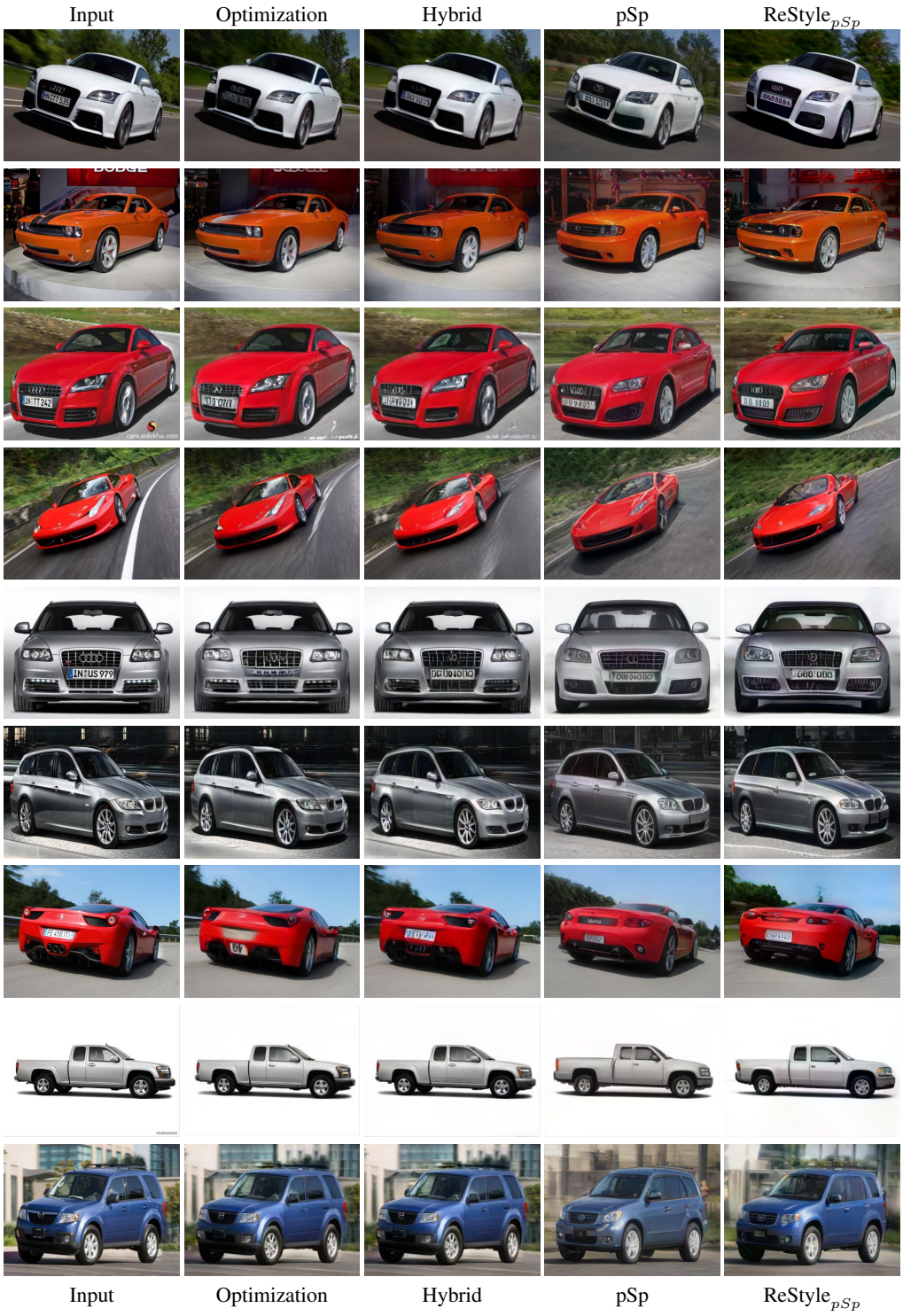

Figure 11. Additional qualitative comparisons on the Stanford Cars [7] test set. Here, we apply ReStyle over the pSp encoder [10].

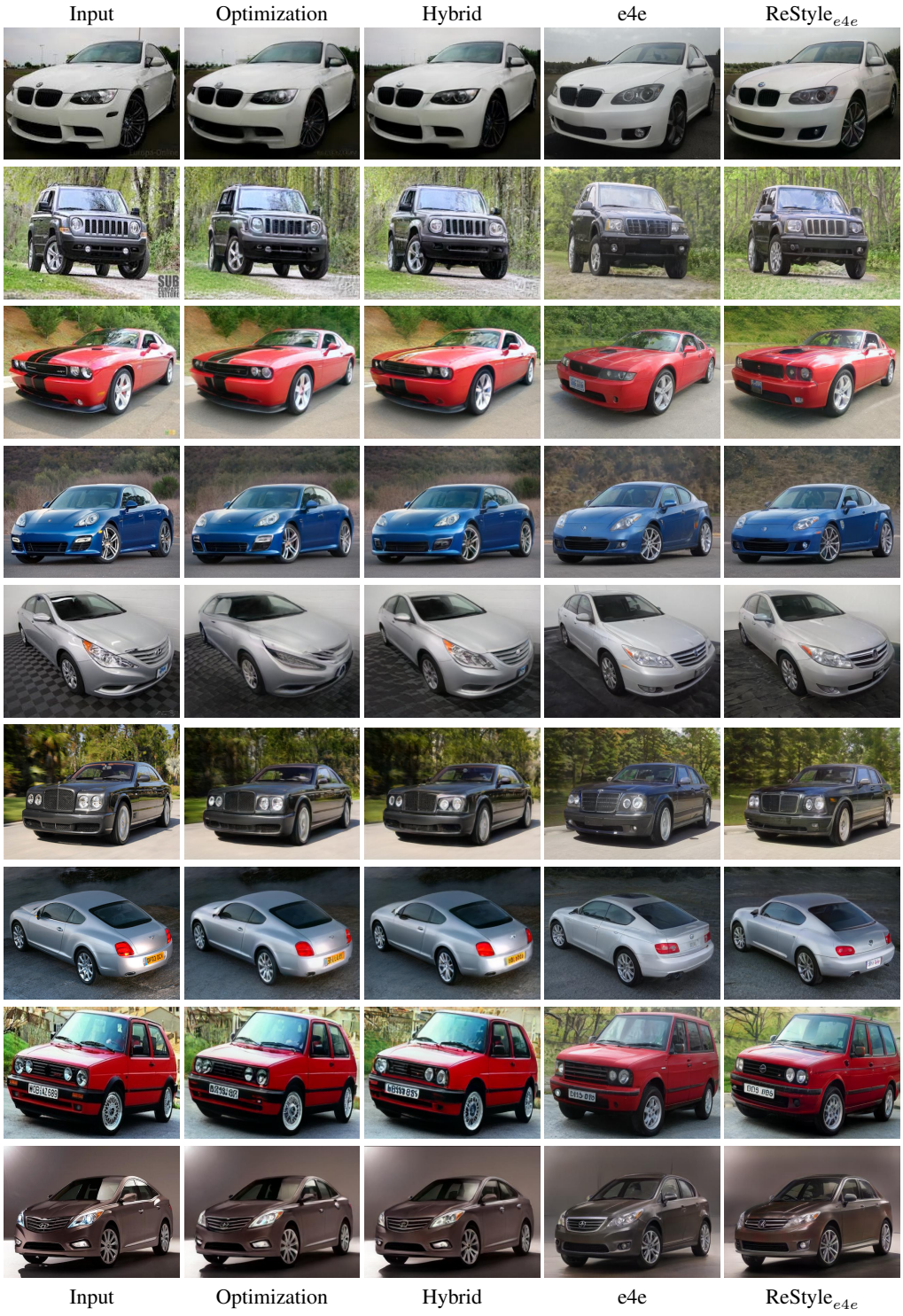

Figure 12. Additional qualitative comparisons on the Stanford Cars [7] test set. Here, we apply ReStyle over the e4e encoder [13].

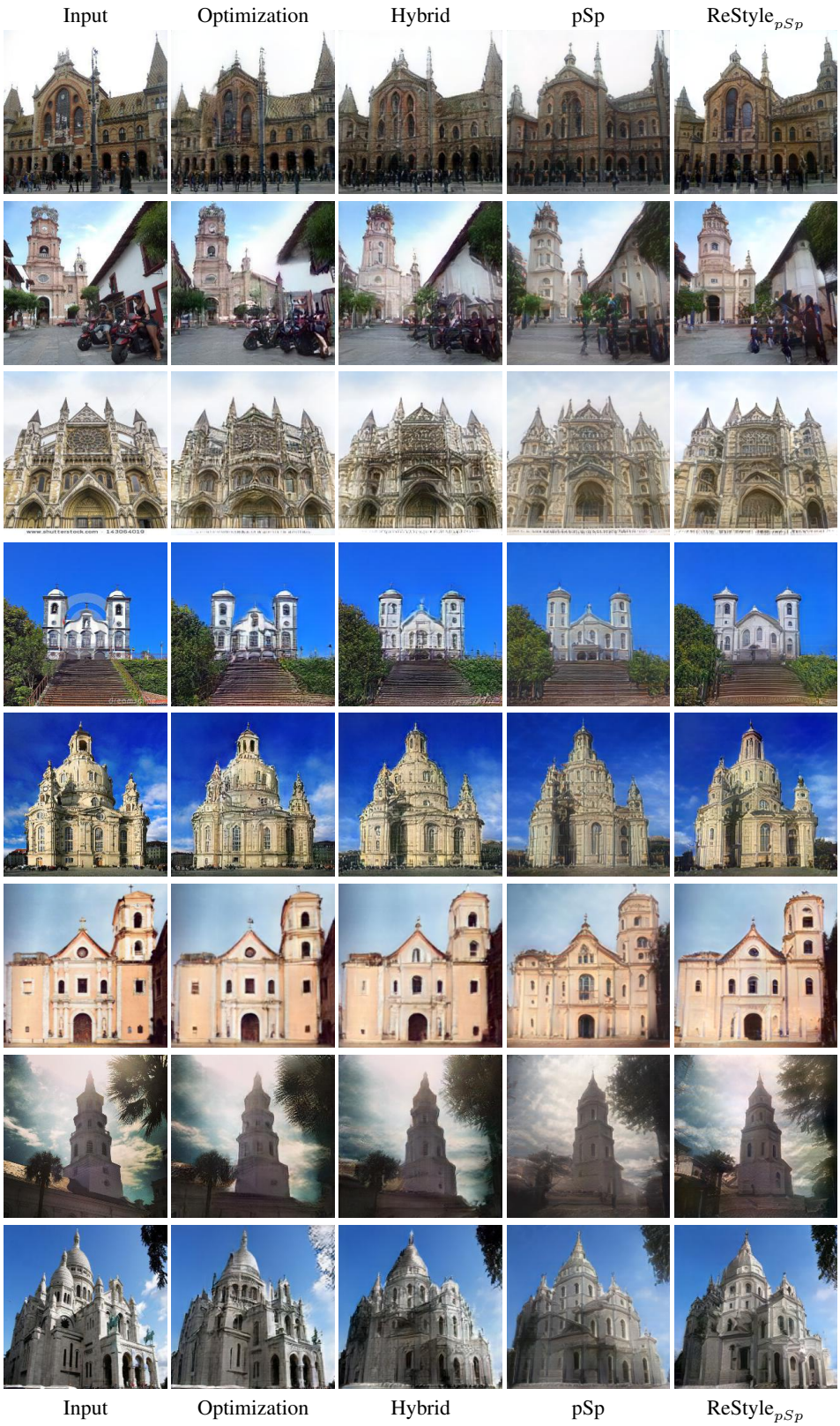

Figure 13. Additional qualitative comparisons on the LSUN [14] Church test set. Here, we apply ReStyle over the pSp encoder [10].

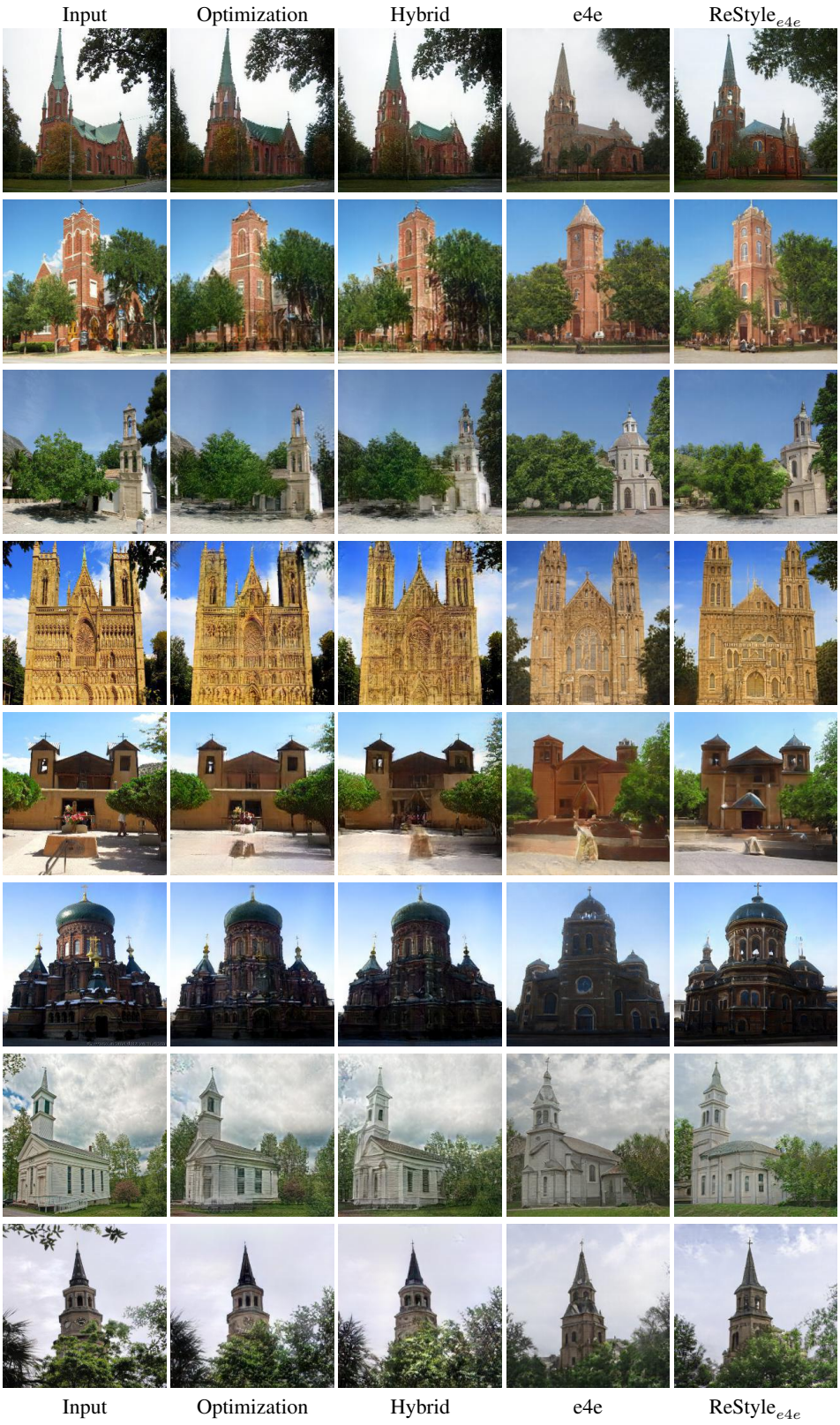

Figure 14. Additional qualitative comparisons on the LSUN [14] Church test set. Here, we apply ReStyle over the e4e encoder [13].

Figure 15. Additional qualitative comparisons on the AFHQ [1] Wild test set. Here, we apply ReStyle over the pSp encoder [10].

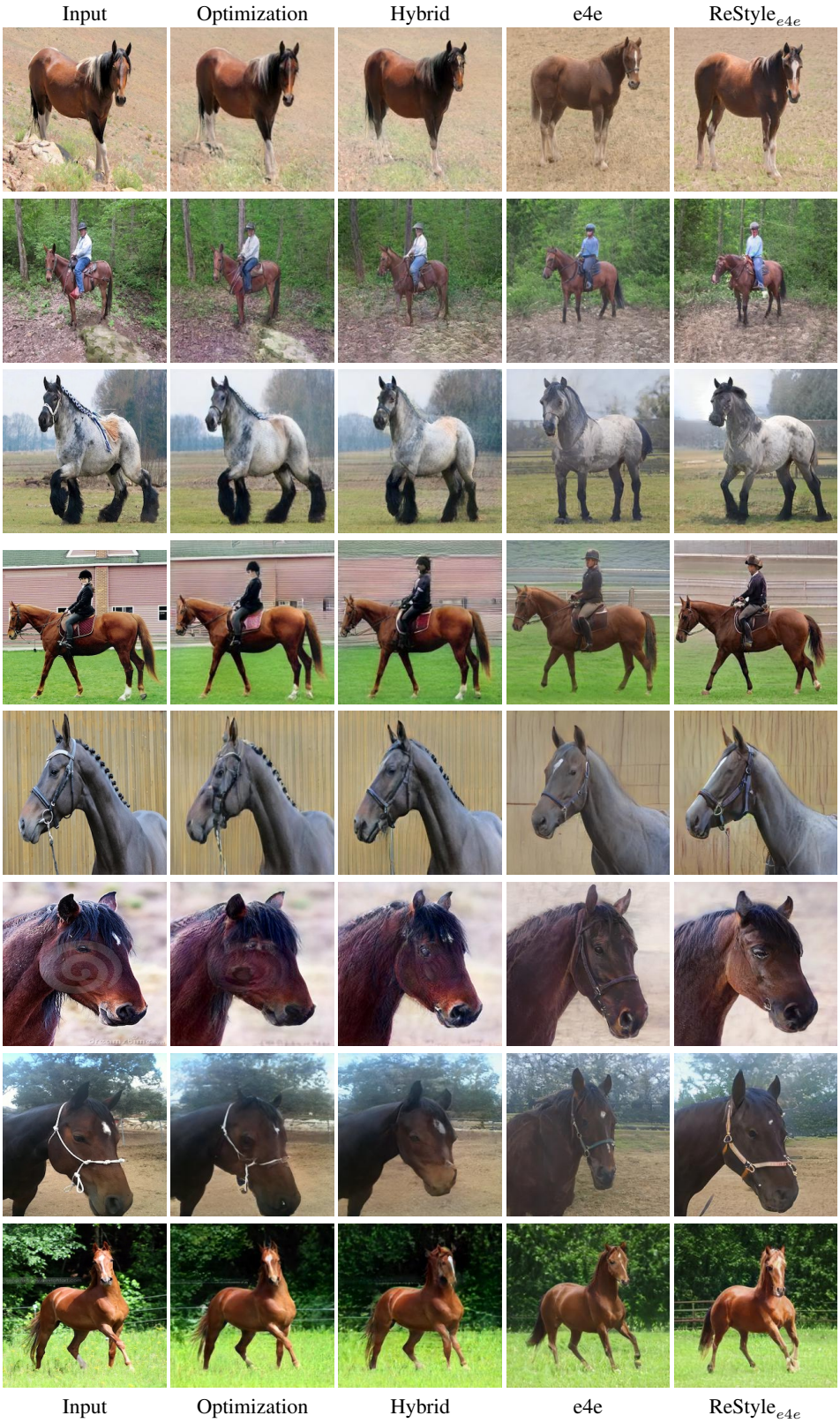

Figure 16. Additional qualitative comparisons on the LSUN [14] Horse test set. Here, we apply ReStyle over the e4e encoder [13].

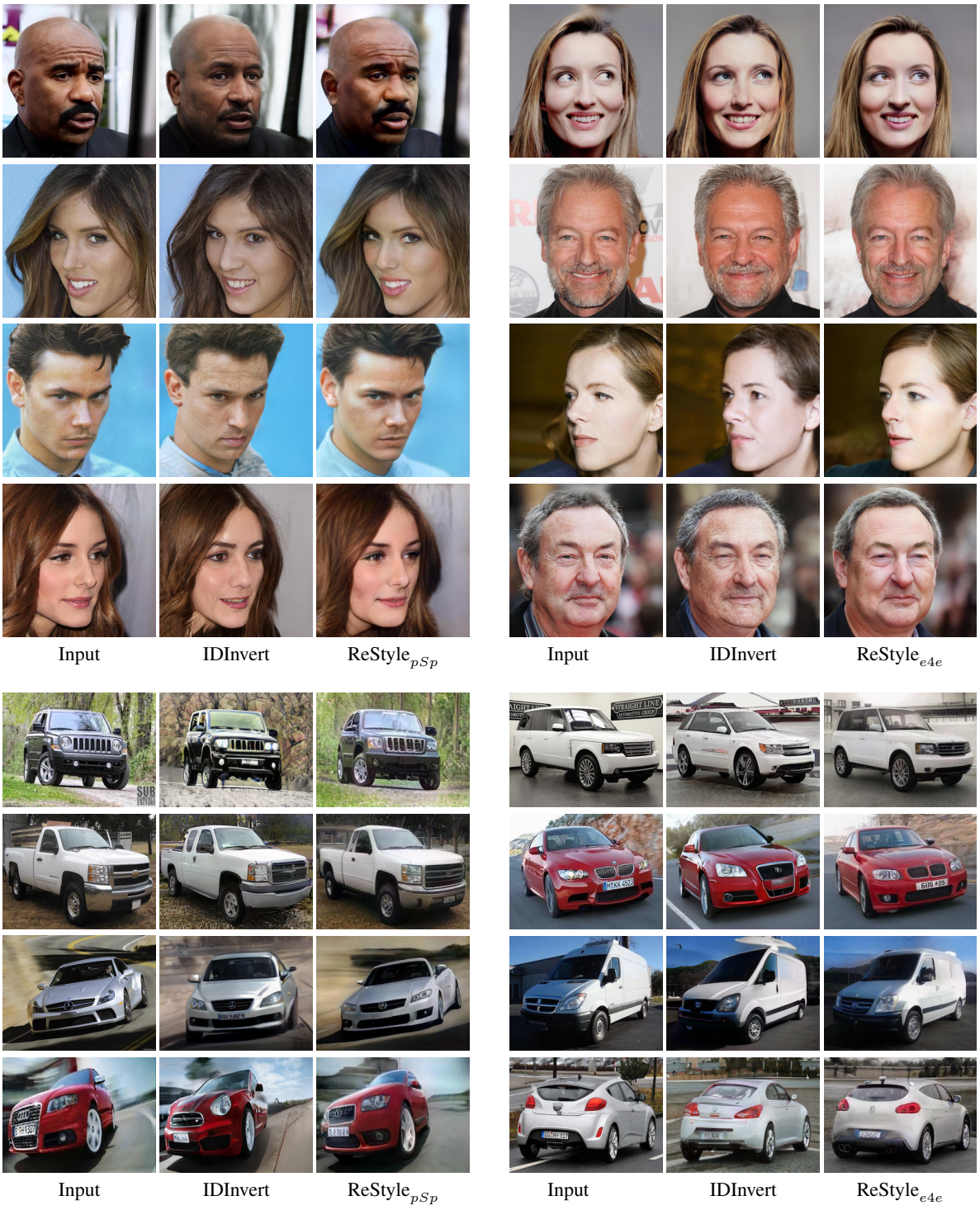

Figure 17. Qualitative comparisons with the IDInvert encoder from Zhu *et al*. [15] on the CelebA-HQ [8, 4] and Stanford Cars [7] test sets. Here, we show results with ReStyle applied over both the pSp and e4e encoders.

Figure 18. Iterative outputs generated by ReStyle applied over pSp on the human facial domain.

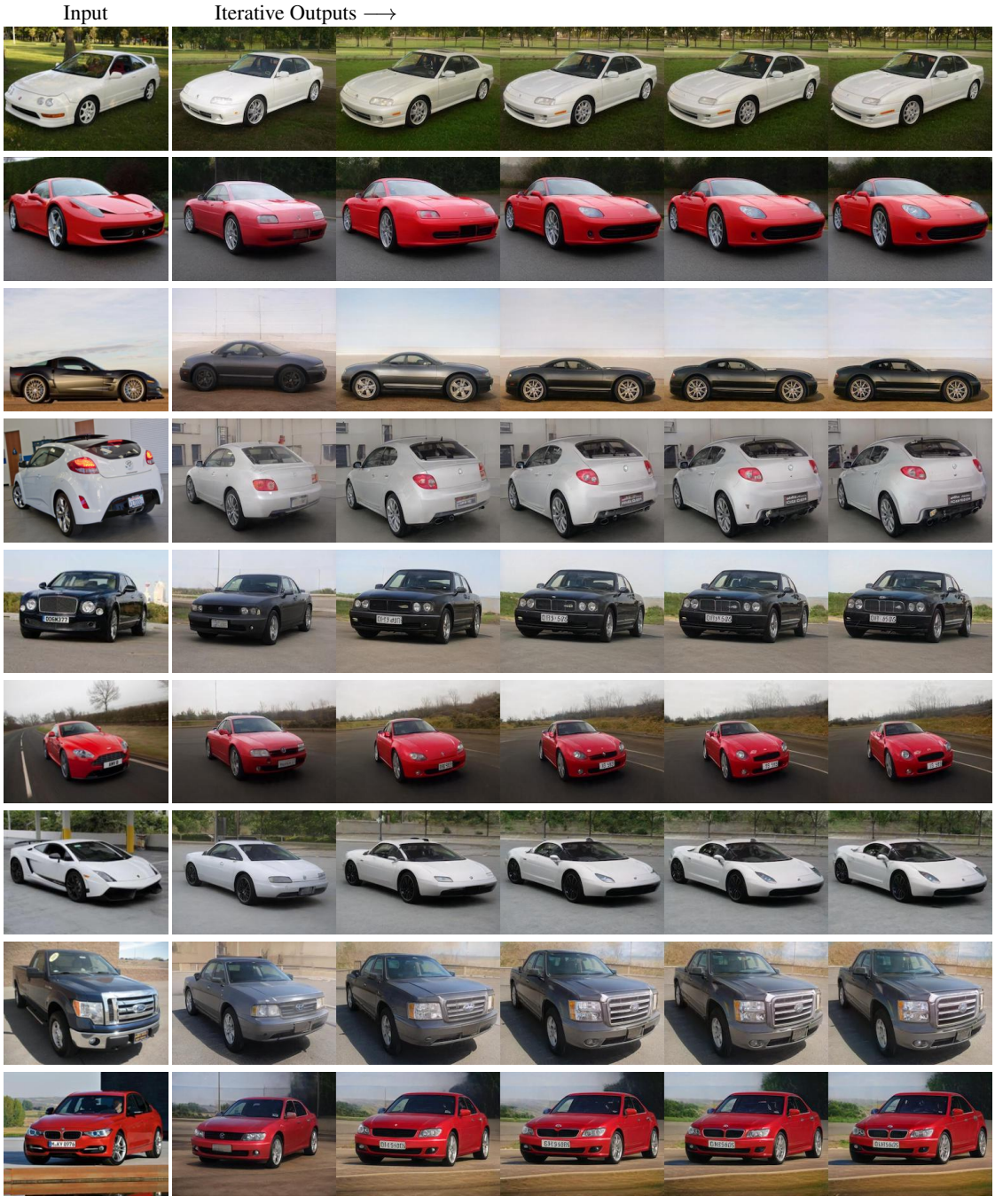

Figure 19. Iterative outputs generated by ReStyle applied over e4e on the cars domain.

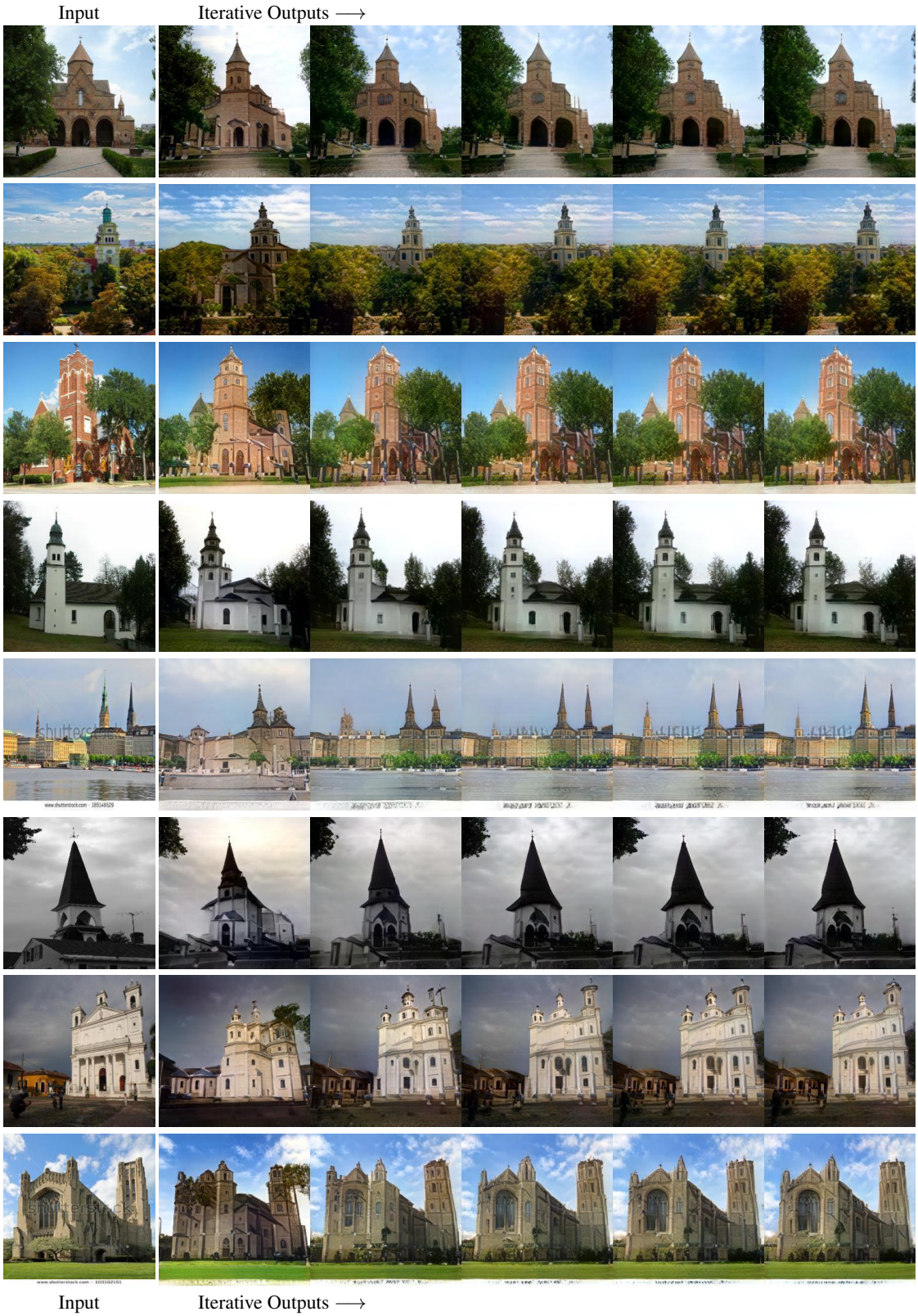

Figure 20. Iterative outputs generated by ReStyle applied over pSp on the churches domain.

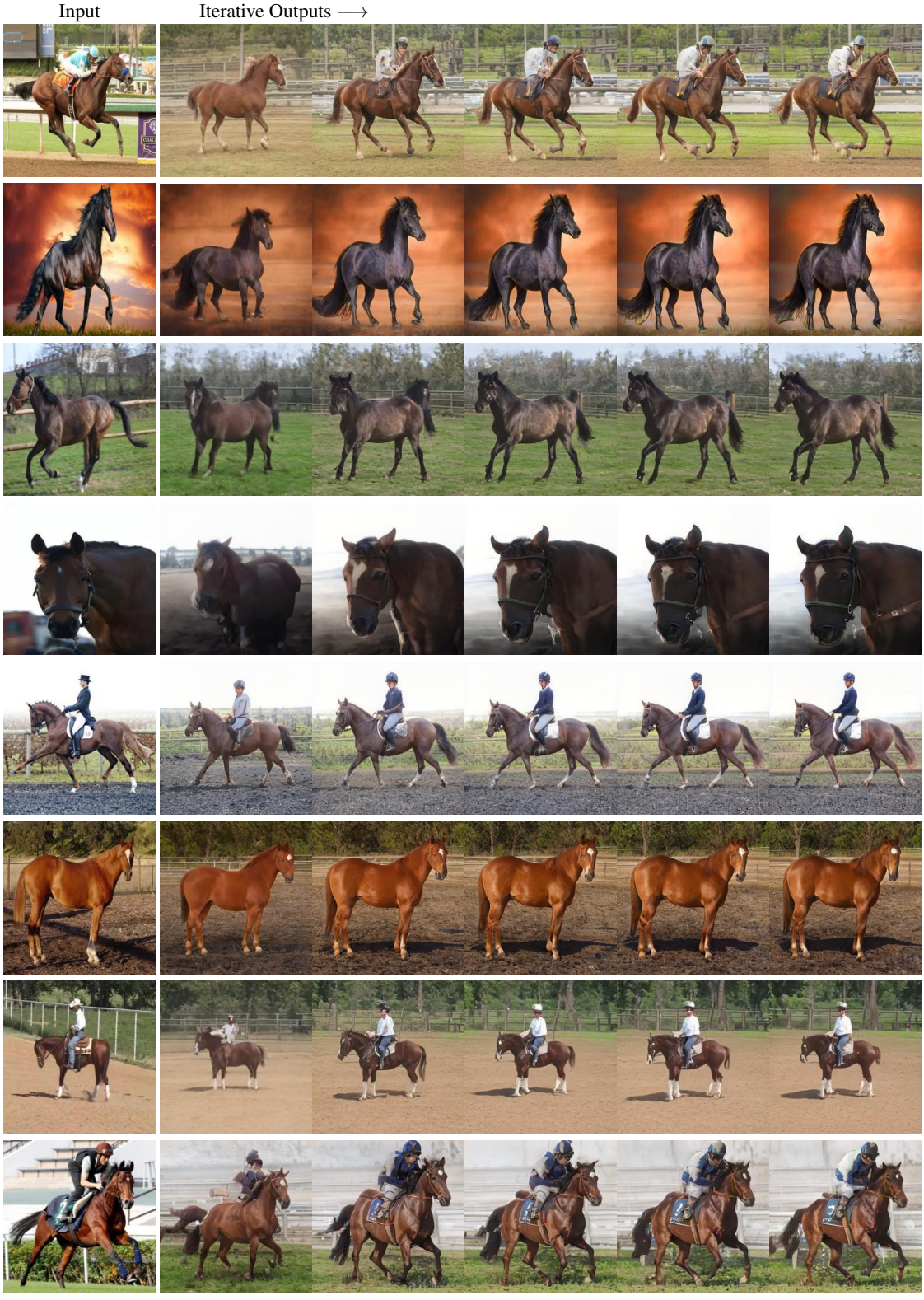

Figure 21. Iterative outputs generated by ReStyle applied over e4e on the horses domain.

Figure 22. Editing comparisons with the optimization technique from Karras *et al*. [6] on the human facial domain obtained using InterFaceGAN [11]. Here ReStyle is applied over the e4e encoder. We perform two edits: an age edit and smile edit.

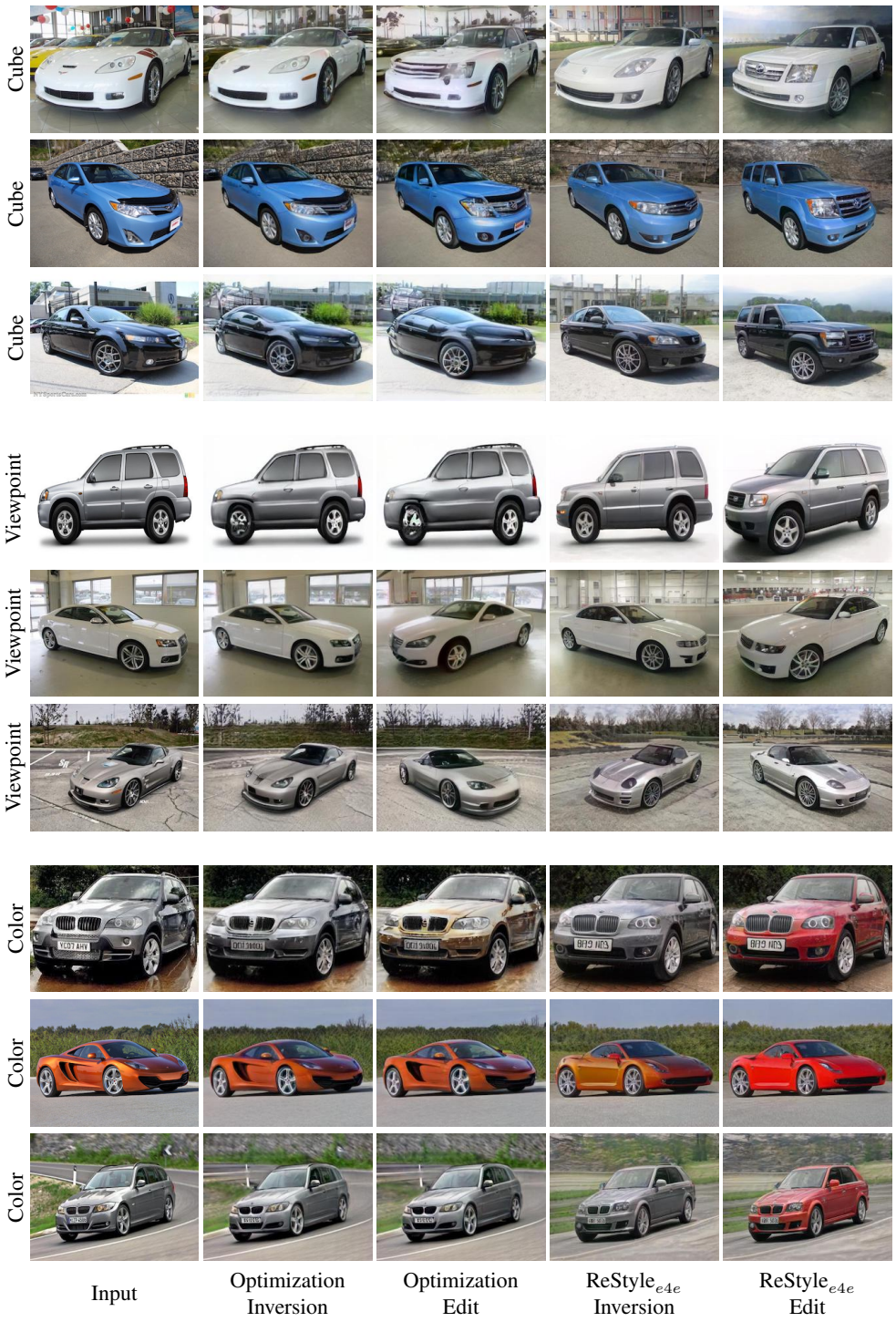

Figure 23. Editing comparisons with the optimization technique from Karras *et al*. [6] on the cars domain obtained using GANSpace [2]. Here ReStyle is applied over the e4e encoder. We perform three edits: a cube-shape edit, viewpoint edit, and color edit.

Figure 24. Editing comparisons with the optimization technique from Karras *et al*. [6] on the horses domain obtained using SeFa [11]. Here ReStyle is applied over the e4e encoder. We perform two edits: a pose edit and an edit to add/remove a horse rider.

Figure 25. Additional results on the image toonification task using ReStyle with our encoder bootstrapping technique. For each input, we show the inverted image obtained after a single step of our ReStyle$_{pSp}$ FFHQ encoder followed by the iterative outputs of our ReStyle$_{pSp}$ toonify encoder.

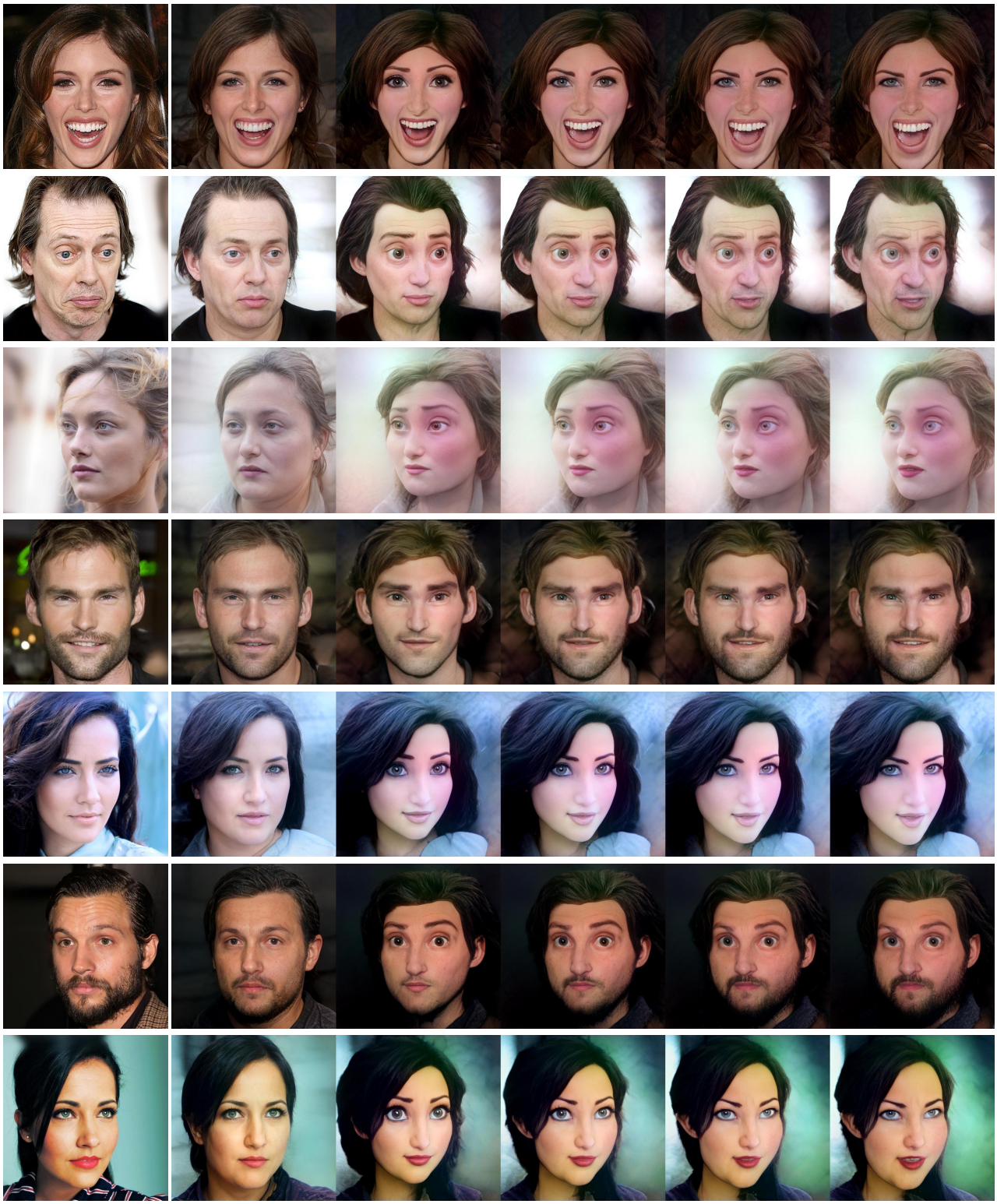

Figure 26. Additional image toonification results obtained using the same setting as Figure 25.