Supplementary Material - Adaptive Convolutions for Structure-Aware Style Transfer

Prashanth Chandran[1,2] Gaspard Zoss[1,2] Paulo Gotardo[1] Markus Gross[1,2] Derek Bradley[1]

1) DisneyResearch|Studios, Zurich
2) Department of Computer Science, ETH Zurich
{chandrap,gaspard.zoss,grossm}@inf.ethz.ch, {paulo.gotardo,derek.bradley}@disneyresearch.com

1. Other Results: Multiscale Style Transfer

As described in the main text, Sec. 3.2, the predicted kernels and biases in AdaConv are applied at every resolution of the decoder independently. This means that it is easy to mix styles at different scales during decoding, for example use the kernels predicted from one style for the coarse scale layers, and the kernels predicted from a different style for the fine scale layers. We illustrate such an application in Fig. 1, showing both the complete style-transferred results for two different styles independently, and in the center of the figure we show the combined multiscale style transfer result where the coarse and fine scale styles come from the different style images. The content image is that on the first row of Fig. 5, main text.

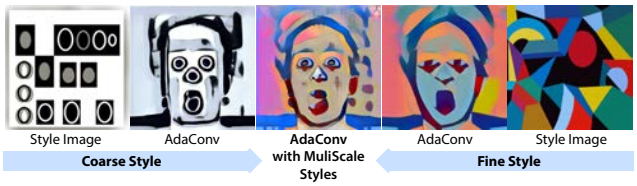

Figure 1: AdaConv allows users to combine different styles at different resolutions to achieve interesting artistic effects.

2. Ablation Studies

In this supplemental section, we present supporting experiments and ablation studies that validate some of our design choices.

2.1. Comparisons to multi-scale AdaIN

As we show in Fig. 2 of the main paper, our decoder applies the predicted kernels at multiple scales. This is different from the simple decoder of AdaIN that transfers feature statistics only at the lowest scale. We modify the standard AdaIN decoder such that feature statistics are transferred at multiple scales of the decoder, similar to our decoder. We refer to this variant of AdaIn for style transfer as *AdaIN-Multiscale*. We also modify our multiscale AdaConv decoder such that it applies the predicted kernels only at the lowest spatial resolution. We refer to this variant of our architecture that performs adaptive convolutions only at the lowest spatial resolution as simply *AdaConv-Singlescale*. We train the newly defined modules *AdaIn-Multiscale*, *AdaConv-Singlescale* similarly to AdaIN and AdaConv respectively. In Fig. 2, we qualitatively compare the style transfers resulting from the 4 methods.

2.2. Effective of style dimension and groups across scales

In Fig. 3, we show how changing the number of groups in our decoder, and the style dimension s_d can affect the resulting style transfer.

2.3. Effect of kernel size

The predicted kernel size is a hyper-parameter in our method that is set by the user at design time. The kernel size has an interesting effect on the result of the style transfer. To demonstrate this qualitatively, we train a dedicated decoder, each with $s_d = 64$ and predict kernels of varying sizes $\{1, 3, 5, 7\}$. We present qualitative results in Fig. 4. As one would expect, increasing the size of the predicted kernel, distorts the content further. Using smaller kernels preserves the contours on the content image better.

2.4. Effect of normalization

AdaIN normalizes each channel in the content features before transferring the feature statistics from the style image. In our method, since we predict *depthwise-separable*

[1]Portions of this figure are © 2017 IEEE. Reprinted, with permission, from Huang and Belongie [1].

[2]Portions of this figure from Jing et al. [2] are © 2020, Association for the Advancement of Artificial Intelligence. All rights reserved. Permission to reuse the figure for other purposes (or granting it to others) is NOT allowed.

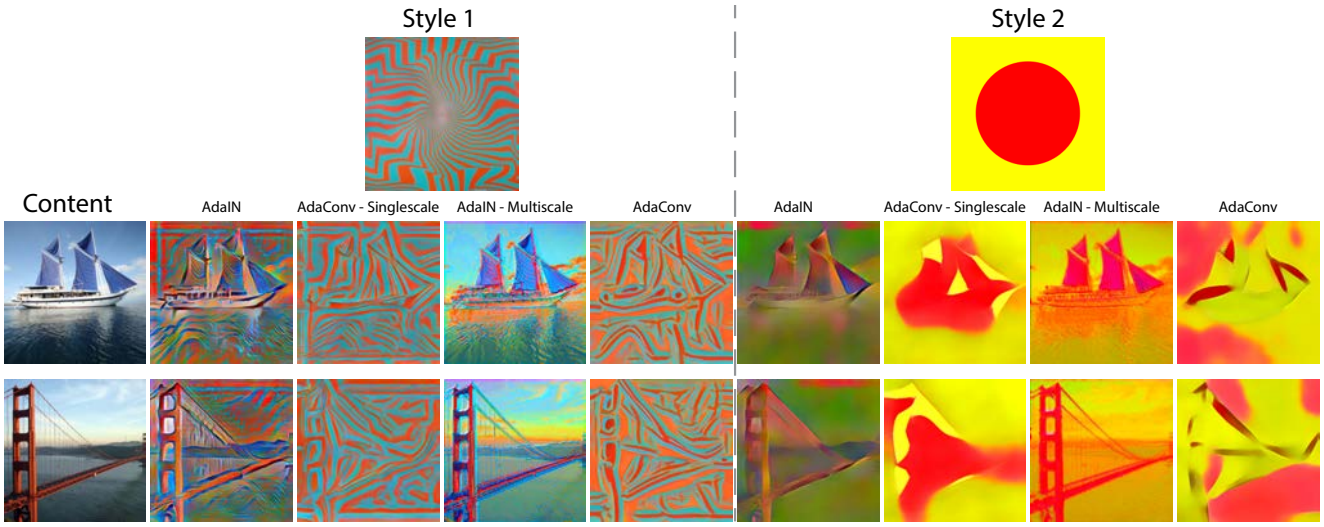

Figure 2: We qualitatively compare AdaIN, *AdaConv-Singlescale* (which applies adaptive convolutions only at the lowest scale), *AdaIN-Multiscale* (the modified decoder for AdaIN where statistics are transferred at all scales of the decoder) and AdaConv (our decoder which applies adaptive convolutions at all scales). For this experiment, both AdaConv models use kernel predictors that predict kernels of size 3x3 and $s_d = 64$. We see that the AdaConv models can result in style transfers with less artifacts and can capture the structural properties of the style image, while the multiscale AdaIN variant seems to remain faithful to the content the most[1].

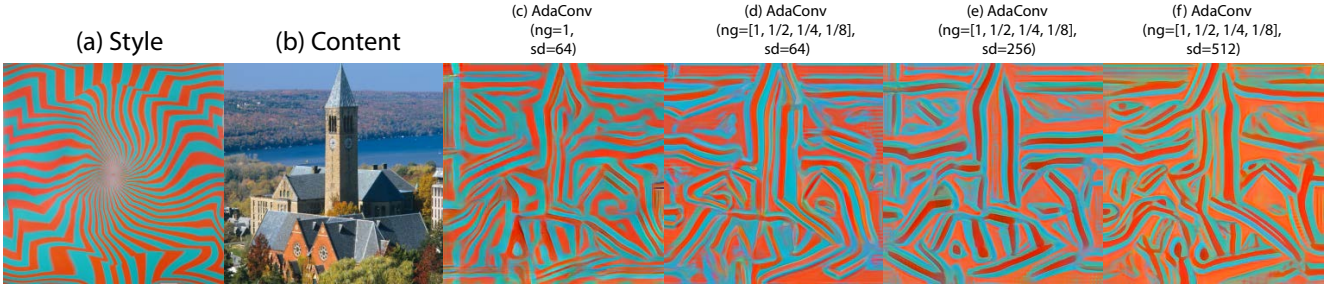

Figure 3: (a) The input style image (b) the content image (c) AdaConv decoder trained with $n_g = 1$ at all scales and $s_d = 64$. (d-f) AdaConv decoder trained with an decreasing number of groups $n_g = \{1, 1/2, 1/4, 1/8\}$ at each scale and $s_d = 64$, 256, and 512 respectively. Although qualitatively, the results look similar for each of these configurations, balancing the decoder's capacity across multiple scales by varying the groups helps with stabilizing the training[1].

kernels that convolve with the content features directly, we found that the explicit normalization of the content features is not necessary. In Fig. 5, we show multiple examples of style transfers where we apply AdaConv with and without normalizing the input features. For this purpose, two separate models were trained. Though qualitatively the results look similar, we find that normalizing the input features before applying AdaConv helps with training stability and aids convergence.

2.5. Weighting the style loss

Since AdaConv directly manipulates both the statistical and structural properties of the content image, it can result in style transfers that prefer the style image structure much more than AdaIN. We would like to note that even after explicitly increasing the weight on the style loss, AdaIN cannot produce similar results. To illustrate this, we retrain both AdaConv and AdaIN by giving varying importance (λ_S) to the style loss $\{1, 10, 100\}$. The original implementation of AdaIN uses $\lambda_S = 10$. For this experiment, to remain fair to AdaIN, we apply the proposed adaptive convolutions only at the lowest scale of the decoder. The rest of the decoder for AdaConv is identical to the one used by AdaIN. As we see in Fig. 6, even with a high weight of 100 on the style loss, AdaIN cannot recreate the style structure in the final result as well as AdaConv . We further observe that a higher weight on the style for AdaConv

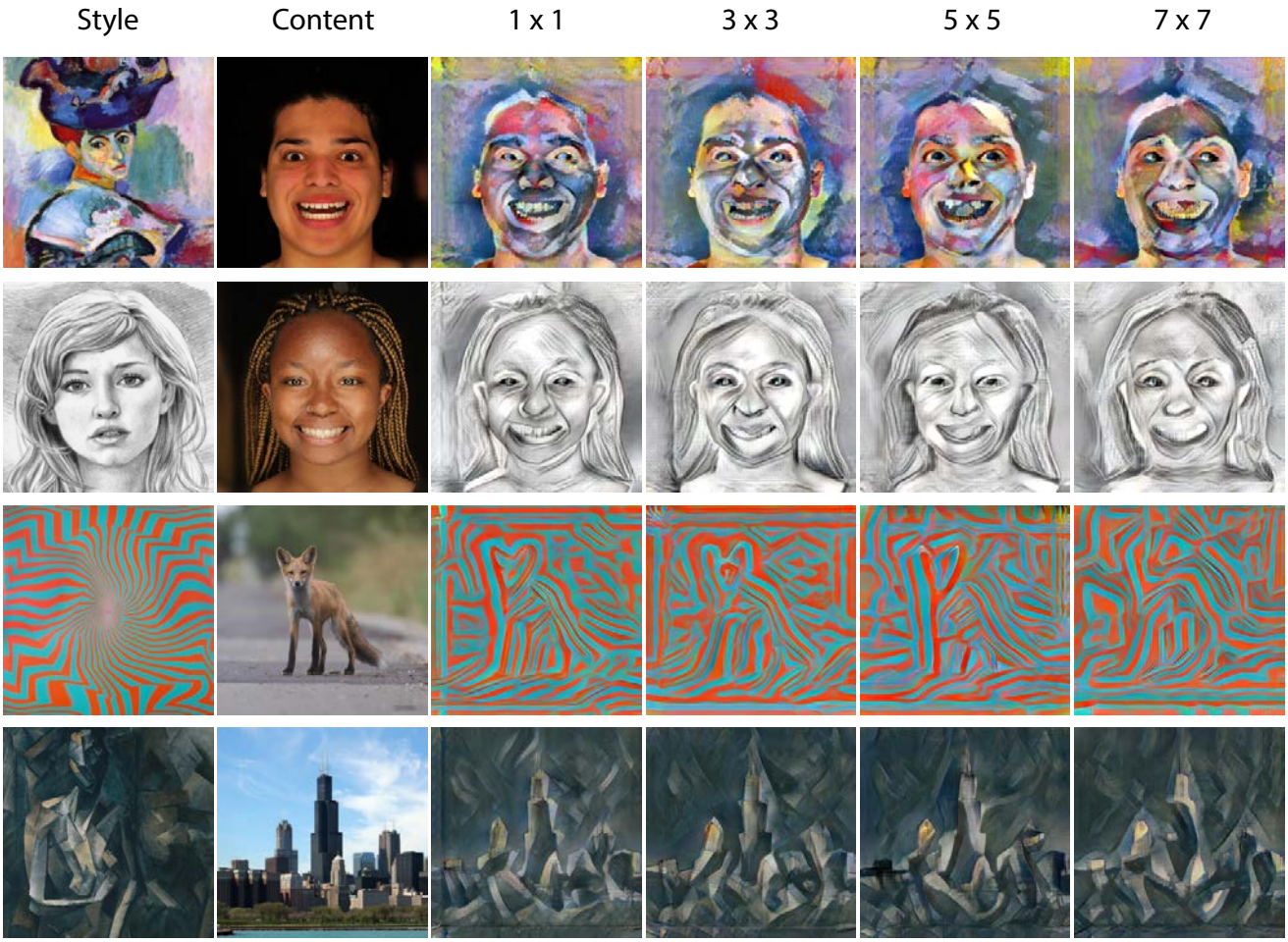

Figure 4: Predicting kernels of bigger sizes alters the structure of the content image more, while kernels of lower sizes preserve contours in the content better[1,2].

results in greater structural modifications to the content image. Therefore, AdaConv offers the possibility for users to choose a style weight that will yield the desired style transfer.

3. Additional results for generative models

3.1. Quantitative Results

In Tab. 1, we provide the FID metric of using AdaConv in styleGAN2's generator [3]. For these results, we train the modified generator from Fig. 9 of the main paper on single Nvidia2080Ti with a batch size of 4. The discriminator only saw a total of 1.2 million real images during training in our case compared to around 25 million images in [3]. Though our FID metrics are well below the state of the art on all the three datasets, our initial results are qualitatively promising and indicate that our method can be also used in generative settings too. We intend to continue our exploration of using AdaConv for such generative networks in the future.

Dataset	n_{iters}	FID
FFHQ (256 x 256)	300K	22.15
CelebHQ (256 x 256)	300K	25.12
AFHQ-Wild (256 x 256)	300K	10.69
AFHQ-Dog (256 x 256)	300K	18.27

Table 1: Preliminary FID metrics of using AdaConv instead of AdaIN in styleGAN2 like generative network.

3.2. Qualitative Results

In Fig. 7, Fig. 8, Fig. 9 and Fig. 10, we show synthetic images generated by using the AdaConv block in style-

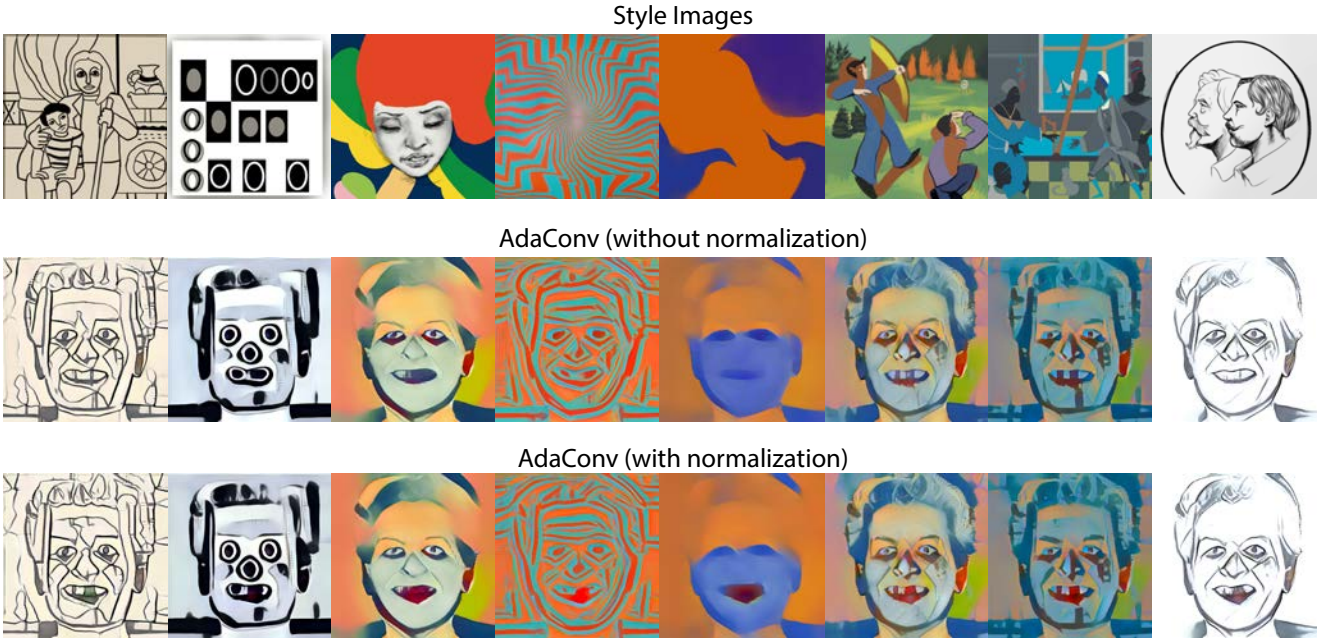

Figure 5: We observed that explicitly normalizing the content features isn't necessary for AdaConv although it helps with convergence. For this experiment, AdaConv uses kernel predictors that predict kernels of size 3x3 and $s_d = 64$.

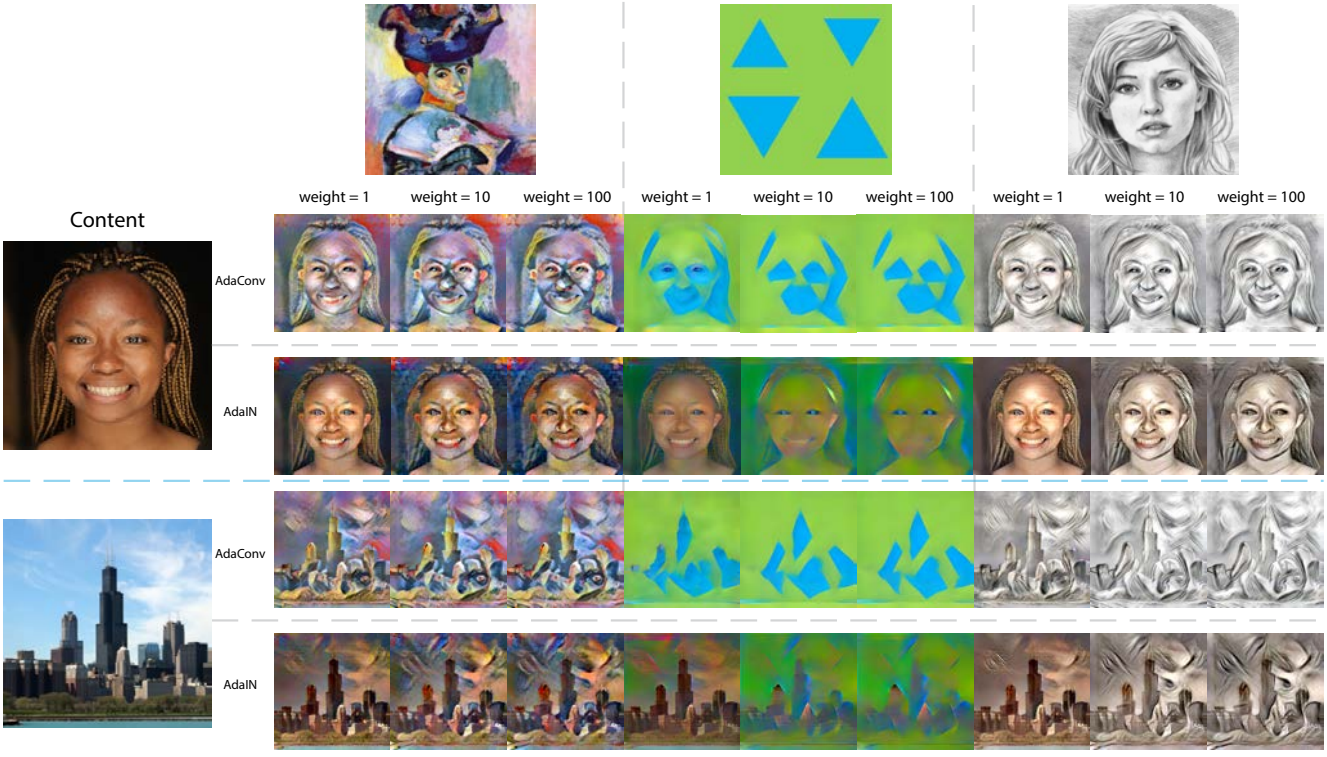

Figure 6: We show the effect of varying the weight of the style loss for both AdaConv and AdaIN. For AdaConv, we use a kernel size of 3x3 and $s_d = 64$. We can see that even with a high weight on the style, AdaIn is not able to produce structure-aware style transfer like AdaConv[1].

GAN2 as discussed in Sec. 4.2 of the main paper. All images shown in these figures were generated by our network and any likeness to persons living/dead is purely coincidental.

References

[1] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1

[2] Yongcheng Jing, Xiao Liu, Yukang Ding, Xinchao Wang, Errui Ding, Mingli Song, and Shilei Wen. Dynamic instance normalization for arbitrary style transfer. In *AAAI*, 2020. 1

[3] 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 (CVPR)*, June 2020. 3

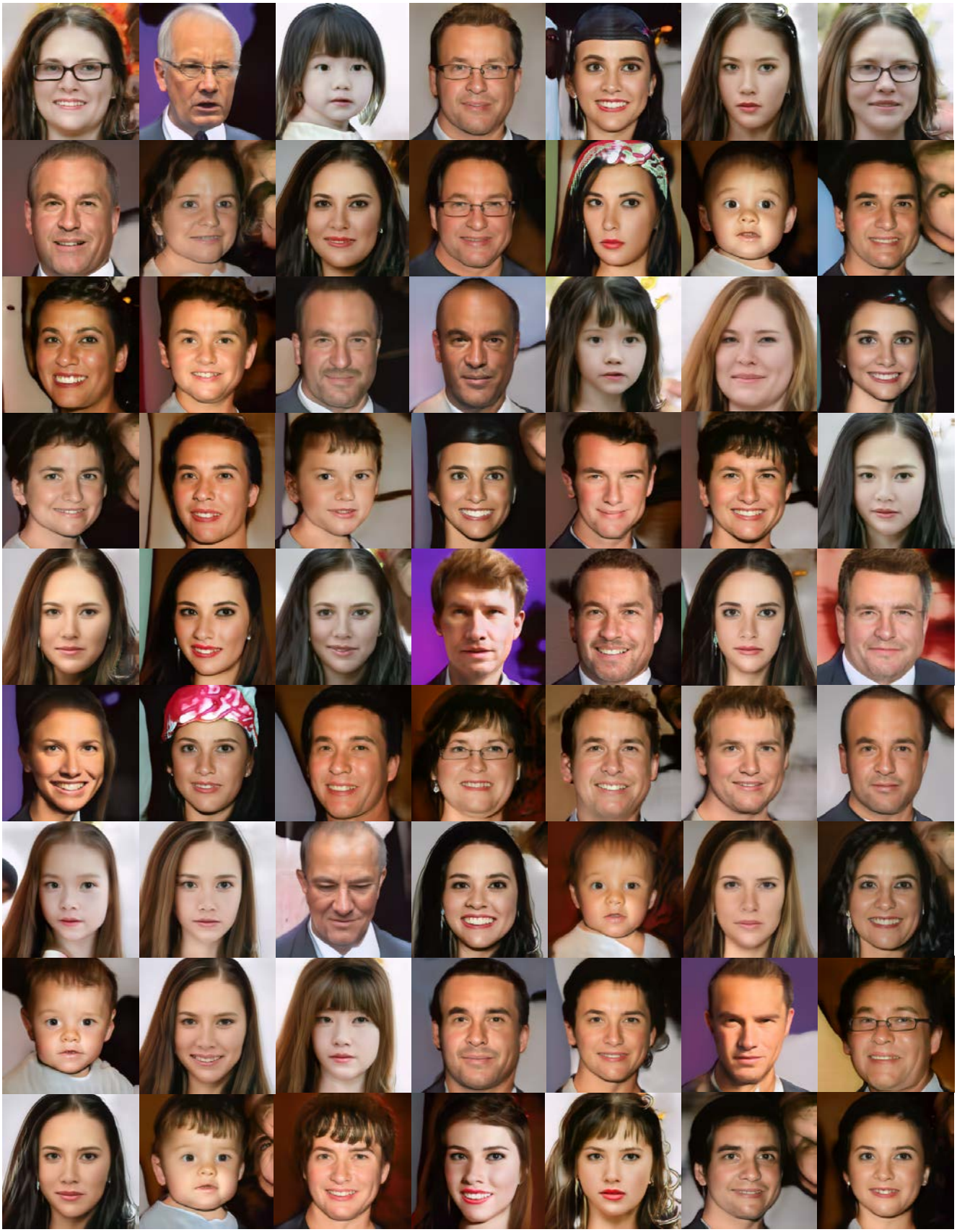

Figure 7: Uncurated generations ($\psi = 0.5$) on FFHQ dataset at 256 x 256

Figure 8: Uncurated generations ($\psi = 0.5$) on CelebHQ dataset at 256 x 256

Figure 9: Uncurated generations ($\psi = 0.5$) on the AFHQ-wild dataset at 256 x 256

Figure 10: Uncurated generations ($\psi = 0.5$) on the AFHQ-Dog dataset at 256 x 256