High Resolution Zero-Shot Domain Adaptation of Synthetically Rendered Face Images - Supplementary Material

Stephan J. Garbin, Marek Kowalski, Matthew Johnson, and Jamie Shotton

Microsoft

1 Overview

We use the supplementary material to provide more detail on our method, specifically related to the control vectors and the CS-ANN algorithm. We also show more results (Figure 2 and Figure 3), as well as some typical failure cases (Figure 15). Please note that our algorithm requires no synthetic training data, but uses just a pretrained model. The method is entirely implemented in tensorflow [1].

2 Control Vectors

The set of 33 control vectors, $v_{control}$, is used in our method both during the sampling stage, and the CS-ANN algorithm. As with the centroids detailed in the main text, the vectors $v_{control}$ are selected purely empirically, based on what we perceived were problem cases for the fitting. We acknowledge that this could be automated in a more principled way in future. We show all control vectors used, added to the mean face of StyleGAN2 (SG2) [3] in Figure 1. These vectors were obtained by annotating 2000 samples from the prior of SG2 based on eight roughly defined categories: (a) Gaze Direction, (b) Beard Style, (c) Head Orientation, (d) Light Direction, (e) Degree of Mouth Open, (f) Hair Style, (g) Hair Type, (h) Skin Texture. Note that we do not claim that these are orthogonal. Interestingly, bias in the FFHQ dataset [2] can be observed from $v_{control}$. For example, the gaze direction (a) is strongly correlated with the head orientation because subjects tend to look at the camera in photographs.

3 Convex Set Approximate Nearest Neighbour Search

After step 1 of the method returns w^s, we want to refine this latent code to map to an image better aligned to the input while staying in the space of plausible samples from G. The algorithm we propose and describe in the main text can be summarised as follows:

Result: Refined latent codes, w^n
Initialise variables and fix random seeds;
for $i_{outer} = 0;\ i_{outer} < 96;\ i_{outer} += 1$ **do**
  Sample 512 interpolation candidates;
  Reset learnable parameters;
  Set learning rate, $l = 0.01$;
  for $i_{inner} = 0;\ i_{inner} < 20;\ i_{inner} += 1$ **do**
    Compute loss & Update parameters;
    if $i_{inner} \% 4 == 0$ **then**
      $l = l/10.0$;
    end
    Update current w^n if better loss achieved;
  end
end

Algorithm 1: Convex Set Approximate Nearest Neighbour Search

While it would be possible to return the *last* result of Algorithm 1, we found it slightly more effective to return the one that achieves the best loss across all iterations. To do this, we simply store the best loss and corresponding w at any point in the outer or inner loop. If not using a learnable β and the corresponding control vectors, we found that convergence was substantially slower. Thus, while it carries the danger of producing implausible results (as it no longer guarantees the result is a combination of samples), we allow β to be learned. We note that clamping the range of β during the optimisation appeared to be a sufficient constraint in practise.

As pointed out in the text, we allow different weights α for each of the 18 SG2 inputs. We also experimented with a single weight per sample, using different alphas for each of the actual 18×512 inputs, as well as mixtures of these combinations. The former was substantially slower to converge while the latter was quick to produce implausible results.

For matching synthetic images with illumination substantially different from the dataset of real images, we also found it helpful to include a single learnable parameter (clamped to be in the range of $0.7 - 1.3$) to multiply the reproduced images with, thereby changing brightness by a simple linear transformation. We acknowledge that additional transformations such as histogram equalisation could be explored in future.

More results with intermediate progress are shown in Figure 4, Figure 5, Figure 6, Figure 7, Figure 8, Figure 9, Figure 10, Figure 11, Figure 12, Figure 13, and Figure 14.

4 Failure Cases

We count any image for which our method fails to produce a relatively close match after 10 runs with different random seeds as a failure case. As can be seen in Figure 15, we found very light hair, extreme poses / facial expressions and a lack of contrast between the face and hair challenging cases. We also observe that

the further the synthetic image semantics are from the data space (i.e. FFHQ), the less likely our method is to produce satisfactory results. Cases (a) and (b) in Figure 15, combining beards and haircuts more typically associated with female faces, are examples of this. In such instances, the extension to our method that retains the hair from the synthetics can still produce plausible results as long as the facial shapes are approximately matched.

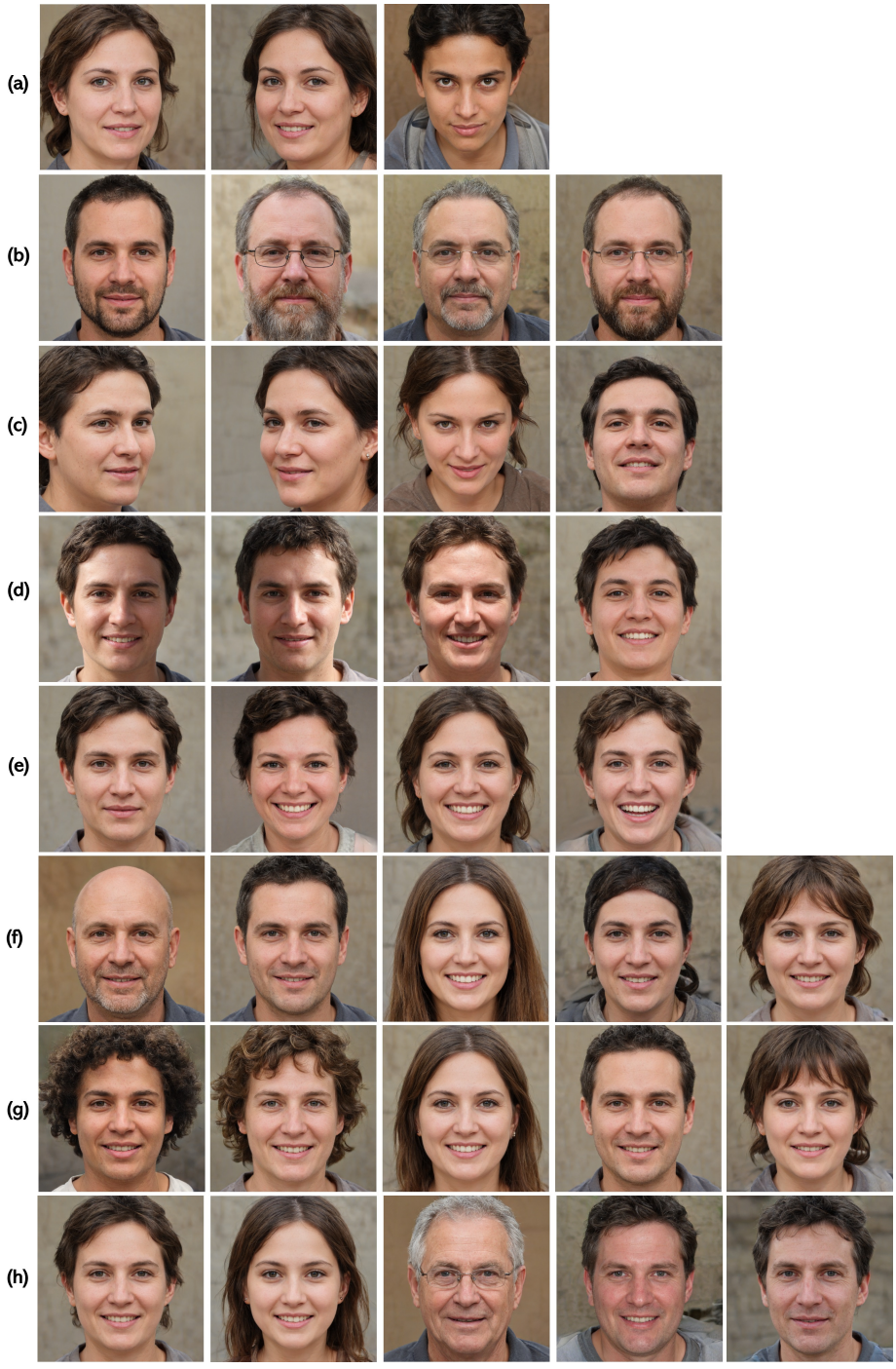

Fig. 1: All control vectors, added to the 'mean' face of SG2. The categories are (a) Gaze Direction, (b) Beard Style, (c) Head Orientation, (d) Light Direction, (e) Degree of Mouth Open, (f) Hair Style, (g) Hair Type, (h) Skin Texture.

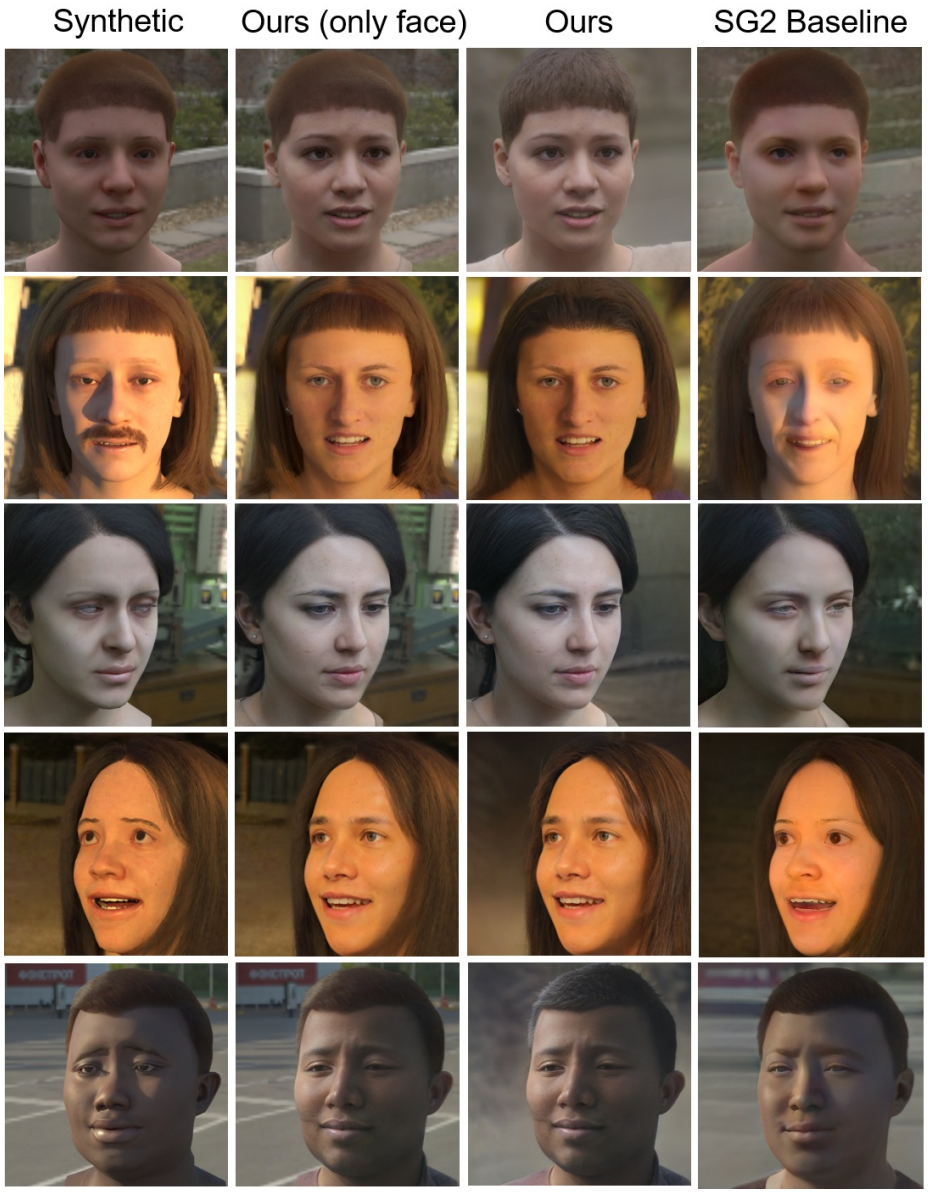

Fig. 2: More results of our method vs the *StyleGAN2 Baseline*. We note that while our method preserves most of the synthetic image semantics, smaller beards, and eye gaze can be lost. Best viewed digitally, as every image is at 1K resolution.

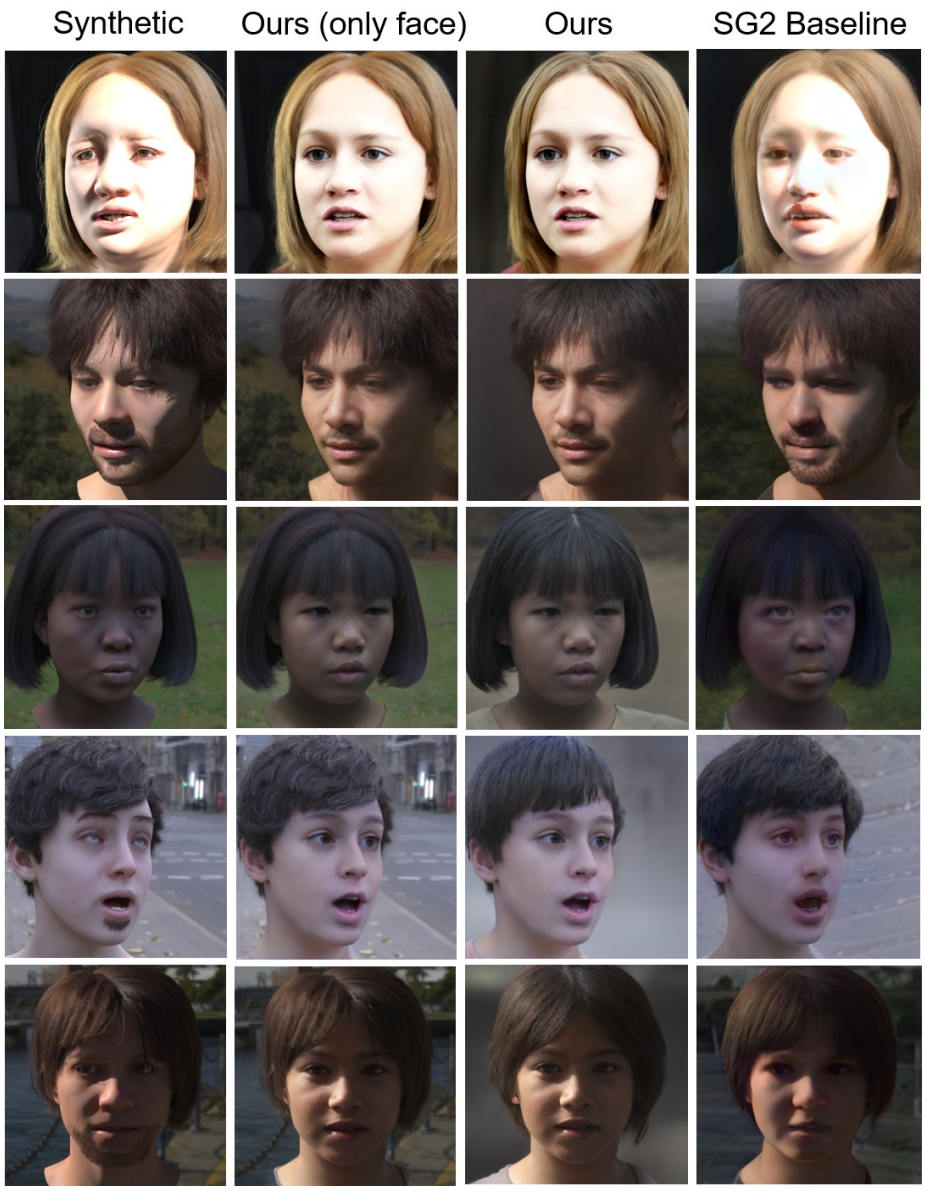

Fig. 3: Yet more results of our method vs the *StyleGAN2 Baseline*. We note that while our method preserves most of the synthetic image semantics, smaller beards, and eye gaze can be lost. Best viewed digitally, and in colour.

Fig. 4: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

Fig. 5: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

Fig. 6: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

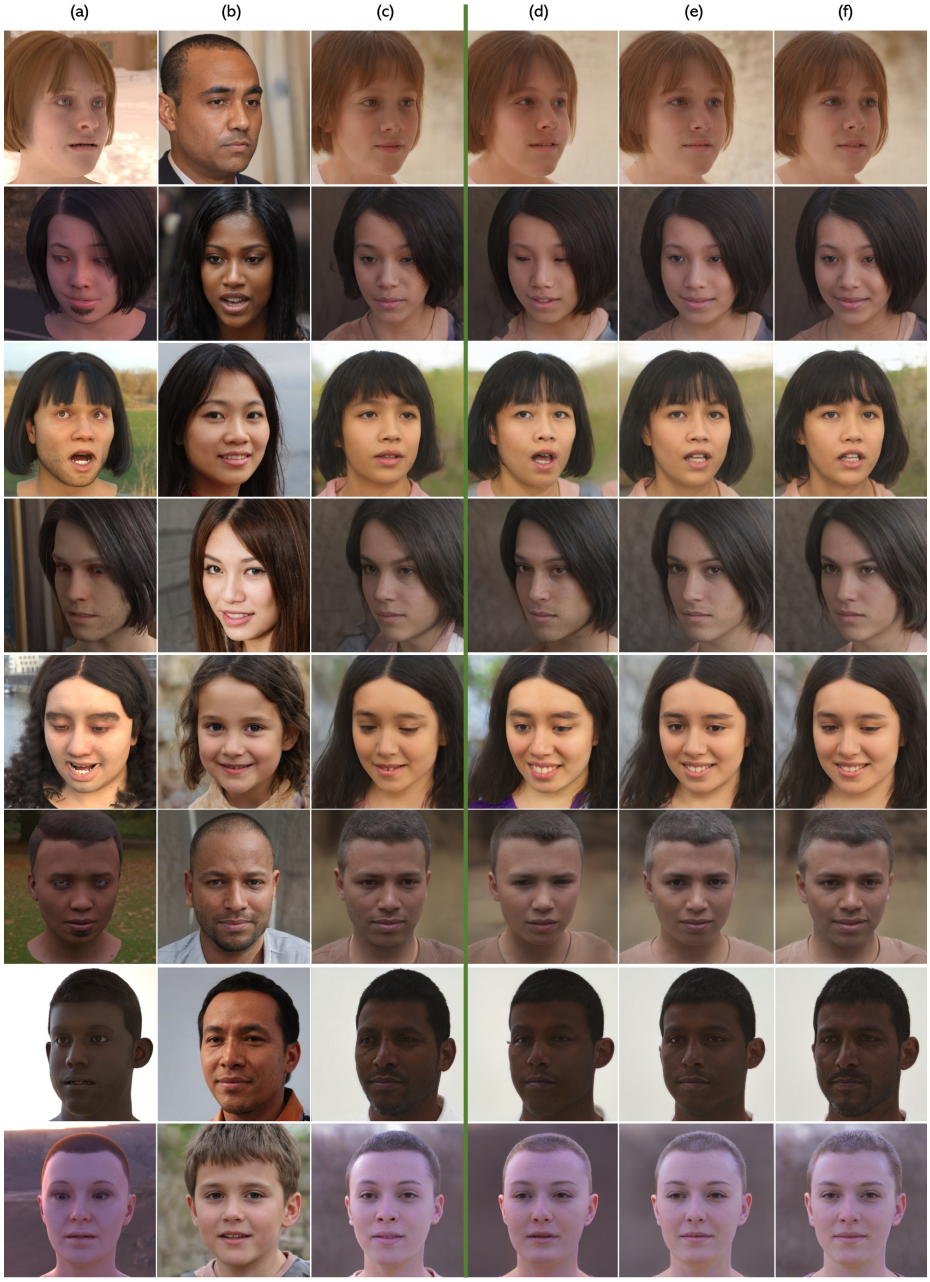

Fig. 7: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

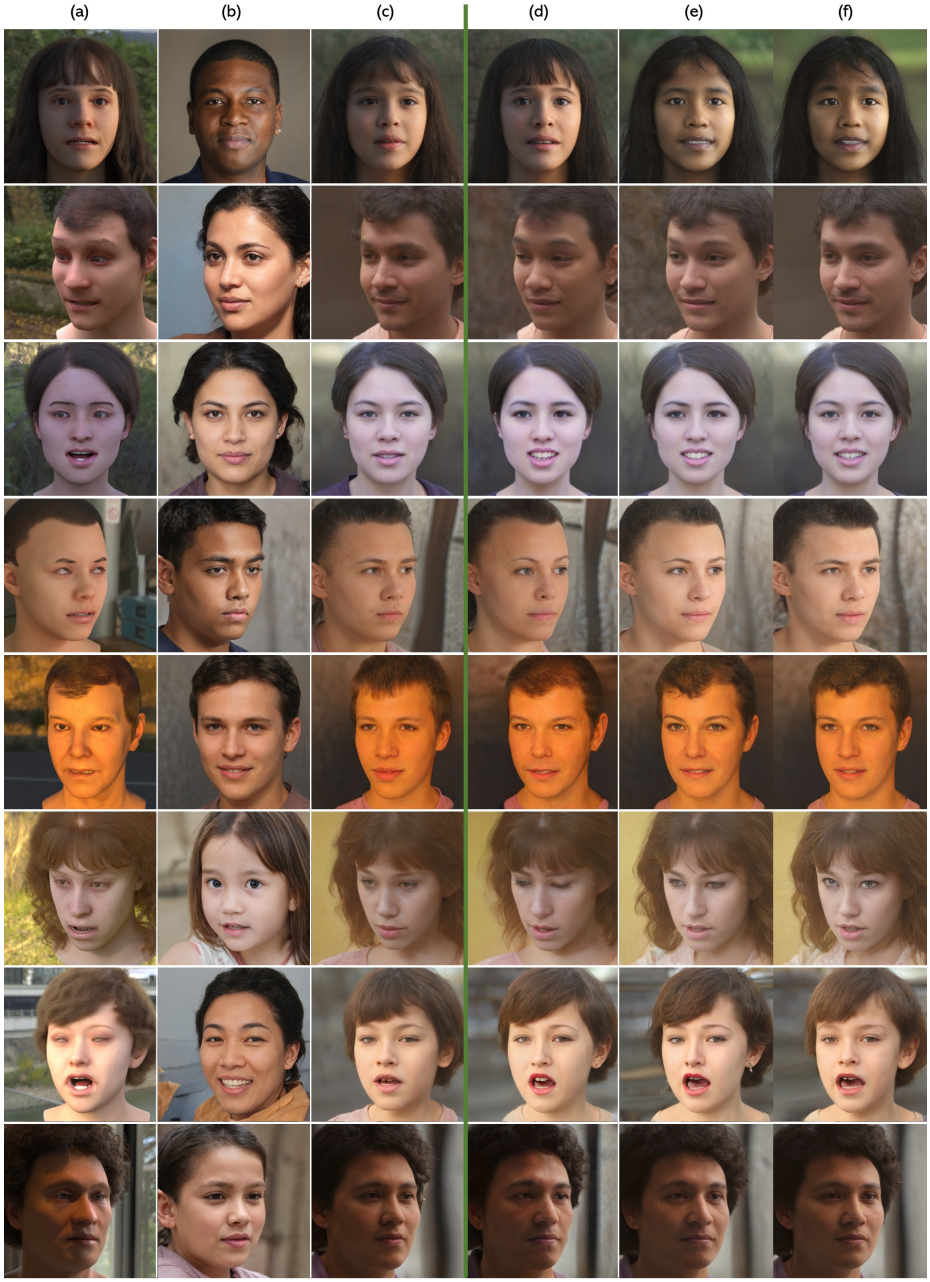

Fig. 8: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

Fig. 9: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

Fig. 10: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

Fig. 11: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

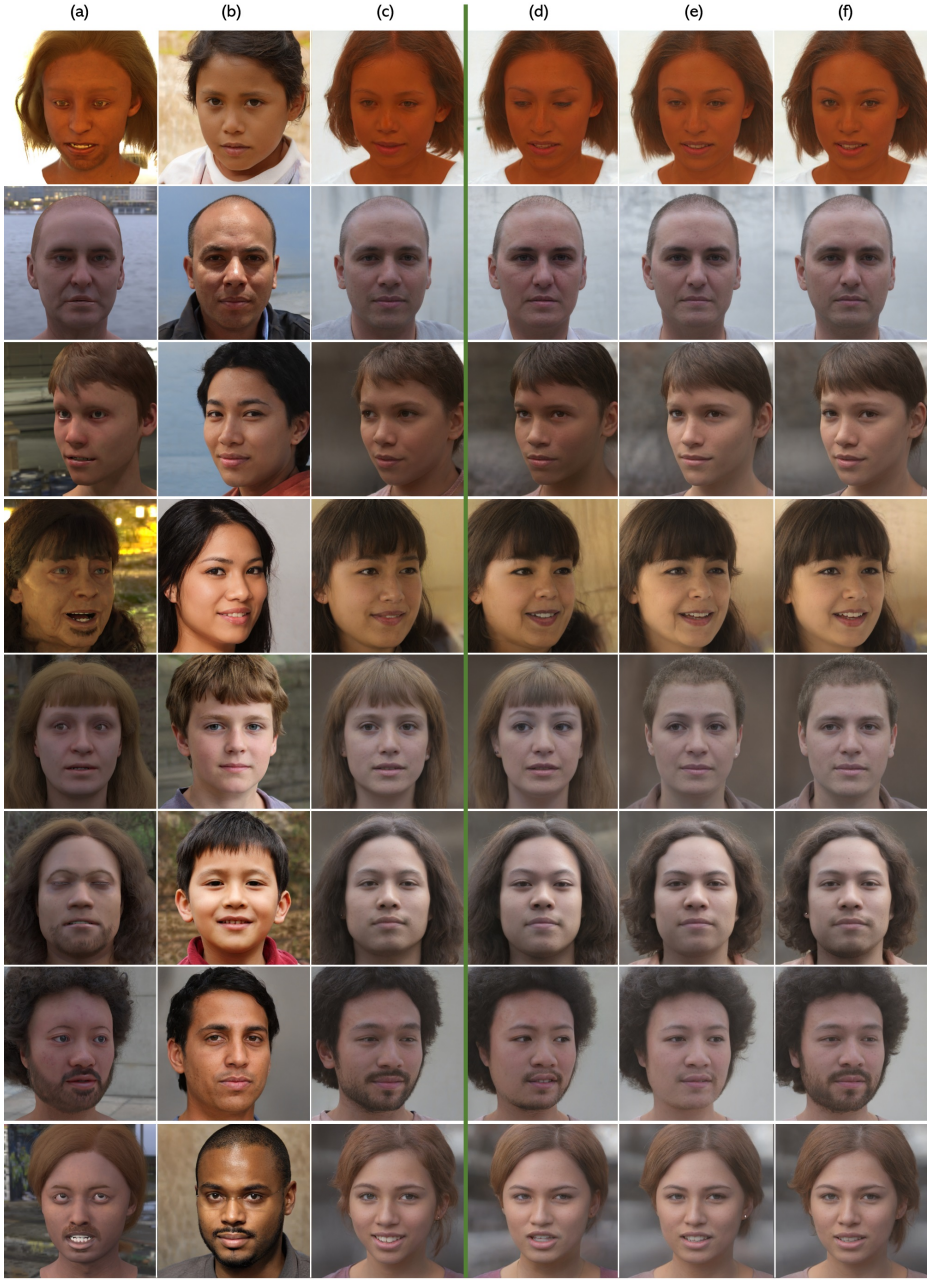

Fig. 12: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

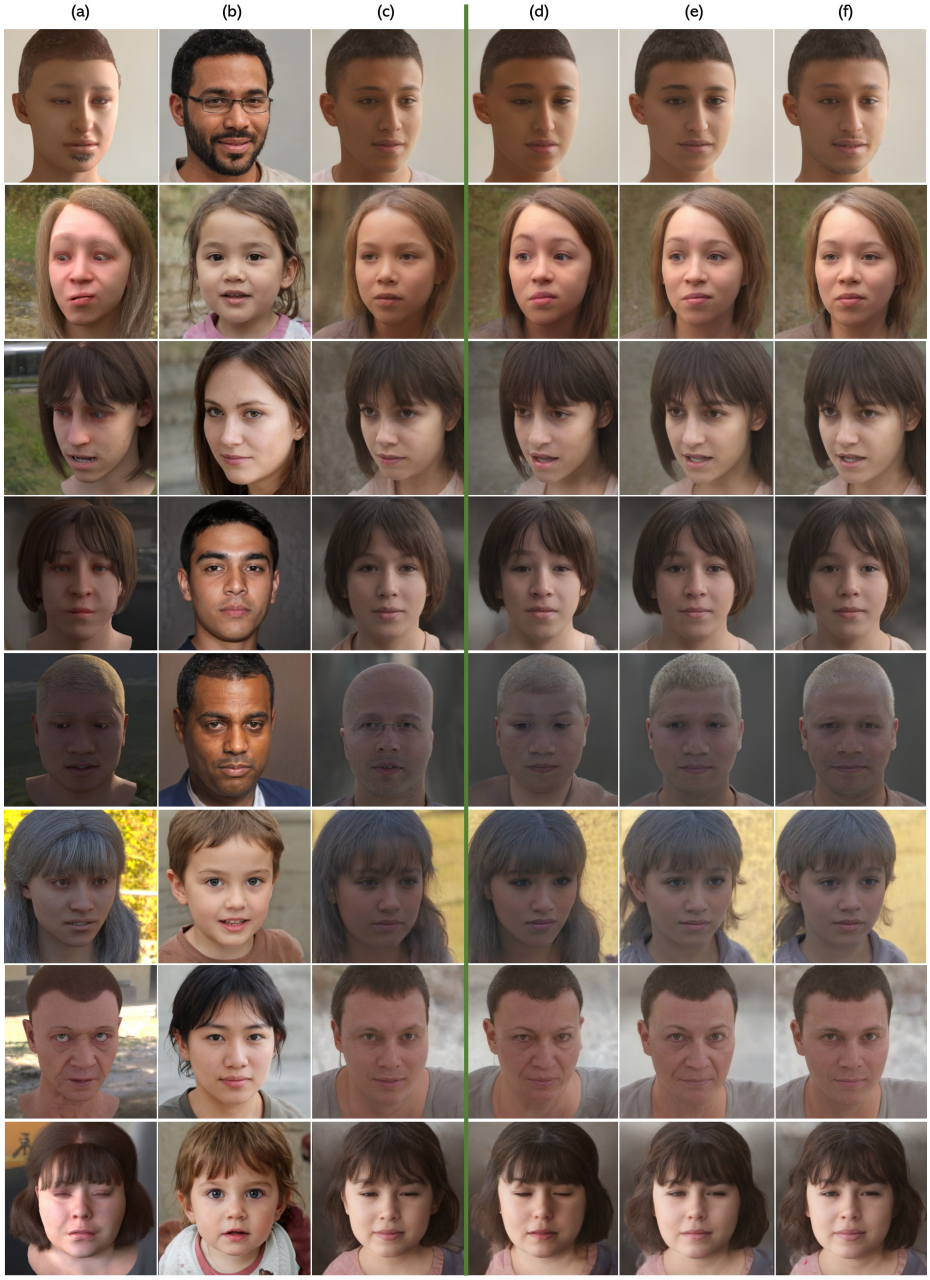

Fig. 13: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

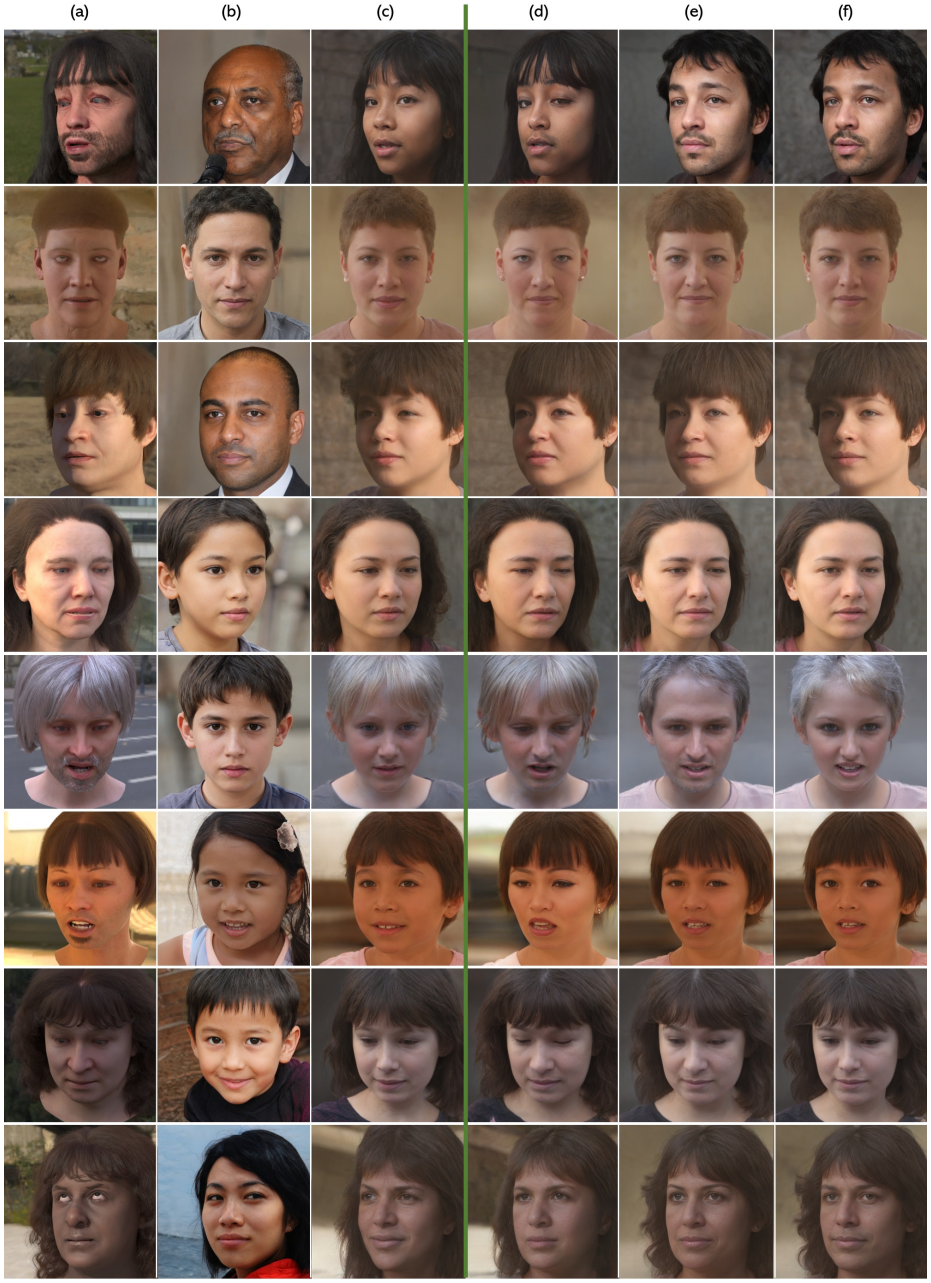

Fig. 14: Extended illustration of the different steps of our method. (a) synthetic render, (b) output of the sampling in step 1, (c) the result of CS-ANN Search in step 2, and (d-f) some samples from the possible results of step 3. Please refer back to the method section of the main text.

Fig. 15: Example failure cases (persistent after 10 random starts of our method). Note how retaining the hair from synthetics in *(ours (only face))* can help in such instances. It is of interest that the more unrealistic hair appearance in (e) is matched to a hat by our algorithm. The SG2 generator also appears to resist being steered with combinations unlikely to appear in FFHQ, such as (a) or (b).

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), https://www.tensorflow.org/, software available from tensorflow.org
2. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. CoRR **abs/1812.04948** (2018), http://arxiv.org/abs/1812.04948
3. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan (2019)