Delta Denoising Score – Supplementary Materials

Amir Hertz[1,2] Kfir Aberman[1] Daniel Cohen-Or[1,2]

[1]*Google Research* [2]*Tel Aviv University*

A. Societal Impact

Our method introduces an unsupervised editing framework for images. Our framework might be exploited for producing fake content, however, this is a known problem, common to all image editing techniques. Moreover, our method relies on generative priors of a large text-to-image diffusion models that might contain undesired biases due to the auto-filtered enormous dataset that they were trained on. Those undesired biases could infiltrate to our editing results thorough our optimization process or our distilled networks. However, we believe that our DDS optimization technique can help in the future to reveal such undesired biases and editing directions, similarly to the way it operates today to clean noisy undesired components within editing directions.

B. Implementation Details

Zero-shot DDS optimization Unless specified otherwise, in all of our zero shot experiments, we initialize the latent image $\mathbf{z}$ by the reference latent image $\hat{\mathbf{z}}$ and apply our DDS optimization for 200 iterations ($\sim$ 18 seconds on a single A100 GPU) using SGD with learning rate of 2 and applying learning rate decay of 0.9 in intervals of 20 iteration.

Image-to-image network training Our image-to-image networks were trained using a batch size of 2 for 125000 iterations ($\sim$ 23 hours on a single A100 GPU). We use the Adam optimizer with a learning rate of 10^{-5} and learning rate warmup over 10000 iterations using a linear scheduler. For the DDS, we set the classifier-free guidance scale (CFG) to 25 using CFG warmup starting from 1 over 20000 iterations using a cosine scheduler. For the identity regularization, we start with $\lambda_{\text{id}} = 3$ and cool it down to 0.1 over 20000 iterations using a cosine scheduler.

C. Additional Ablations

Number of optimization steps We set the number of optimization steps to 200, as we found it sufficient for most desired edits. However, for obtaining large structural changes or color modifications, the number of optimization steps should be increased. See example in Figure 2 where structural modification, of replacing a flamingo to an elephant requires 400 steps, while modify the flamingo to a bronze sculpture converges faster. Notice that in some cases, where we seemingly request for a large modification, like replacing the flamingo to a giraffe, 200 optimization are still sufficient since the overall structure of the giraffe is similar to that of the flamingo.

Optimizer Selection We tested our DDS optimization using both SGD and Adam [4] optimizer and found that using *vanilla* SGD leads to higher quality results. Figure 3 compares the two alternatives (top rows) with respect to the baseline SDS with SGD optimization (bottom row). We can clearly see that both DDS optimization achieve higher quality results compared to the baseline. However, by looking at the accumulated difference of the image during the optimization (right saliency map beneath each image), we can see that the Adam based optimization results with more changes and artifacts that are not related to editing prompt. The reason for the change in the quality can be explained through the adaptive nature of Adam. For simplicity, consider the simpler update rule of Adagrad [2]:

$$\theta_t \leftarrow \theta_{t-1} - \gamma \frac{g_t}{\sqrt{\sum_{i=1}^{t} g_t^2}},$$

where t enumerates the optimization steps, g_t is the gradient of θ calculated at step t and γ is the optimization learning rate. We can see that the normalization to the gradients may magnify outlier gradients or reduce the weight of *good* gradients. Figure 3 visualize such update to the pixels for different steps across the optimization (left saliency map beneath each image). It can be seen that the Adam based optimization (middle row) leads to a uniform update across the pixels compared to SGD (top row), where most of the energy is located at the pixels that are relevant to the edit.

Regularized SDS optimization An alternative to our DDS approach is to use the SDS with additional regulariza-

tion that prevents large changes in the edited image $\mathbf{z}$ with respect to the input $\hat{\mathbf{z}}$. The simplest choice is to ~~add~~ regularise the optimization with a weighted $\mathcal{L}_2$ loss between $\mathbf{z}$ and $\hat{\mathbf{z}}$. In this setting, the gradient of $\mathbf{z}$ is given by:

$$\nabla_{\mathbf{z}} = \nabla_z \text{SDS} + \lambda_{id} \left(\mathbf{z} - \hat{\mathbf{z}}\right),$$

Where λ_{id} is the weight for the regularisation $\mathcal{L}_2$ loss. Figure 4 shows the results of regularized SDS optimizations using increasing weight for the regularization term. As expected, increasing the value of λ_{id} harms the fidelity to the edit prompt, while the blurriness side-effect of SDS cannot be avoided.

Effect of CFG on the similarity between δ_{bias} and $\hat{\delta}_{\text{bias}}$ We state that DDS guides the direction towards δ_{text} if $\delta_{\text{bias}} \approx \hat{\delta}_{\text{bias}}$ which we validate empirically using a fixed CFG $\omega = 7.5$, a typical CFG value used in T2I generation and editing. However, it can be easily seen by the definition of CFG that any error is scaled by ω, which we validate by repeating the experiment of Fig. 8 (right). See the results in Fig. 1.

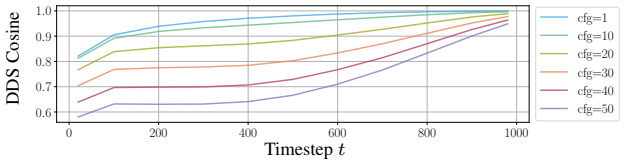

Figure 1: Cosine similarity between the SDS directions (Eq. 4) using increasing values of classifier free guidance (CFG) scale.

D. Additional Results

Zero-shot image editing results on real images Additional image editing results using DDS optimization are shown in Figures 5 and 6. All results applied on real images from COCO, and Unsplash datasets [1, 6]. Notice that our method works with simple input prompts describing the edit we want to apply to the image. We use a reference text $\hat{y}$ only in cases where we want to apply the edit over a specific object. Otherwise, we set $\hat{y}$ to the embedding of the null text.

Image-to-image networks results Additional results are shown in Figures 7, 8 and 9. The first network was trained to change the material of a sofa in an input image. The network was trained on a synthetic dataset containing 5000 images of living rooms. An additional network was trained to synthesize different flowers in images of potted plants. The network was trained on a synthetic dataset containing 5000 of potted plants in living rooms, kitchens, in gardens, and in city streets. Finlay, we train different networks to modify a person in an input image to other characters. Synthesizing images of persons using Stable Diffusion usually results in poor results. Therefore we train our *character* networks over FFHQ in-the-wild dataset (unaligned) [3]. During training, we set a single pair of fixed prompts for the DDS optimization, "A photo of a person." for the embedding $\hat{y}$, and for the target embedding y, we replace the word "person" with one of the characters: a Claymation character, A sculpture, a 3D Pixar character and to a zombie. All results are shown over images from ILSVRC [5] and COCO [1] datasets.

References

[1] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *Computer vision and pattern recognition (CVPR), 2018 IEEE conference on*. IEEE, 2018. 2, 8, 9, 10

[2] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. 1

[3] 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. 2, 10

[4] Diederik P Kingma, J Adam Ba, and J Adam. A method for stochastic optimization. arxiv 2014. *arXiv preprint arXiv:1412.6980*, 106, 2020. 1

[5] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 2, 8, 9

[6] Unsplash. Unsplash image collection, 2022. 2

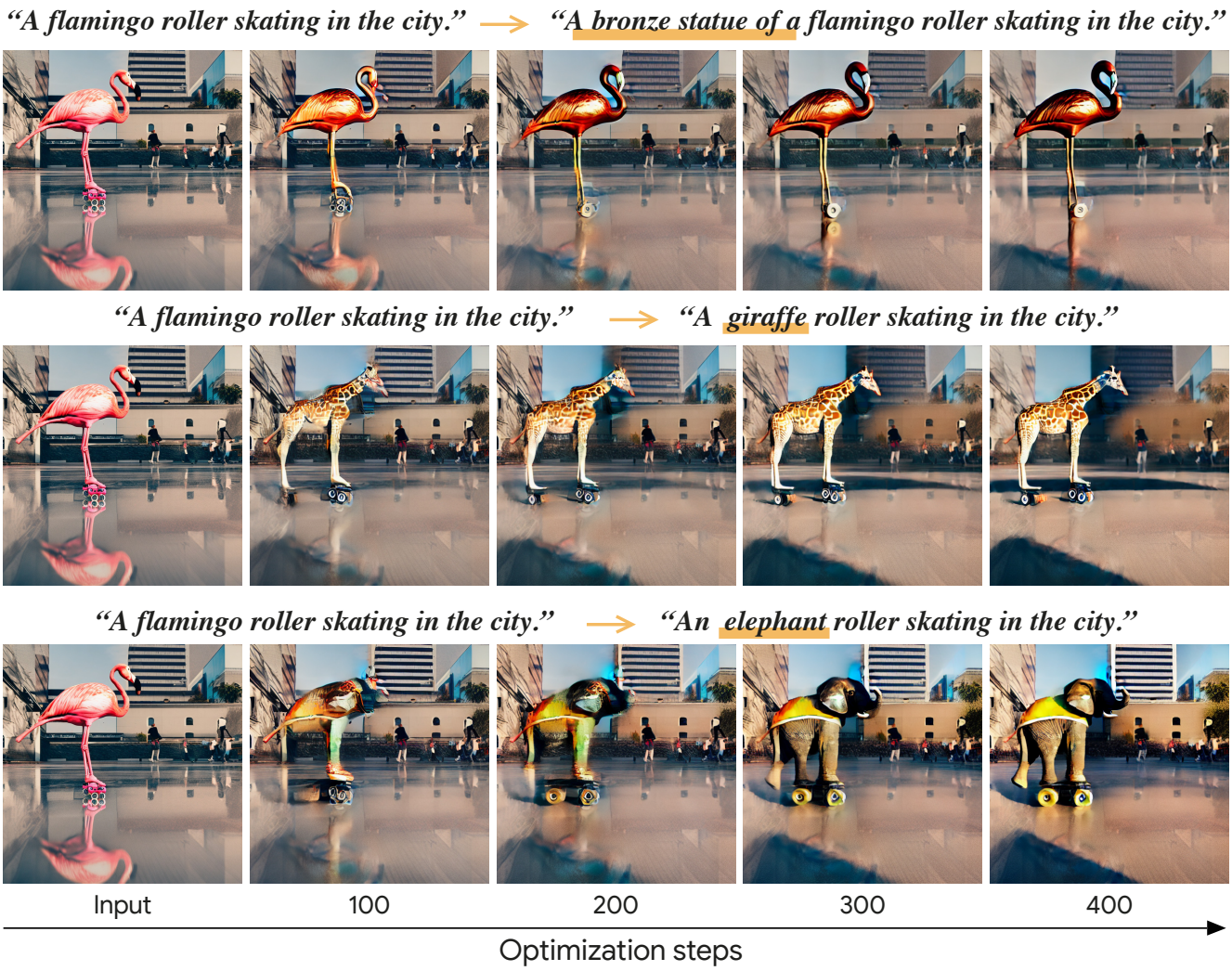

Figure 2: **DDS Optimization convergence.** *Some challenging edits, such as changing a flamingo to an elephant, may require more time to converge compared to other edits. In most cases, we have found that* 200 *steps are sufficient.*

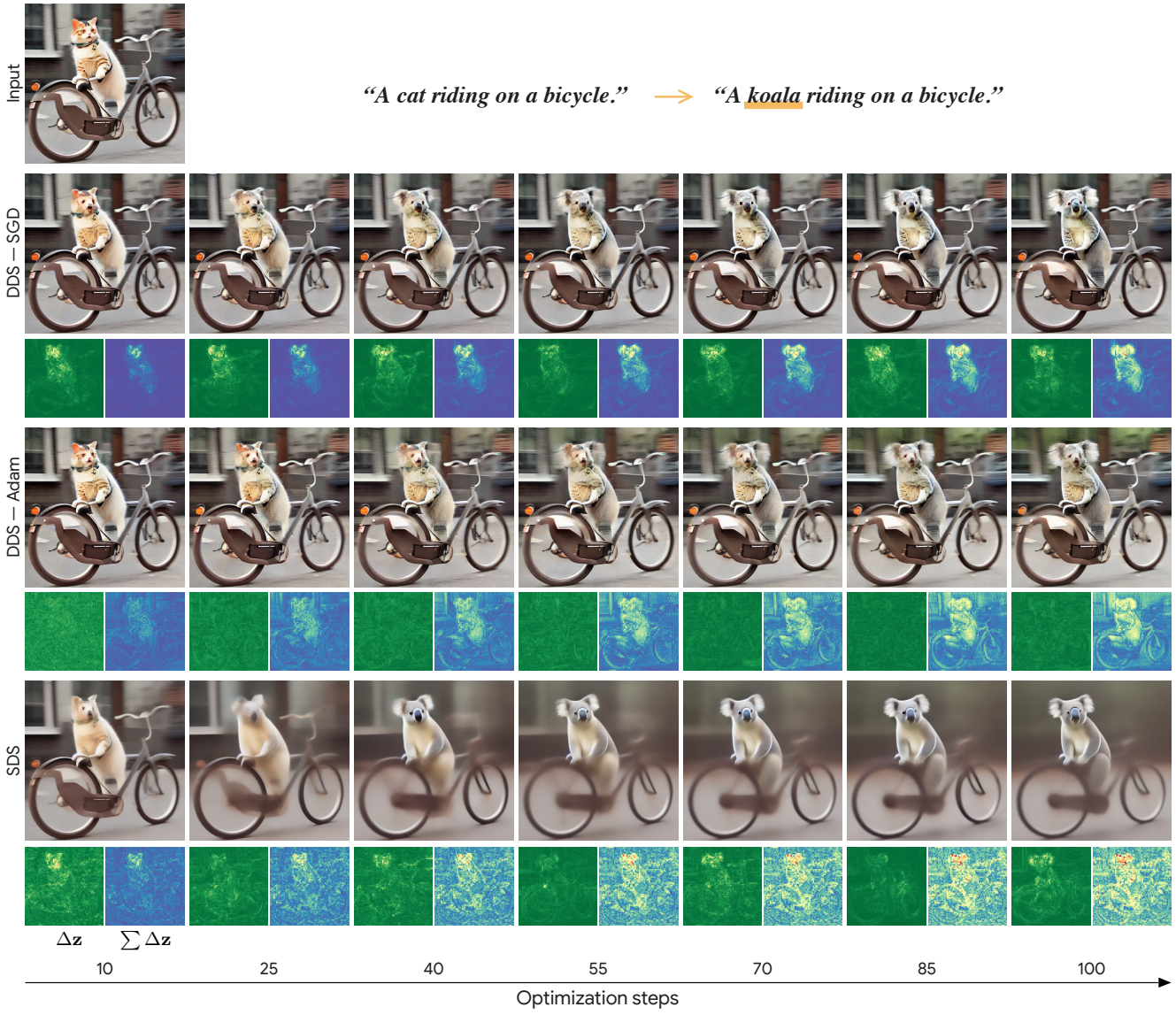

Figure 3: **Optimizer selection.** *Each row shows a zero-shot editing optimization sequence using different settings. The green-to-red heatmaps (left map beneath each image) show the norm of the latent pixels update in a specific step. The blue-to-red heatmaps (right map) show the accumulated difference norm with respect to input image (top).*

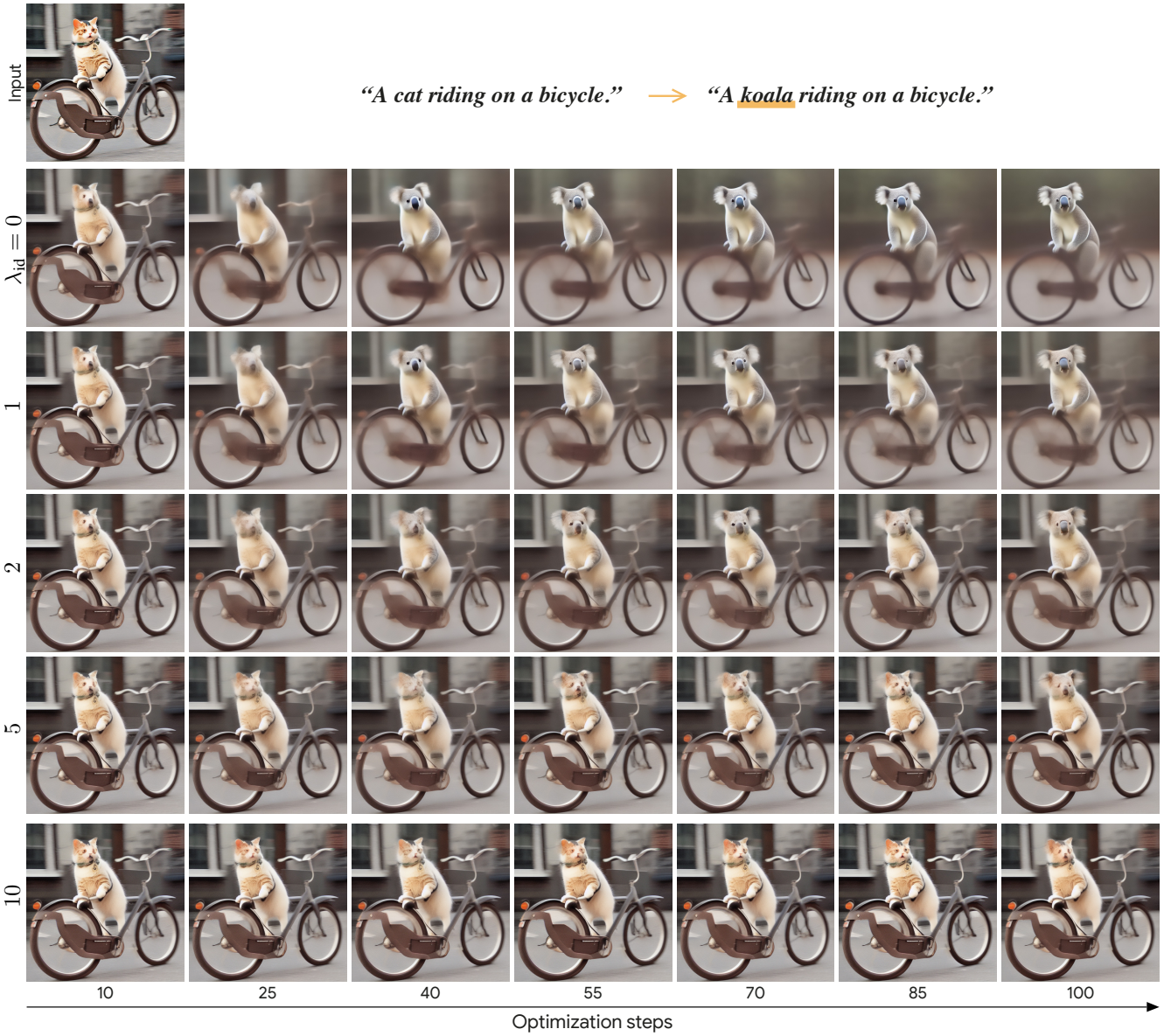

Figure 4: **Regularized SDS optimization as a baseline.** *We combined a* vanilla *SDS optimization with a weighted $\mathcal{L}_2$ loss between the optimized image and the input image (top). From top row to bottom: we use a larger weight (λ_{id}) for the $\mathcal{L}_2$ loss. As can be seen, by modifying λ_{id} we can trade off between fidelity to the text prompt and fidelity to the input image.*

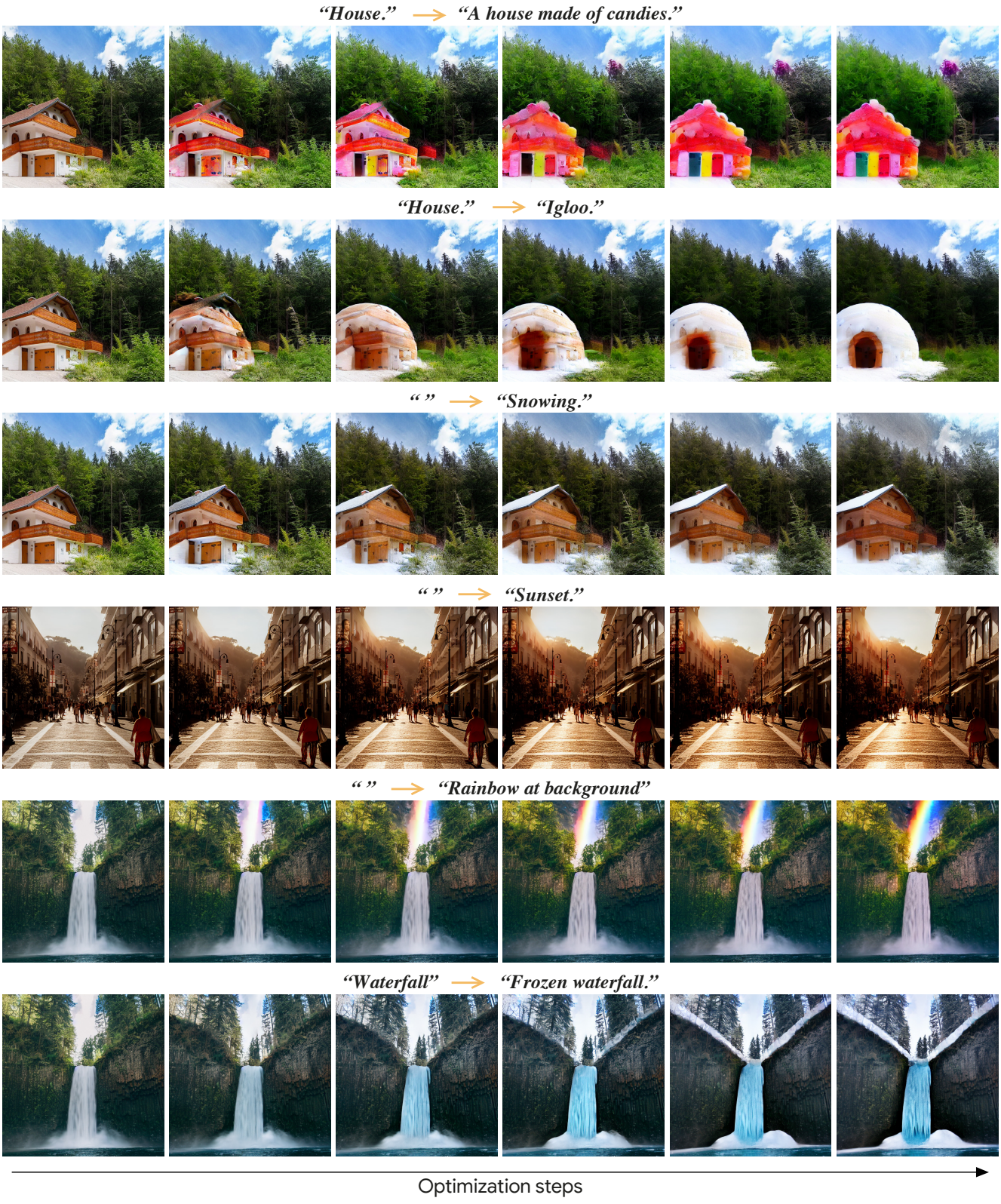

Figure 5: **Zero-shot image editing using DDS optimization on real images.** *Using DDS, we can apply a variety of edits over real images using simple input prompt descriptions (like "house", or even an empty prompt). The editing operations are mask free and may contain global descriptions, for example, changing the lighting in the image, or local descriptions, such as changing a house to an igloo.*

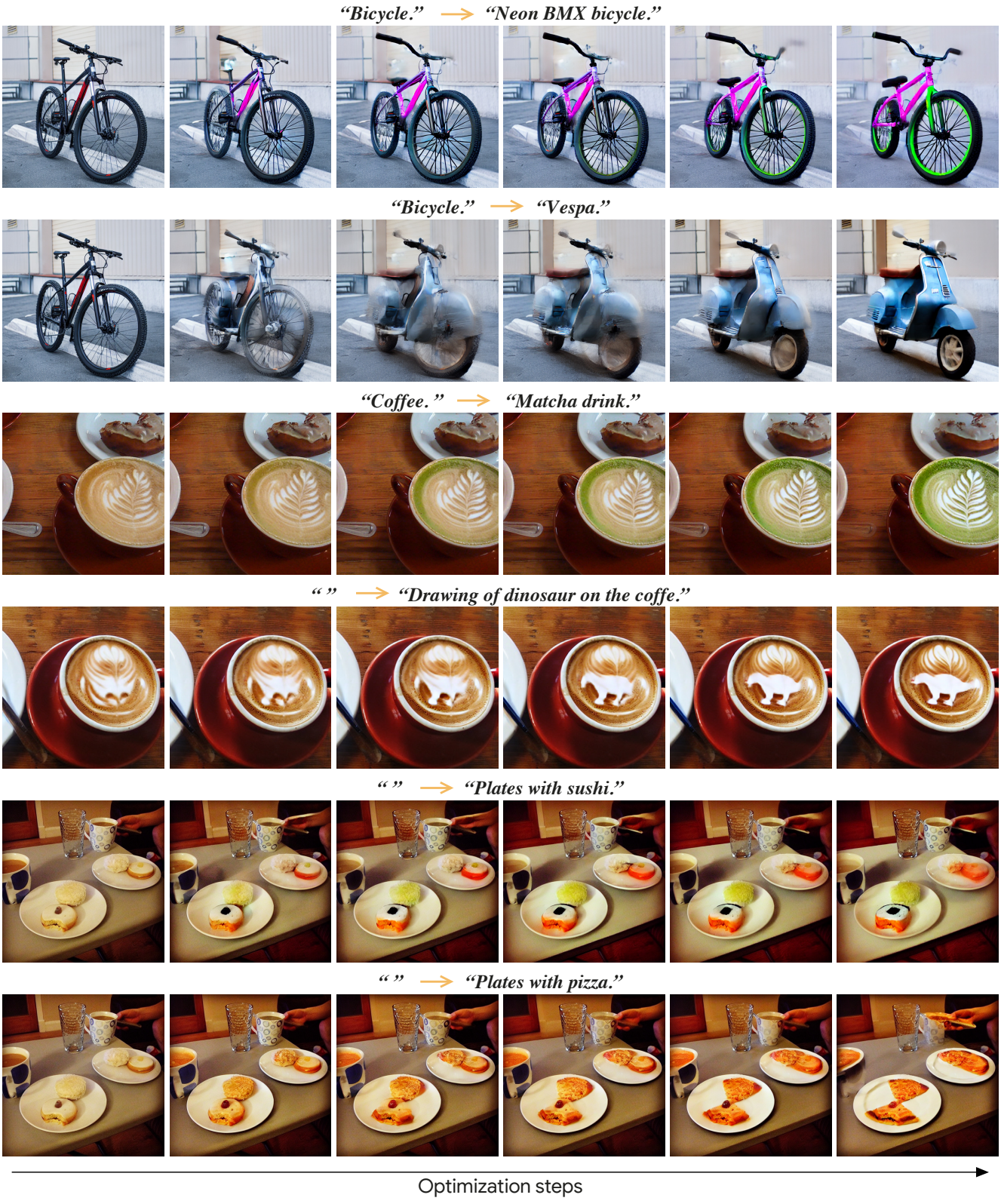

Figure 6: **Zero-shot image editing using DDS optimization on real images.** *Using DDS, we can apply a variety of edits over images using simple input prompt descriptions (like "Coffee", or even an empty prompt). The editing operations are mask free and may contain structural changes, for example, changing a bicycle to a Vespa, or stylistic changes, such as changing coffee to a matcha drink.*

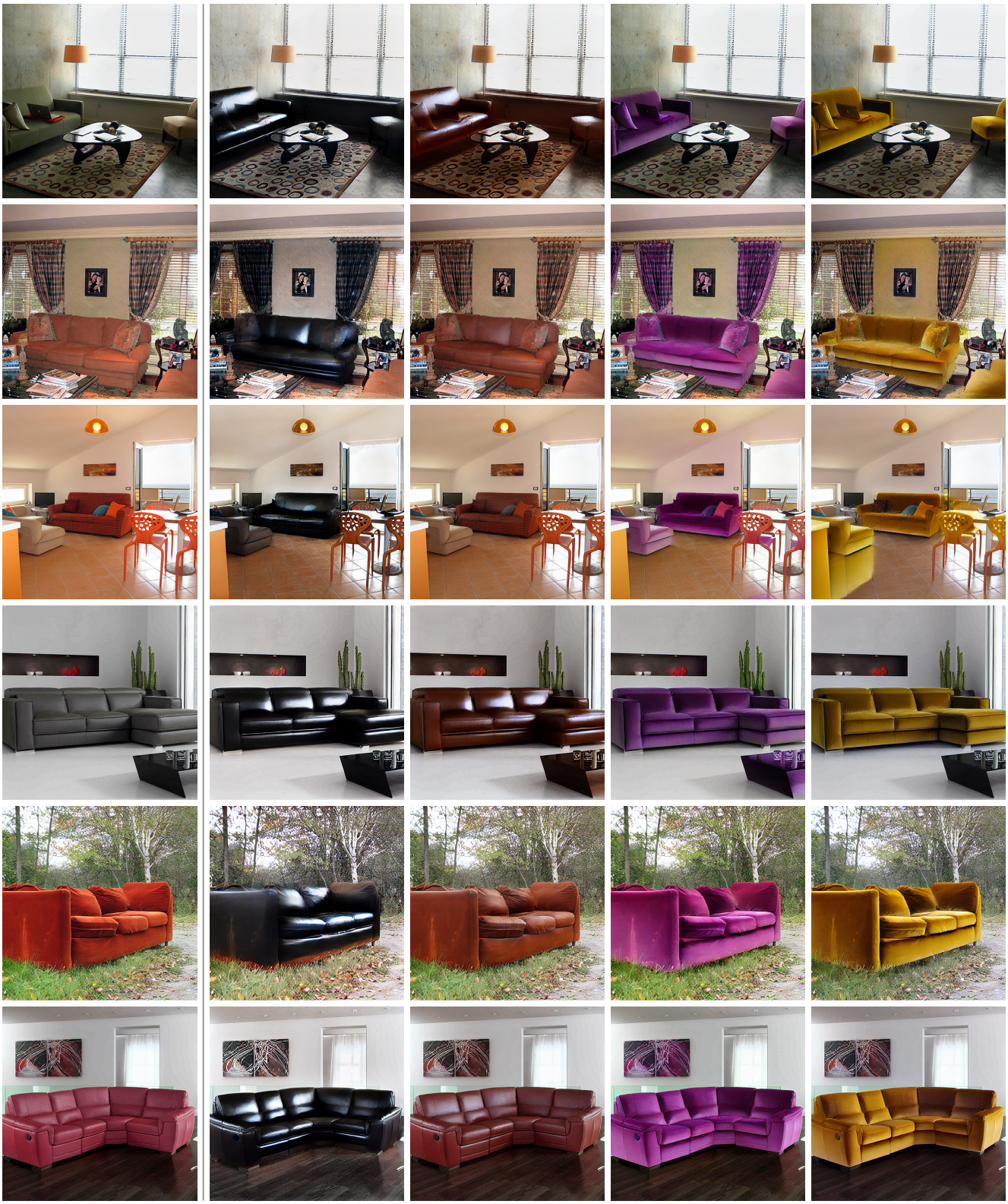

Figure 7: **Unsupervised multi-task image-to-image translation– sofas network results.** *The network was trained to change the color and material of the sofa in the input image. The network was trained on synthetic images of living rooms and tested on real living rooms images from ILSVRC [5] and COCO [1] validation sets.*

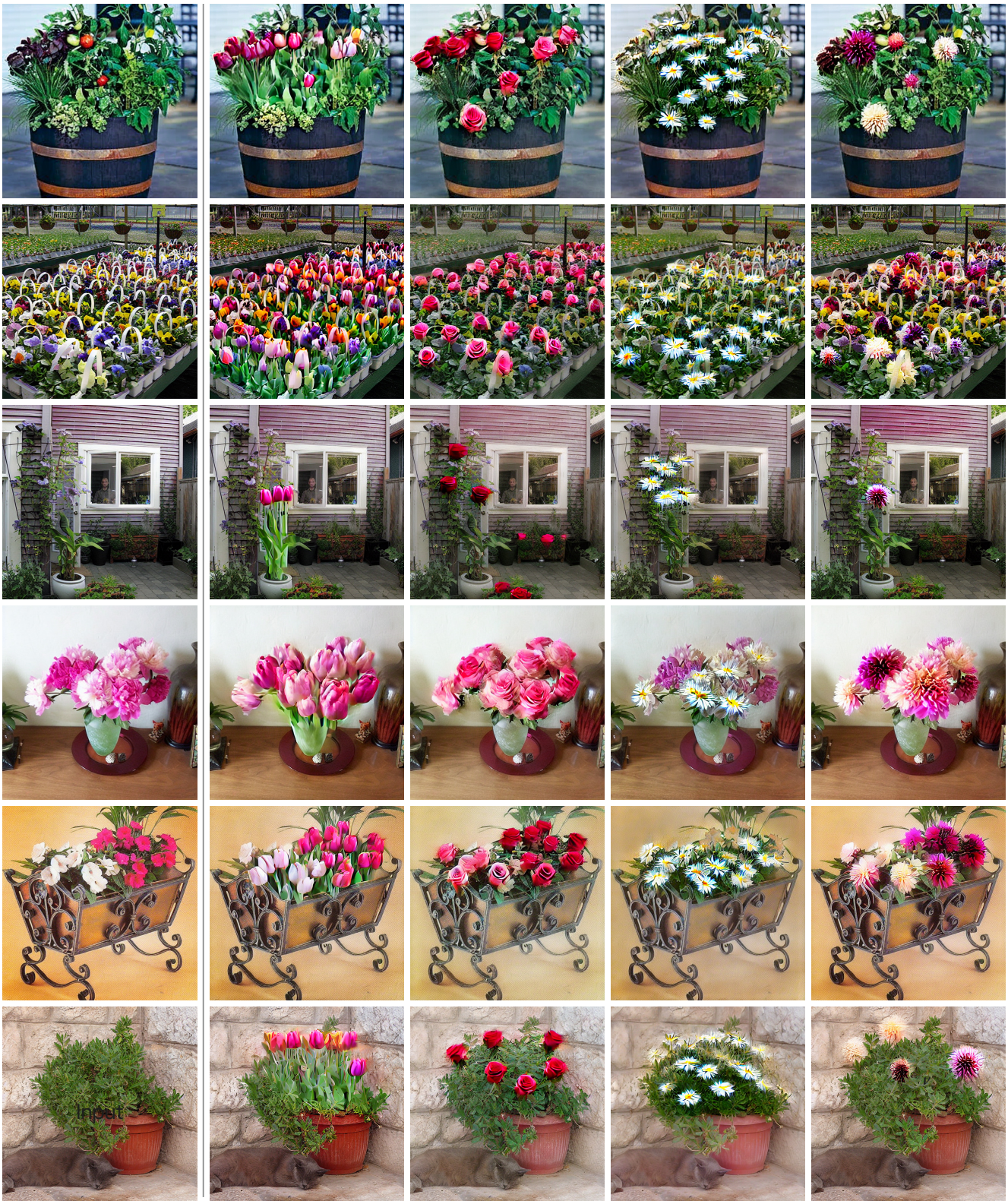

Figure 8: **Unsupervised multi-task image-to-image translation– flowers network results.** *The network was trained to change to add different flowers potted plant in the input image. The network was trained on synthetic images and tested on real images of flowerpots [5] and potted plant [1].*

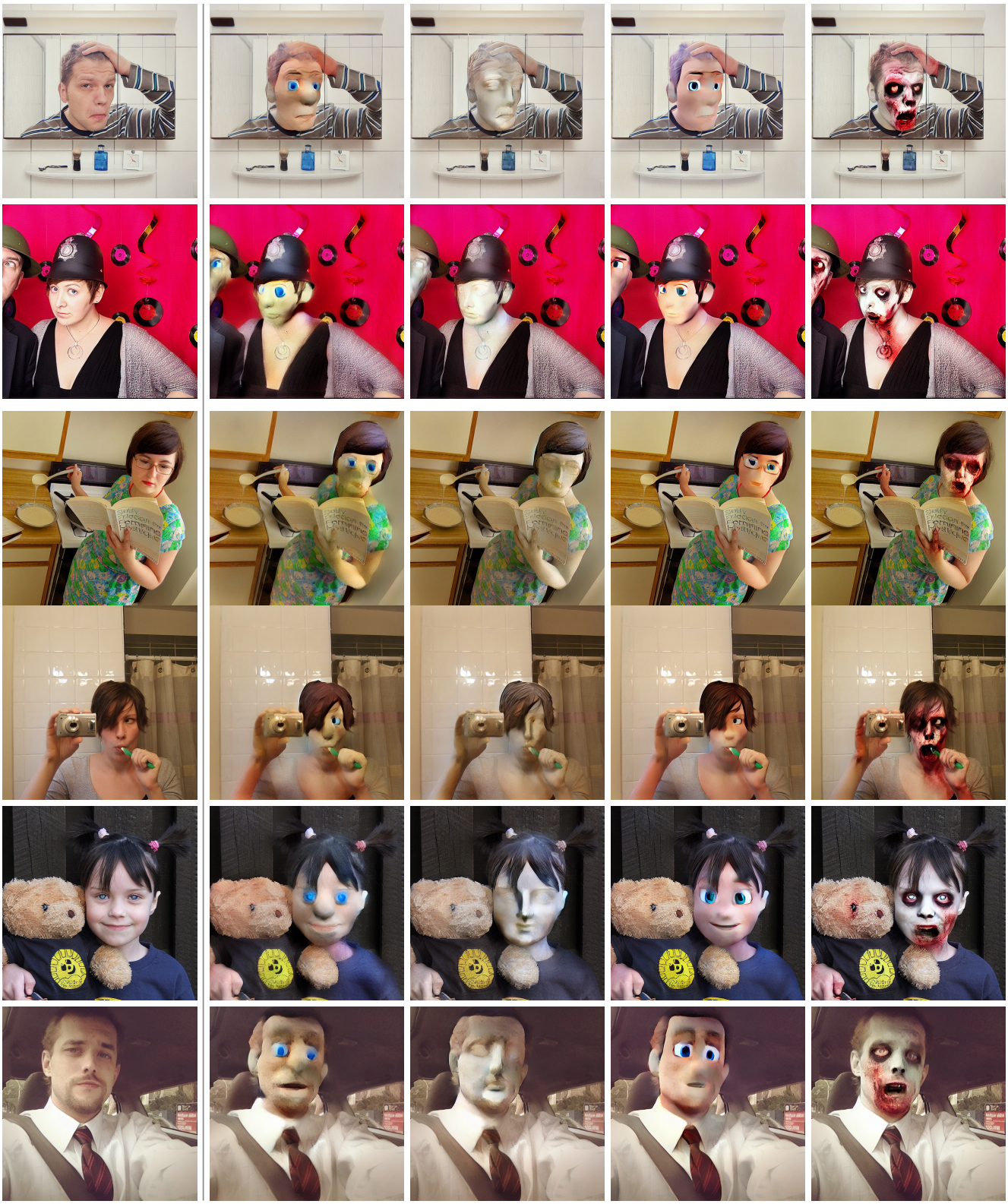

Figure 9: **Unsupervised image-to-image translation– characters.** *Different networks were trained to change the person in the image to various characters. The networks were trained on FFHQ in-the-wild dataset (unaligned) [3] and tested on images from COCO dataset [1].*