Supplementary material - Neural Rerendering in the Wild

Moustafa Meshry*[1], Dan B Goldman[2], Sameh Khamis[2], Hugues Hoppe[2], Rohit Pandey[2], Noah Snavely[2], Ricardo Martin-Brualla[2]

[1]University of Maryland, [2]Google Inc.

A. Supplementary Results

Appearance variation. Figure 2 shows additional results of diverse appearances modeled by our proposed staged training method on the San Marco dataset. As in Figure 6 in the main text, it shows realistic renderings of five different scenes/viewpoints under four different appearances obtained from other photos.

Qualitative comparison. We evaluate our technique against Shan *et al.* [3] on the Colosseum. In Section 4, we report the result of a user study run on 20 randomly selected sets of output images that do not contain close-ups of people or cars, and were not in our training set. Figures 3, 4 show a side-by-side comparison of all 20 images used in the user study.

Quantitative evaluation with learned segmentations To quantitatively evaluate rerendering using estimated segmentation masks, we generate semantic labelings for the validation set, as described in Section 3.3, and recompute the quantitative metrics, as in Table 1 in the main paper, for our proposed method. Note that estimated semantic maps will not perfectly match those of the ground truth validation images. For example, ground truth semantic maps could contain the segmentation of transient objects, like people or trees. So, it is not fair to compare reconstructions based on estimated segmentation maps to the ground truth validation images. While results in Table 1 show some performance drop as expected, we still get a reasonable performance compared to that in Table 1 in the main text. In fact, we still perform better than the BicycleGAN baseline on the Trevi, Pantheon and Dubrobnik datasets, even though the BicycleGAN baseline uses ground truth segmentation masks.

B. Implementation Details

We use different networks for the staged training and the baseline mode. We obtain best results for each model with different networks. Below, we provide an overview of the different architectures used in the staged training and the baseline models. Code will be available at `https://bit.ly/2UzYlWj`.

	+Sem+StagedApp		
Dataset	VGG	L_1	PSNR
Sacre Coeur	67.74	28.66	16.45
Trevi	77.35	26.03	17.90
Pantheon	62.54	25.40	17.29
Dubrovnik	74.44	30.39	16.18
San Marco	75.58	26.69	17.34

Table 1: We evaluate our staged training approach using estimated segmentation masks, as opposed to Table 1 in the paper, which uses segmentation masks computed from ground truth validation images.

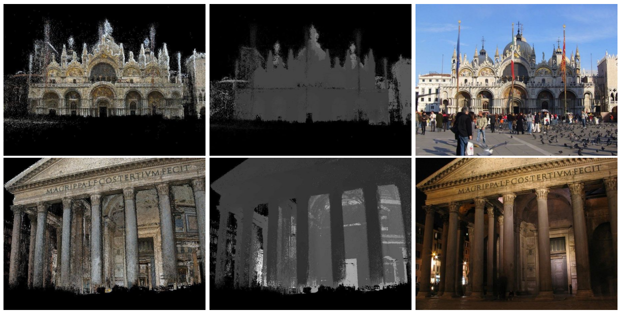
B.1. Neural rerender network architecture

Our rerendering network is a symmetric encoder-decoder with skip connections. The generator is adopted from [1] without using progressive growing. Specifically, we extend the GAN architecture in [1] to a conditional GAN setting. The encoder/decoder operates at a 256 × 256 resolution, with 6 downsampling/upsampling blocks. Each block has a downsampling/upsampling layer followed by two single-strided 3 × 3 *conv* layers with a *leaky ReLu* ($\alpha = 0.2$) and *pixel-norm* [1] layers. We add skip connections between the encoder and decoder by concatenating feature maps at the beginning of each decoder block. We use 64 feature maps at the first encoder and double the size of feature maps after each downsampling layer until it reaches size 512.

B.2. Appearance encoder architecture

We implement the appearance encoder architecture used in [2] except that we add *pixel-norm* [1] layers after each downsampling block. We observe that adding a pixel-wise normalization layer stabilizes the training while at the same time avoids mixing information between different pixels as

[*] Work performed during an internship at Google.



(a) initial rendering (b) depth map (c) original image

Figure 1: Sample frames of the aligned dataset. Even though interior structures can be seen through the walls in the point cloud rendering (bottom), neural rerendering is able to reason about occlusion among the points and thereby avoid rerendering artifacts. Image credits: James Manners, Patrick Denker (Creative Commons).

in *instance norm* or *batch norm*. We use a latent appearance vector $z^a \in \mathbb{R}^8$. The latent vector is injected at the bottleneck between the encoder and decoder in the rendering network. We tile z^a to match the dimension of feature maps at the bottleneck and concatenate it to the feature maps channel-wise.

B.3. Baseline architecture

We use a faithful Tensorflow implementation of the encoder-decoder network and appearance encoder in [2] using their PyTorch released code as a guideline. We adapt their training pipeline to the single-domain supervised setup as described in Section 3.2 in our paper.

B.4. Aligned datasets

Figure 1 shows sample frames from aligned datasets we generate as described in Section 3.1 in the paper.

B.5. Latent space visualization

Figure 5 visualizes the latent space learned by the appearance encoder, E^a, after appearance pretraining and finetuning in our staged training, as well as training E^a with the BicycleGAN baseline. The embedding learned during the appearance pretraining stage shows meaningful clusters, but has lower quality than the one learned after finetuning, which is comparable to the one of the BicycleGAN baseline.

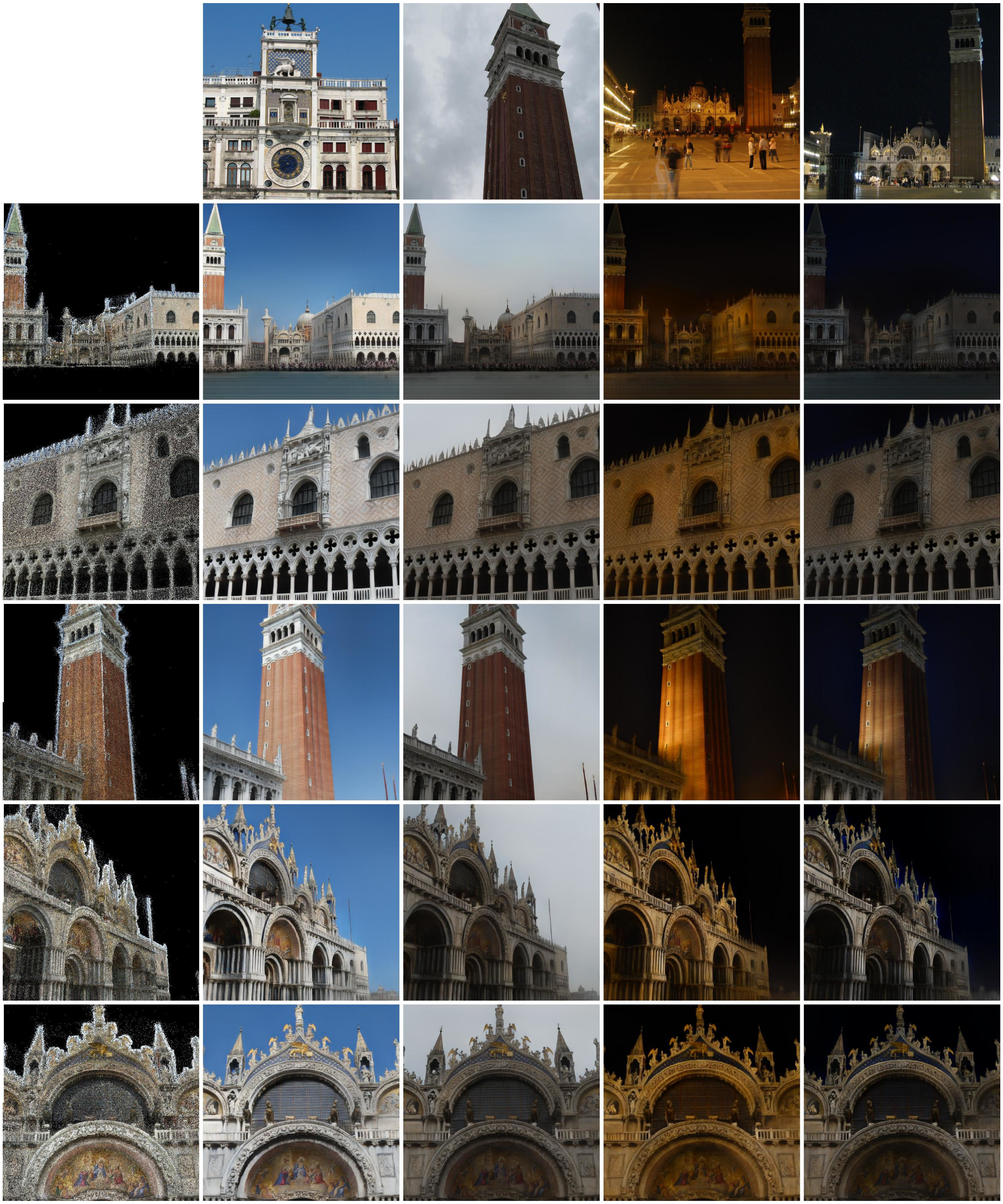


Figure 2: We capture the appearance of the original images in the first row, and rerender several viewpoints under them. The first column shows the rendered point cloud images used as input to the rerenderer. Image credits: Michael Pate, Jeremy Thompson, Patrick Denker, Rob Young (Creative Commons).

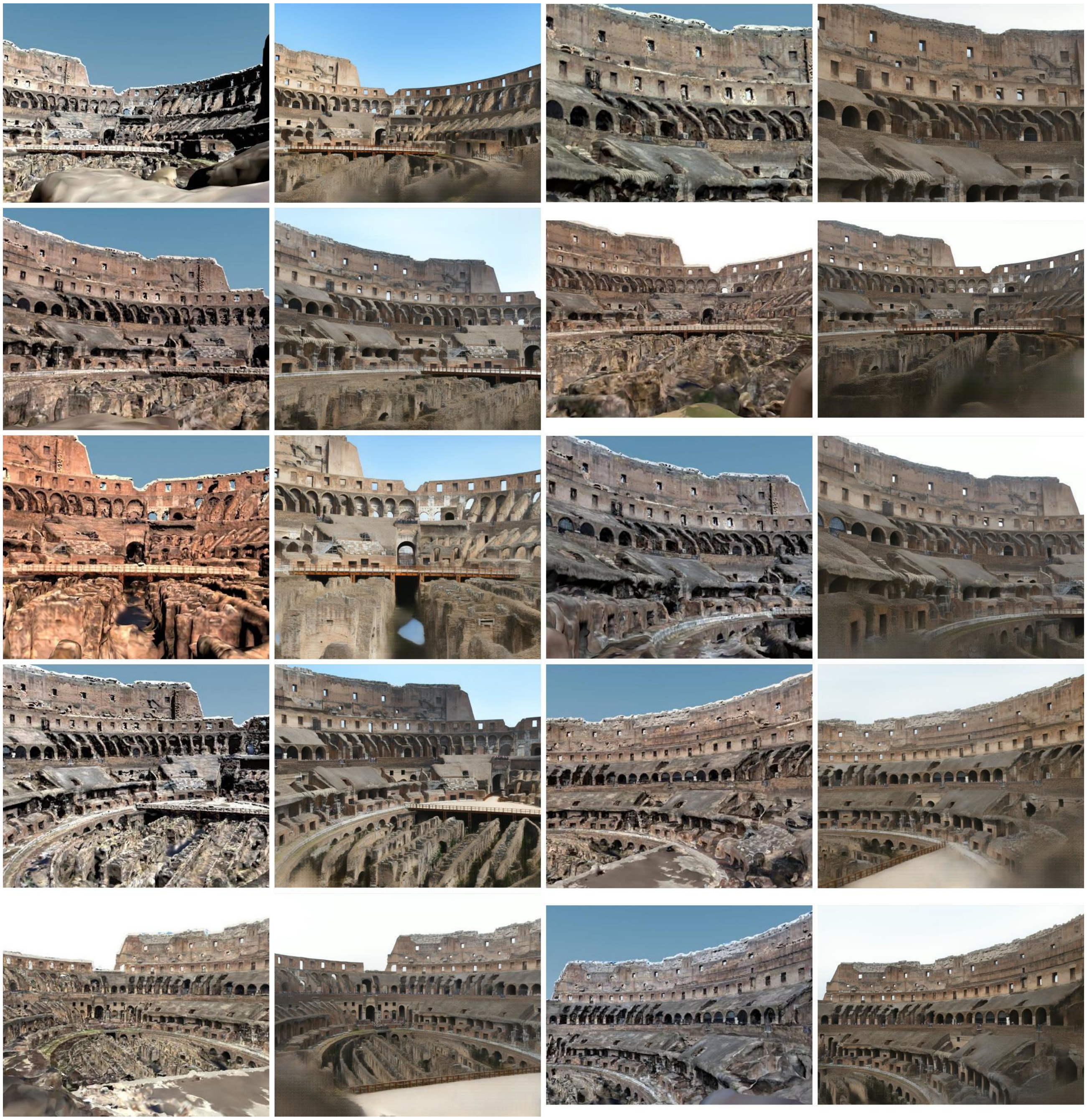


Figure 3: Comparison with Shan *et al*. [3] – set 1 of 2. First and third columns show the result of Shan *et al*. [3]. Second and fourth columns show our result.

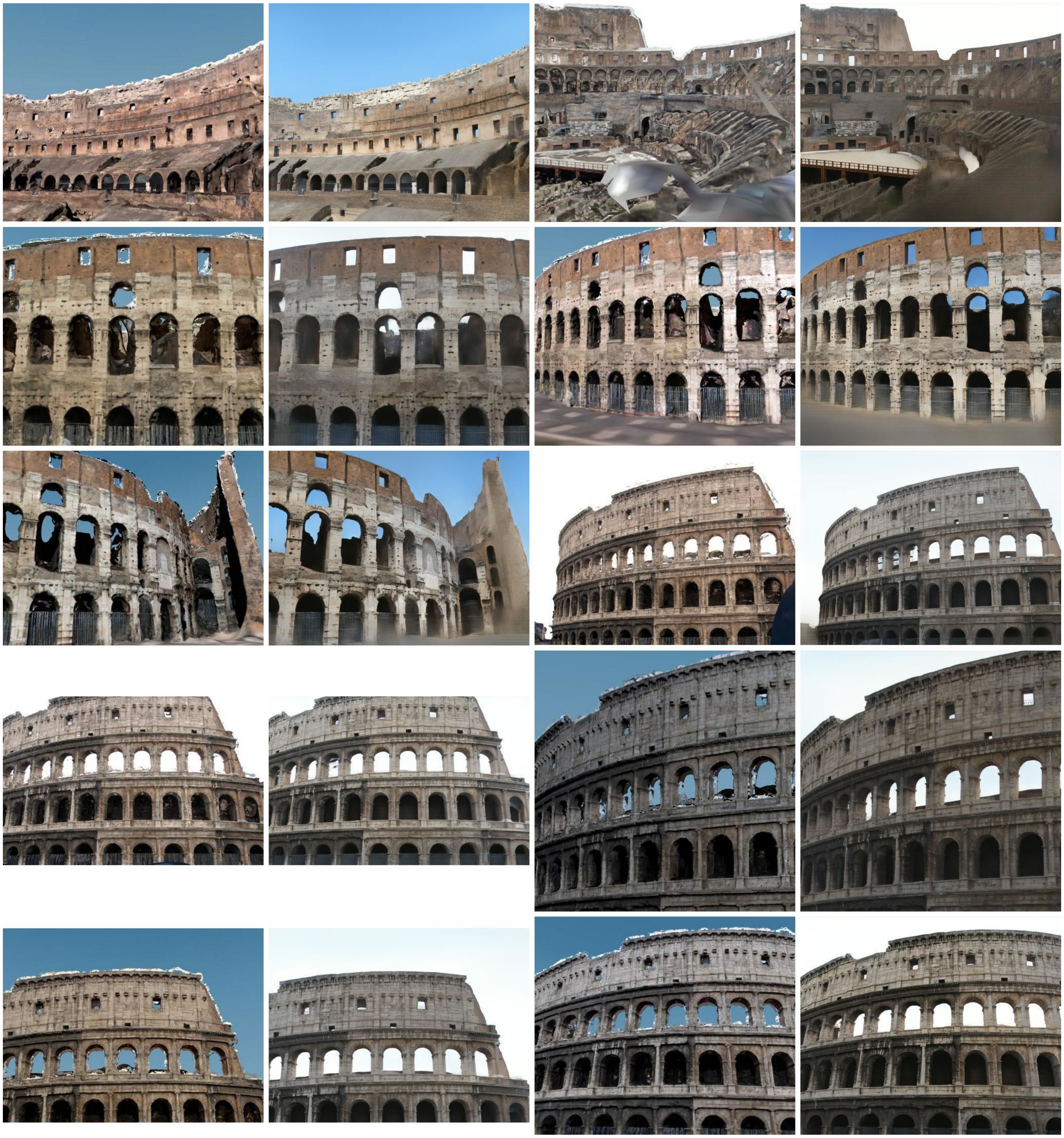
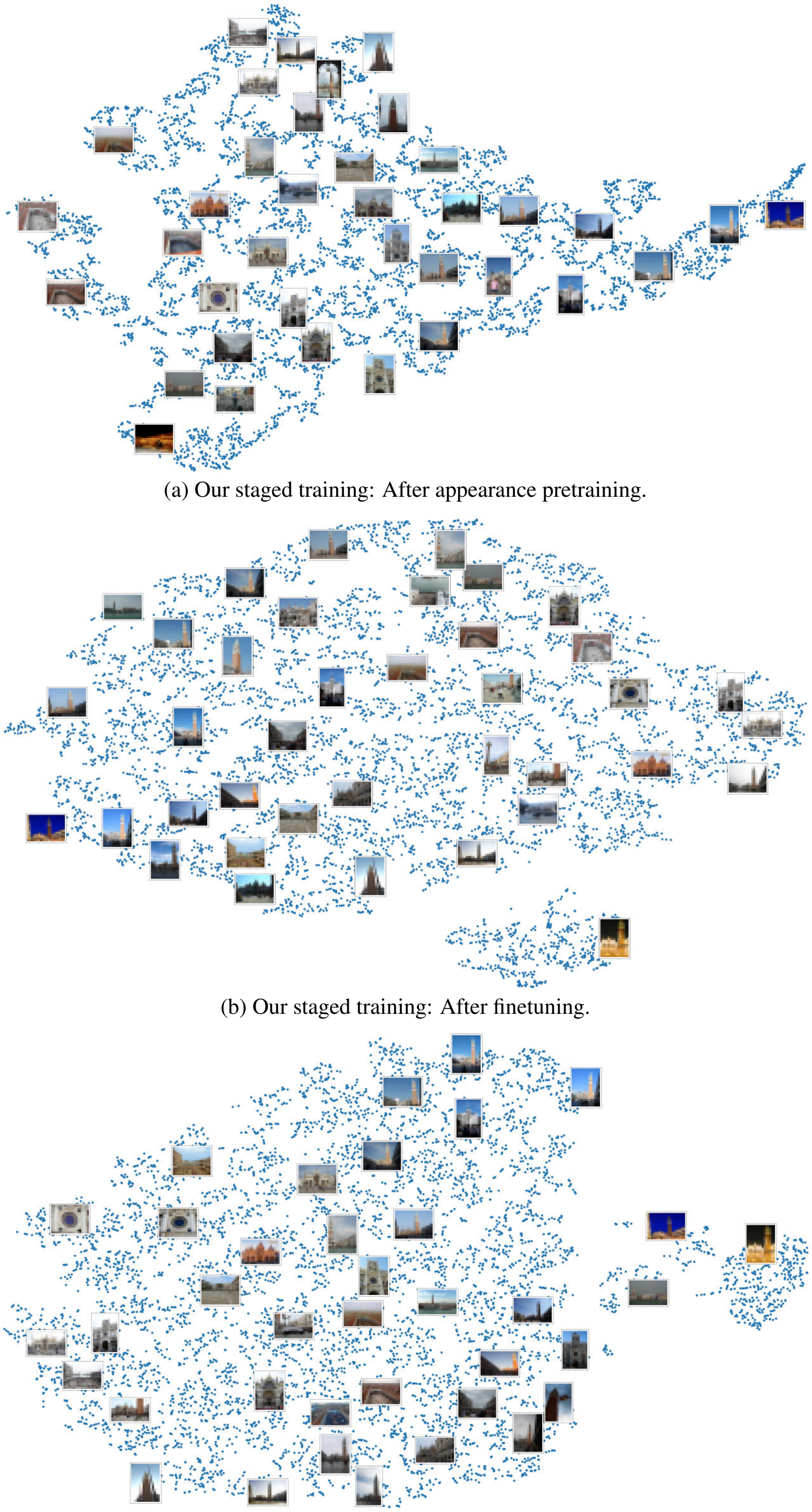


Figure 4: Comparison with Shan *et al*. [3] – set 2 of 2. First and third columns show the result of Shan *et al*. [3]. Second and fourth columns show our result.



(a) Our staged training: After appearance pretraining.

(b) Our staged training: After finetuning.

(c) BicycleGAN baseline.

Figure 5: t-SNE plots for the latent appearance space learned by the appearance encoder (a) after appearance pretraining in our staged training, (b) after finetuning in our staged training, and (c) using the BicycleGAN baseline.

References

[1] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 1

[2] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. K. Singh, and M.-H. Yang. Diverse image-to-image translation via disentangled representations. In *ECCV*, 2018. 1, 2

[3] Q. Shan, R. Adams, B. Curless, Y. Furukawa, and S. M. Seitz. The Visual Turing Test for scene reconstruction. In *Proc. 3DV*, 2013. 1, 4, 5