Supplemental: Learning a Neural 3D Texture Space from 2D Exemplars

1. Network Architecture

1.1. Encoder

The architecture for the encoder network remains consistent for both ours and competitor methods. Depending on training for *space*, *single*, *w/o transform* the parameter N changes accordingly.

Table 1. Network architecture for encoder.

Layer	Kernel	Activation	Shape	# params
Input	—	—	3 x 128 x 128	—
Conv	3x3	IN+LReLU	32 x 128 x 128	~1k
Conv	4x4	IN+LReLU	64 x 64 x 64	~32k
Conv	4x4	IN+LReLU	128 x 32 x 32	~130k
Conv	4x4	IN+LReLU	256 x 16 x 16	~524k
Conv	4x4	IN+LReLU	256 x 8 x 8	~1M
Conv	4x4	IN+LReLU	256 x 4 x 4	~1M
Linear	—	—	8	~32k
Linear	—	—	N	~0.5k
# params	—	—		~2.8M

1.2. Sampler

The sampler architecture used for both our and the *mlp* [1] method consists of following convolutional architecture with 1x1 kernels emulating Linear layers:

Table 2. Network architecture for sampler.

Layer	Kernel	Activation	Shape	# params
Input	—	—	N x 128 x 128	—
Conv	1x1	ReLU	128 x 128 x 128	~10k
Conv	1x1	ReLU	128 x 128 x 128	~16.5k
Conv	1x1	ReLU	128 x 128 x 128	~16.5k
Conv	1x1	ReLU	128 x 128 x 128	~16.5k
Conv	1x1	ReLU	128 x 128 x 128	~16.5k
Conv	1x1	ReLU	3 x 128 x 128	~400
# params	—	—		~77k

1.3. CNN

For *cnn* and *cnnD* competitors we use a similar architecture to the proposed method of [2]:

Table 3. Network architecture for convolutional methods.

Layer	Kernel	Activation	Shape	# params
Input	—	—	(32) + 256	—
Linear	—	—	(32) + 256	~80k
Linear	—	—	256	~70k
Reshape	—	—	16 x 4 x 4	—
ConvT	4x4	ReLU	128 x 8 x 8	~32k
ConvT	4x4	ReLU	128 x 16 x 16	~260k
ConvT	4x4	ReLU	128 x 32 x 32	~260k
Upsample	—	—	128 x 64 x 64	—
Conv	3x3	ReLU	64 x 64 x 64	~70k
Upsample	—	—	64 x 128 x 128	—
Conv	3x3	ReLU	3 x 128 x 128	~2k
# params	—	—		~790k

2. Results

Additional results are displayed below. Furthermore, a webpage containing more results for all four classes (WOOD, MARBLE, GRASS and RUST) including competitors can be accessed online: `https://geometry.cs.ucl.ac.uk/projects/2020/neuraltexture`. Videos of rotating shapes textured by our method can be found in the *videos* folder.

Figure 1. Results derived from the encoded WOOD space.

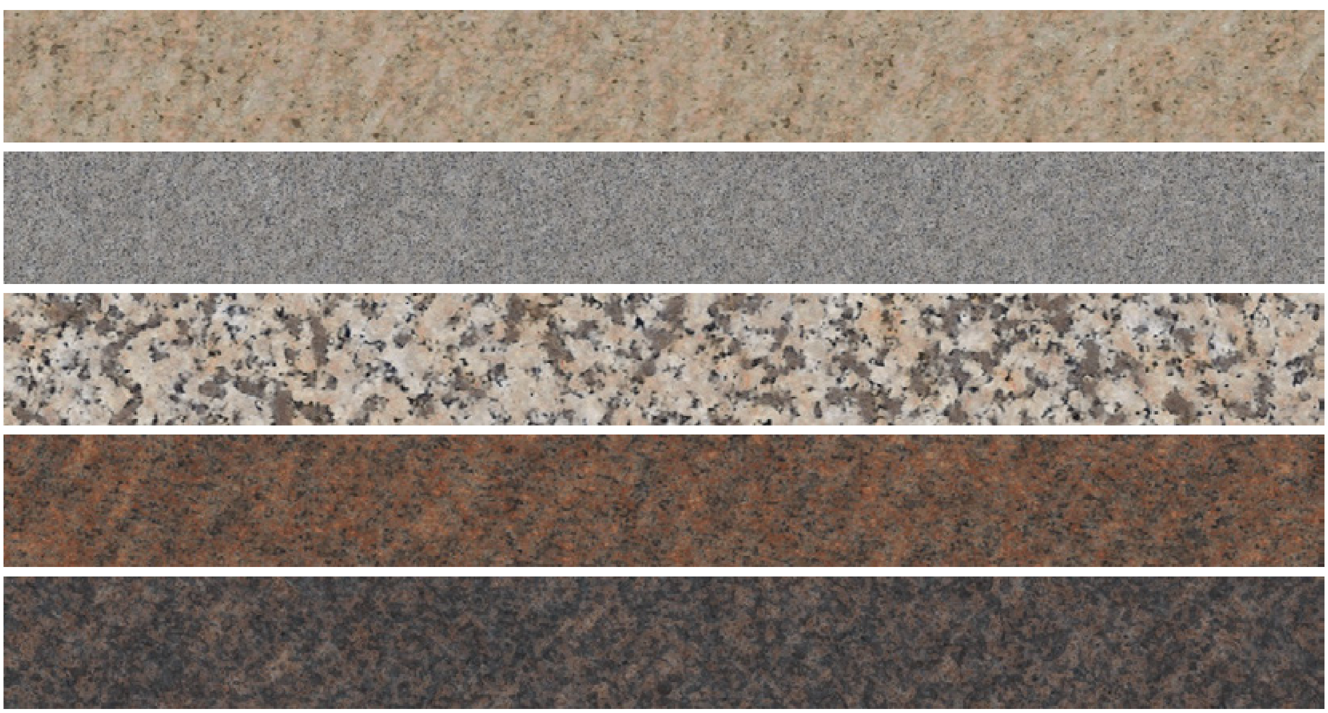

Figure 2. Results derived from the encoded MARBLE space.

Figure 3. Results derived from the encoded GRASS space.

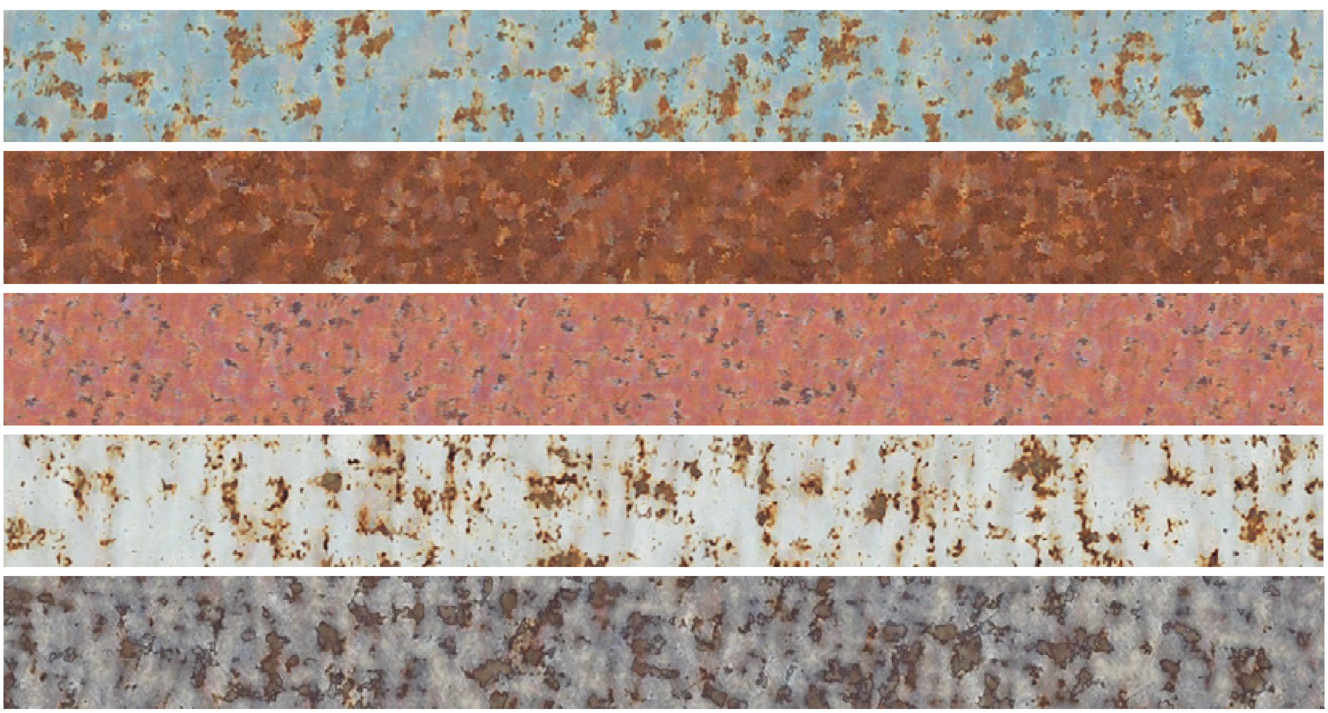

Figure 4. Results derived from the encoded RUST space.

Figure 5. Latent space interpolation from one ground truth wood exemplar (left) into secondary ground truth exemplar (right). Each row corresponds to independent interpolations.

Figure 6. Latent space interpolation from one ground truth grass exemplar (left) into secondary ground truth exemplar (right). Each row corresponds to independent interpolations.

Figure 7. Latent space interpolation from one ground truth marble exemplar (left) into secondary ground truth exemplar (right). Each row corresponds to independent interpolations.

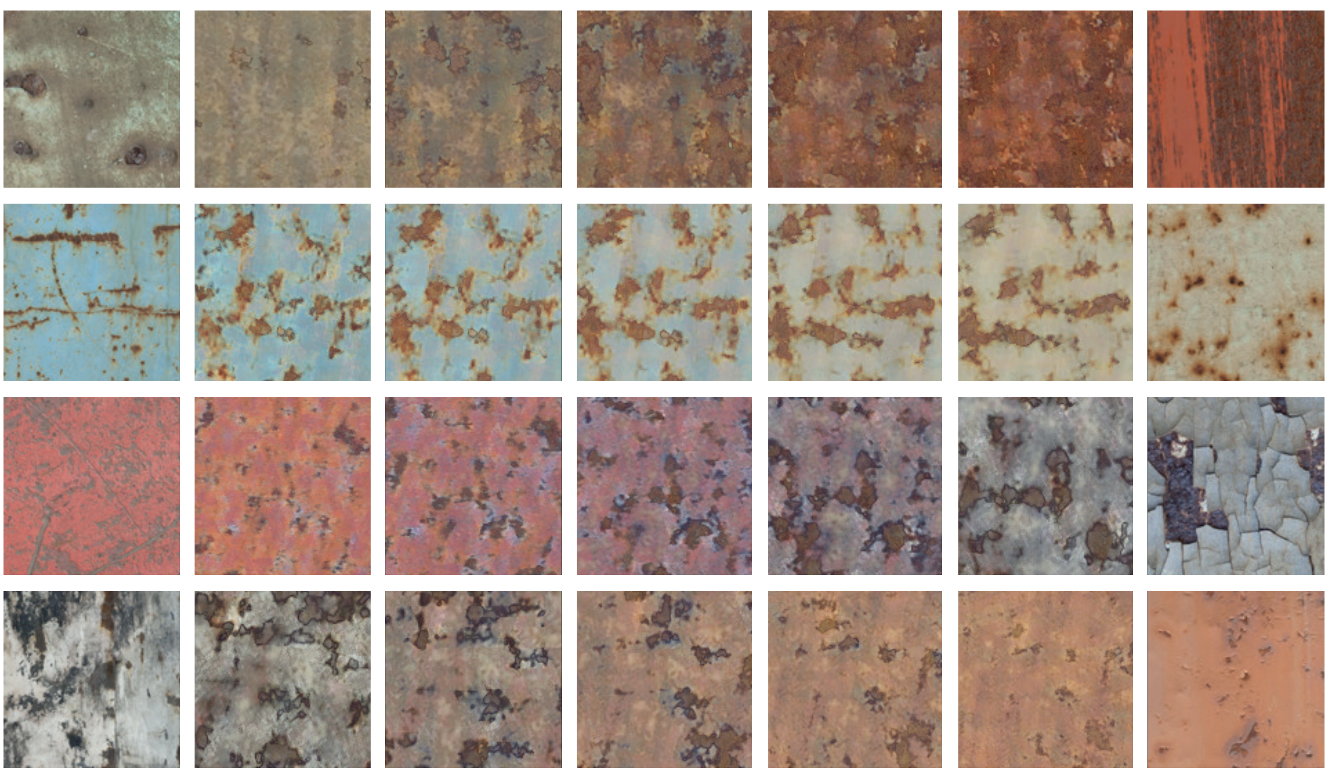

Figure 8. Latent space interpolation from one ground truth rust exemplar (left) into secondary ground truth exemplar (right). Each row corresponds to independent interpolations.

References

[1] Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *ICCV*, 2019. 1

[2] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *CVPR*, 2017. 1