Supplementary Material: Controlling Perceptual Factors in Neural Style Transfer

Leon A. Gatys
University of Tübingen

Alexander S. Ecker
University of Tübingen

Matthias Bethge
University of Tübingen

Aaron Hertzmann
Adobe Research

Eli Shechtman
Adobe Research

November 23, 2016

1 Spatial Control

This section gives further details on section 4 in the main text. Sections 1.1 and 1.2 of this supplement are referenced by section 4.1 and section 4.2 in the main text.

1.1 Propagation of guidance channels to feature maps

As described in the main text, due to increasing stride and receptive field sizes as well as the quadratic shape of the receptive fields, spatial resolution in the upper layers of the CNN is limited. Therefore care needs to be taken when propagating an accurate spatial mask defined on the image pixels into a spatial mask defined on a feature map in the network. We found three strategies generally useful for this propagation:

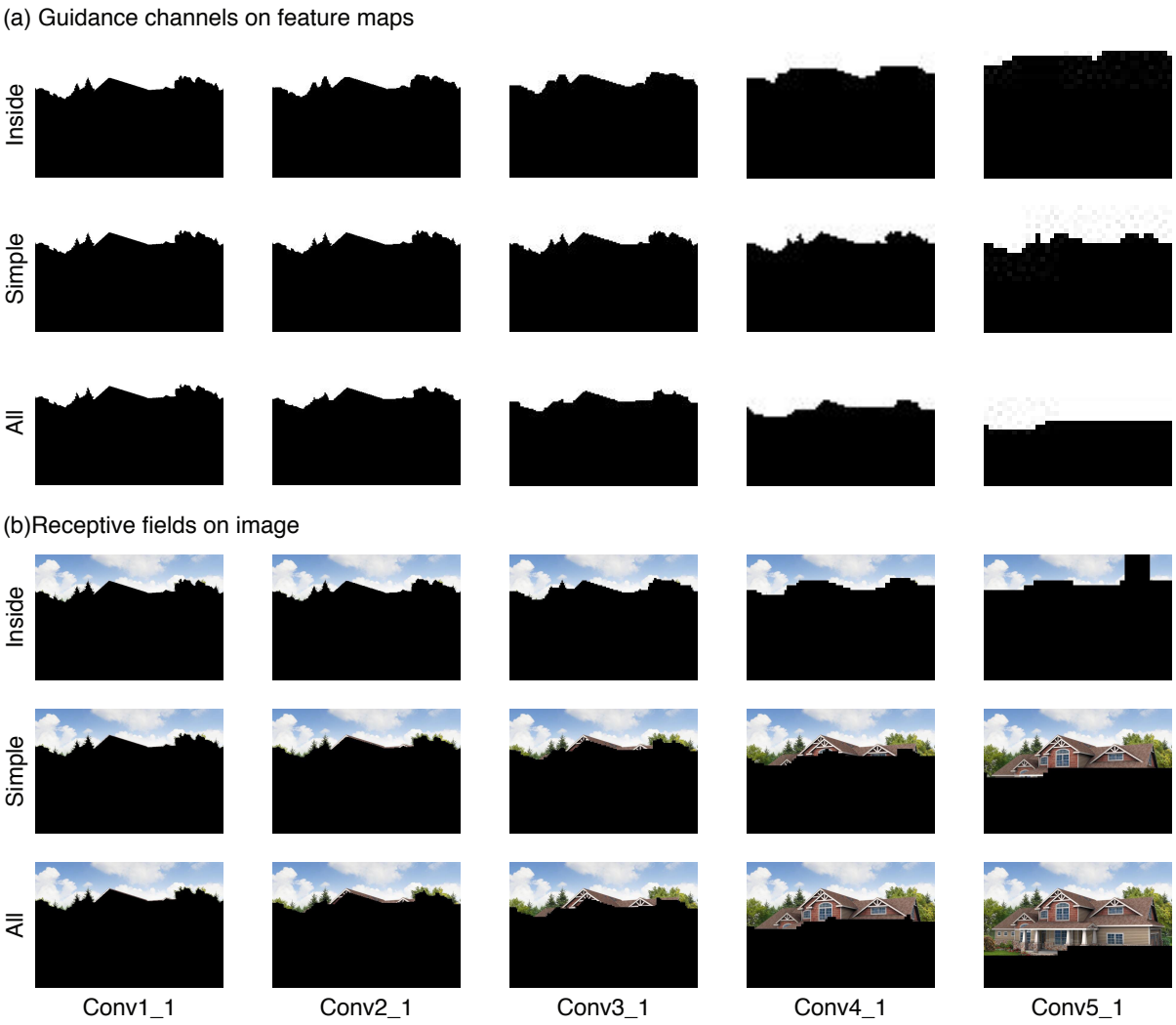
1. Propagate guidance channels only to neurons whose receptive field is entirely inside the guidance region ('Inside')

2. Simply propagate the guidance channel down-sampled to the size of the feature map ('Simple').

3. Propagate the guidance channel to all neurons that overlap with the guidance region ('All')

The propagated guidance channels produced with all three strategies for the sky region of the content image of Fig. 2(a) in the main text are shown in Supp. Fig. 1. The actual guidance channels on the feature maps of different layers are shown in Supp. Fig. 1(a). The effective guidance on the image is shown in Supp. Fig. 1(b). It shows the part of the images covered by the receptive fields of the guided neurons in the different layers. The general problem becomes apparent: If simply down-sampling the feature maps, the receptive fields of neurons in higher layers overlap substantially with adjacent guidance regions.

The results for each of these strategies are shown in Supp. Fig. 2. 'Inside' guidance does not stylise the boundary regions properly (Supp. Fig. 2(a)). With 'Simple' and 'All' guidance the ground texture is leaking into the sky, leading to artefacts (Supp. Fig. 2(b),(c)). However, we can change these initial guidance channels with erosion and dilation to get a good solution for most images. The best strategy usually depends on the particular content image and regional guidance used. For this example, the best results we could achieve with dilation/erosion adjustment are shown in Supp. Fig. 2 (d) and (e). Probably the best result is given by using the 'inside' guidance propagation and then dilating the guidance channels on each feature map with a square kernel of size 7×7 giving a dilation of 3 (Supp. Fig. 2 (d)).



Supp. Fig. 1: **(a)** The propagated guidance channels for all three strategies ('Inside', 'Simple', 'All'). **(b)** The effective guidance on the image: The part that is inside of the receptive fields of the guided neurons. On higher layers the resolution decreases: For 'Simple' and 'All' a considerable part of the ground region is inside the receptive fields of the guided neurons. For 'Inside' the receptive fields of the guided neurons miss parts of the guidance region near the boundary.

(a) 'Inside' guidance

(b) 'Simple' guidance

(c) 'All' guidance

(d) 'Inside' guidance, dilation: 3

(e) 'Simple' guidance, erosion: 1

Supp. Fig. 2: **(a)** Spatial Control with 'Inside' guidance propagation. The boundary between sky and ground is not stylised well. **(b)** Spatial Control with 'Simple' guidance propagation. Ground texture is leaking into the sky region. **(c)** Spatial Control with 'All' guidance propagation. More ground texture is leaking into the sky region. **(d)** 'Inside' guidance propagation with dilation of 3. Boundary region is stylised. **(c)** 'Simple' guidance propagation with erosion 1. Less ground texture is leaking to the sky region.

Another solution that does not require to manually erode or dilate the guidance channels but often works well is to combine spatial guidance with the 'Inside' strategy with the original unguided algorithm. In that way spatial guidance is enforced for all neurons that have a clear assignment to a guidance region. At the same time, the neurons that overlap with the region boundary retain the greater flexibility of the unguided stylisation loss and can in that sense 'pick' the best matching image structures from the whole style image to stylise the boundary region. Fig. 2(e) in the main text was generated using that method.

Nevertheless, in some cases just simple down-sampling may give decent results. For example, the guidance maps for Fig. 2(f) in the main text were generated in that way.

1.2 Comparison with guided sums

As described in section 4.2 in the main text, stacking the guidance channels onto the feature maps and computing one Gram Matrix for all of them is equivalent to combining the global style loss with regionally guided feature map sums. The contribution to the style loss of a particular layer is then:

$$E_\ell = \frac{\lambda_{global}}{4N_\ell^2} \sum_{ij} \left(\mathbf{G}_\ell(\hat{\mathbf{x}}) - \mathbf{G}_\ell(\mathbf{x}_S)\right)_{ij}^2 + \frac{1}{2N_\ell} \sum_{r=1}^{R} \lambda_r \sum_i \left(\langle \mathbf{F}_\ell^r(\hat{\mathbf{x}})\rangle - \langle \mathbf{F}_\ell^r(\mathbf{x}_S)\rangle\right)_i^2 \tag{1}$$

where

$$\langle \mathbf{F}_\ell^r(\mathbf{x})\rangle_i = \frac{1}{M} \sum_j \mathbf{F}_\ell^r(\mathbf{x})_{ij} \tag{2}$$

is the vector of guided feature map means and the weights of λ_{global} and $\lambda_r, r \in R$ controls the strength of the spatial guidance.

A comparison between spatial control with guided Gram Matrices and spatial control with guided sums is shown in Supp. Fig. 3. Both methods nicely achieve the separation between sky and ground stylisation and give decent results (Supp. Fig. 3 (d)-(g)). However, using guided sums fails to capture the painterly texture of the style image quite as well as when using guided Gram Matrices. For example, the brush-strokes of the style images are not captured as well or the colours of the grass and sky regions look different to those in the style image when using guided sums (Supp. Fig. 3 (e),(g)).

2 Colour Control

This section gives further details on section 5 in the main text. Section 2.1 in the supplement is referenced by section 5.3 in the main text and section 2.2 in the supplement refers to section 5.2 in the main text.

2.1 Colour preservation

For more details on preserving colour in Neural Style Transfer, please see the Technical Report *Preserving Color in Neural Artistic Style Transfer* published at *arxiv.org/abs/1606.05897* and the additional example images attached.

2.2 Preserving colour of the style image

Empirically, results often look better when initialising the optimisation procedure from the content image instead of from white noise. If there is a strong mismatch in colour information between content and style images, the optimisation procedure can converge to unappealing local minima if it is initialised from the content image (Supp. Fig. 4(d),(i)). In many cases, rather than preserving the colour of the content image, one would like to preserve the colour distribution of the style image (eg. for pencil drawings and line-art). This can be done exactly as described in section 5.2 of the main text, only that this time we

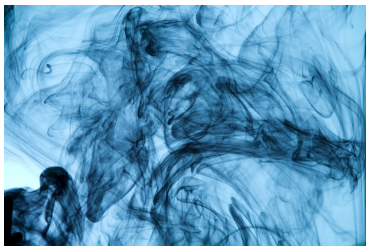
(a) Content (b) Style I (c) Style II

(d) Output with guided Gram Matrices (e) Output with guided sums

(f) Mix styles with guided Gram Matrices (g) Mix styles with guided sums

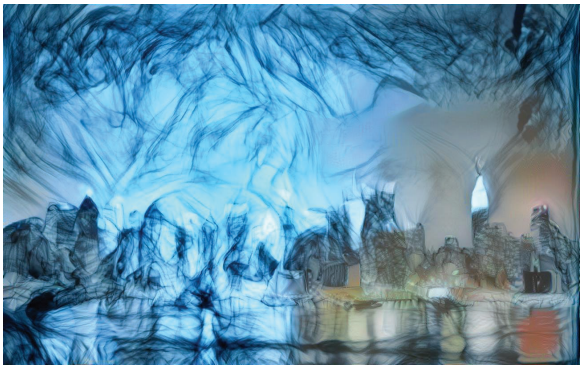
Supp. Fig. 3: Comparison between guided Gram Matrices and guided sums for spatial control. **(a)** Content image. **(b)** Style image I. **(c)** Style image II. **(d)** Output using style I with guided Gram Matrices. **(e)** Output using style I with guided sums. **(f)** Output using style I for the ground and style II for the sky region with guided Gram Matrices. **(g)** Output using style I for ground and style II for the sky region with guided sums.

(a) Content I

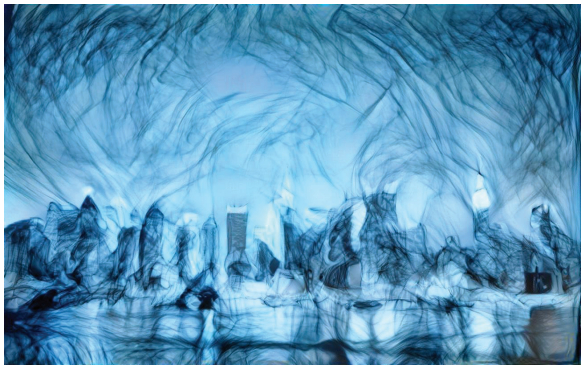


(b) Style I

(c) Colour-matched Content I

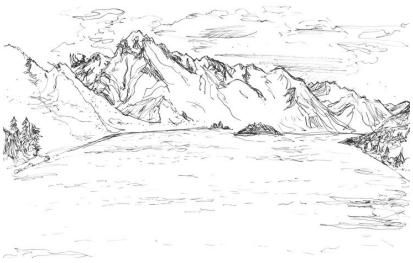


(d) Output using Content I



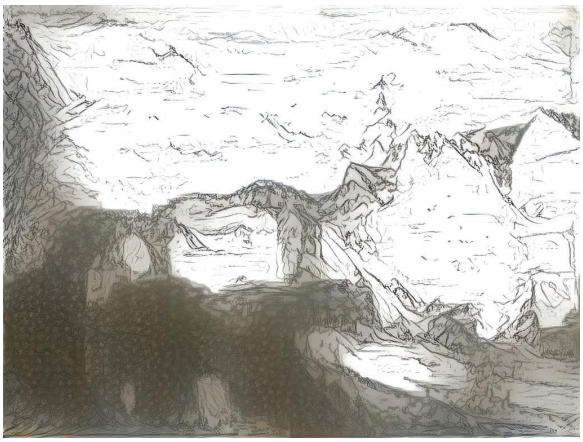
(e) Output using colour-matched Content I

(f) Content II

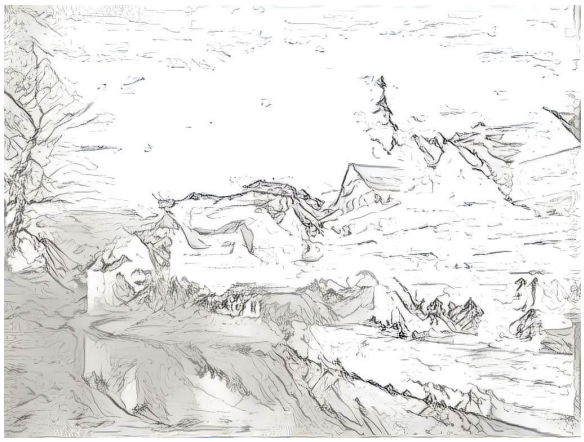


(g) Style II

(h) Colour-matched Content II

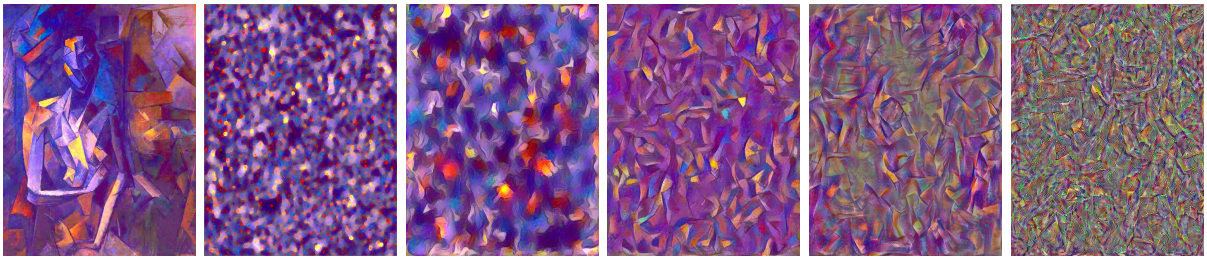


(i) Output using Content II



(j) Output using colour-matched Content II

Supp. Fig. 4: Improved colour transfer from style to content image. **(a)** Content image I. **(b)** Style image I. **(c)** Content image I after colour transfer from style image I. **(d)** Output using content image I and style image I. **(e)** Output using colour-matched content image I and style image I. **(f)** Content image II. **(g)** Style image II. **(h)** Content image II after colour transfer from style image II. **(i)** Output using content image II and style image II. **(j)** Output using colour-matched content image II and style image II.



Supp. Fig. 5: Visualising style features from individual layers. **(a)** Source image. **(b)** Image generated by matching Gram Matrix only on layer "conv1_1". **(c)** Same using layer "conv2_1". **(c)** Same using layer "conv3_1". **(c)** Same using layer "conv4_1". **(c)** Same using layer "conv5_1".

transfer the colour histogram from the style image onto the content image before stylisation. We found that this can substantially improve stylisation results in cases where there is a strong mismatch in colour information between content and style images (Supp. Fig. 4(e),(j)).

3 Scale Control

This section gives further details on section 6 in the main text. Sections 3.1, 3.2 and 3.3 in the supplement are referenced by section 6.1 in the main text.

3.1 Naive scale mixing

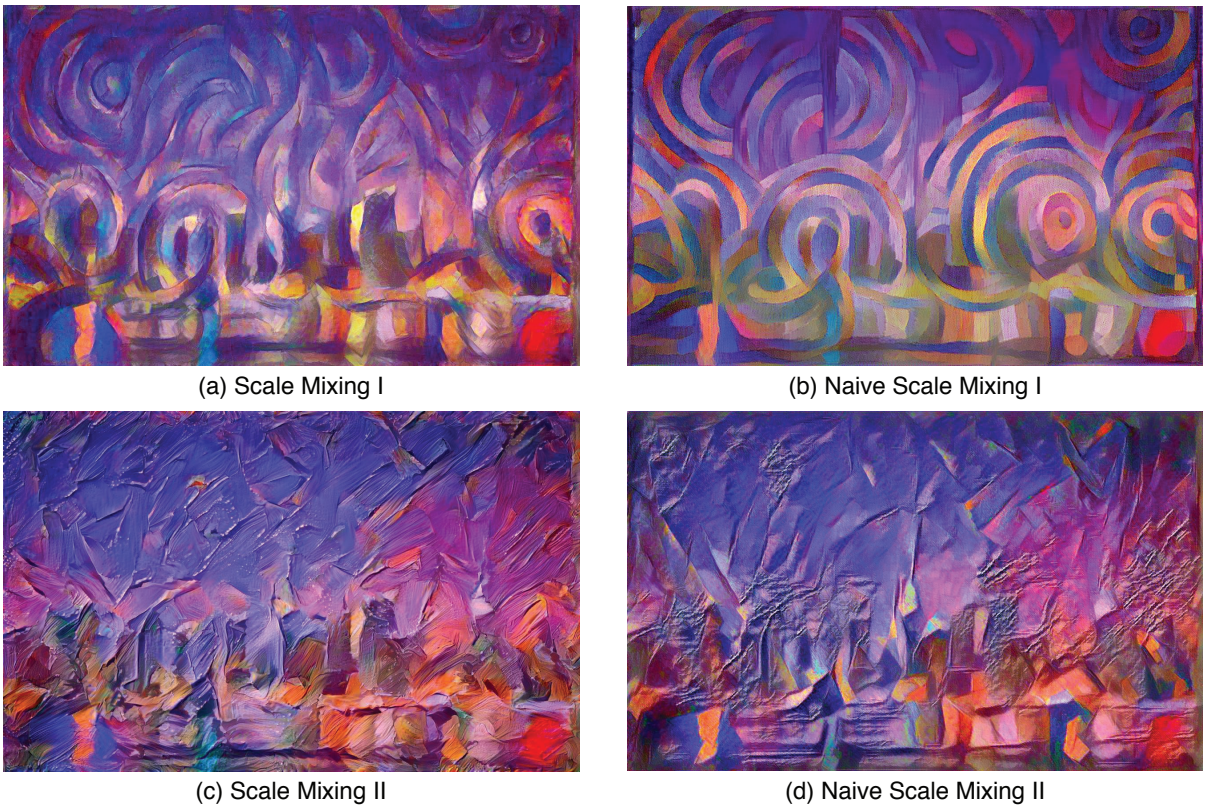
Here we show that the style features on different layers share a lot of low-level information and how that prevents a naive approach of mixing styles on different scales.

To show that, for example, the Gram Matrix at layer "conv2_1" shares information with the Gram Matrix at layer "conv4_1", we generate images that match the Gram Matrix only from a *single* layer (Supp. Fig. 5). We see that up to layer "conv4_1" most of the low-level information is preserved (Supp. Fig. 5(e)).

Now say we want to mix the the fine scale style of image A with the coarse scale style of image B. The naive approach would be to simultaneously match the Gram Matrices at lower layers (e.g. "conv1_1" ,"conv2_1") from image A and the Gram Matrices at higher layers (say "conv3_1" ,"conv4_1","conv5_1") from image B. We compare this naive approach to the one from section 6.1 in the main text in Supp. Fig. 6. As expected, we find that naive scale mixing does not preserve the fine scale style from style A (Supp. Fig. 6 (b),(d)). The painterly fine scale texture is lost in Supp. Fig. 6 (b) and the dominant brushstrokes are mostly gone in Supp. Fig. 6 (d). This is because the higher layers also capture much of the fine scale image information so that the naive approach does not give a stylisation with A on the fine an B on the coarse scale, but rather a mixture between A and B on the fine scale and B on the coarse scale. Hence the naive approach fails do give independent control on stylisation at different scales.

3.2 Scale interpolation

Controlling the stylisation independently on different scales also allows for new ways of interpolating between styles. We can start by using all scales from a style A and then replacing increasingly larger subsets of scales by another style B. So first we generate an image that uses all scales from style A (Supp. Fig. 7(d)). Next we use the finest scale (layer "conv1_1") from style B and all coarser scales (higher layers) from style A (Supp. Fig. 7(e)). Then we can iteratively increase the size of the subset of scales that are taken from style B. So next we use the two finest scales (layer "conv1_1", "conv1_2") from style B



(a) Scale Mixing I

(b) Naive Scale Mixing I

(c) Scale Mixing II

(d) Naive Scale Mixing II

Supp. Fig. 6: Naive scale mixing. **(a)** Scale mixing from Fig. 4(f) in the main text. **(b)** Naive scale mixing for the same example as in (a). **(c)** Scale mixing from Fig. 4(e) in the main text. **(d)** Naive scale mixing for the same example as in (c).

Supp. Fig. 7: Interpolating between two styles along spatial scale. **(a)** Content image. **(b)** Style image I. **(c)** Style image II. **(d)-(i)** Starting from stylisation with style I and replacing larger subsets of scales with style II. **(j)-(o)** Starting from stylisation with style II and replacing larger subsets of scales with style I. Scales are replaced from fine to coarse.

and the rest from style A. Then we use the three finest scales (layer "conv1_1", "conv1_2", "pool1') from style B and the rest from style A and so on, until finally, we use all layers from style B (Supp. Fig. 7(i)). We show snapshots of this interpolation at layers "conv1_1", "conv2_1", "conv3_1" and "conv4_1" in Supp. Fig. 7. An animation of the full interpolation is attached. Generally the finer scales have a much stronger perceptual effect than the coarser scales. That means that most of the interpolation happens when replacing the early layers from style A with style B. To encourage continuity in the interpolation, we always initialise the optimisation procedure from the mixing at the previous scale. This is why the results in Supp. Fig. 7(i) and (j) and (d) and (o) are different.

3.3 New styles with scale mixing

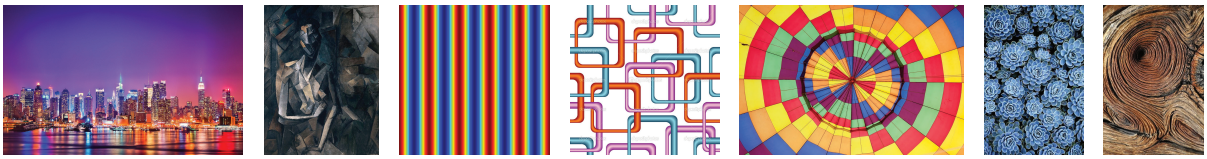
By mixing fine and coarse scales of existing styles, one can generate a large number of new styles. In particular we found that one can combine painterly styles for the fine scale with arbitrary textures for the coarse scale. In that way one can use the same painterly structures such as brushstrokes or material properties while at the same time manipulating the appearance of stylisations in a controlled fashion. A larger collection of examples for the images in Figure 4 of the main text are given in Supp. Fig. 8 and 9.

4 Controlling Fast Neural Style Transfer

This section gives further details on section 7 in the main text.

All networks we trained had the same architecture. We used the default settings of the public implementation [1] with the exception that we halved the number of feature maps in each layer as can be done when using Instance Normalization [2]. Here are the listed parameters:
arch: 'c9s1-16,d32,d64,R64,R64,R64,R64,R64,u32,u16,c9s1-3'
backend: 'cuda'
batch_size: 4
content_layers: ['16']
content_weights: [1]
learning_rate: 0.001
loss_network: 'models/vgg16.t7'
lr_decay_every: -1
lr_decay_factor: .5
max_train: -1
num_iterations: 40000
padding_type: 'reflect-start'
percep_loss_weight: 1
pixel_loss_weight: 0
preprocessing: 'vgg'
style_image_size: 256
style_layers: ['4', '9', '16', '23']
style_target_type: 'gram'
style_weights: [5, 5, 5, 5]
tanh_constant: 150
task: style
tv_strength: 1e-06
use_cudnn: 1
use_instance_norm: 1
weight_decay: 0

[1] github.com/jcjohnson/fast-neural-style



(a) Content (b) Style I (c) Style II/III/IV/V/VI

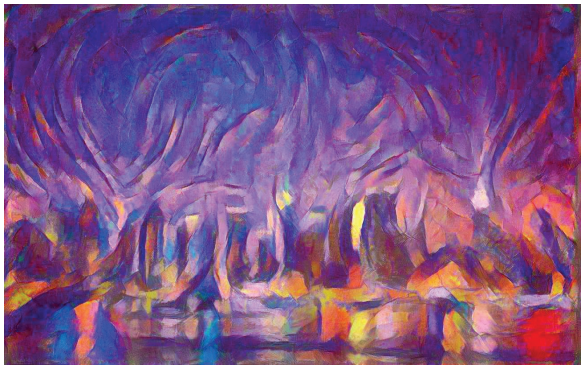
(d) Output using only style I

(e) Output scale mixing style I/style II

(f) Output scale mixing style I/style III

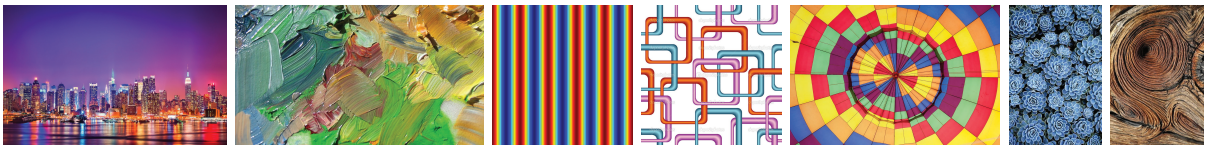
(g) Output scale mixing style I/style IV

(h) Output scale mixing style I/style V

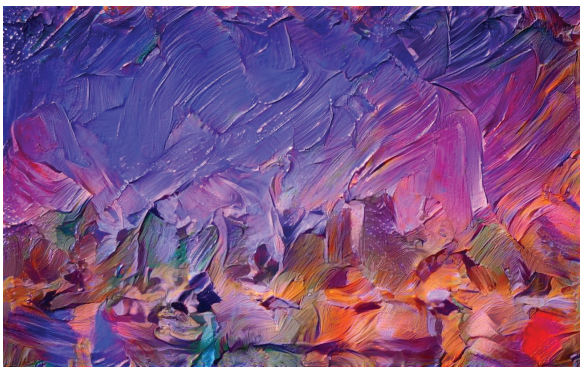


(i) Output scale mixing style I/style VI

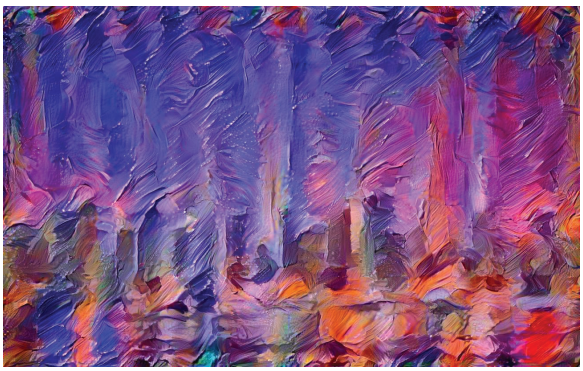
Supp. Fig. 8: More examples of new style combinations. **(a)** Content image. **(b)** Style image I. **(c)** Style images II-VI. **(d)** Output using only style I for both fine and coarse scale. **(e)** Output using style II for coarse scale. **(f)** Output using style III for coarse scale. **(g)** Output using style IV for coarse scale. **(h)** Output using style V for coarse scale. **(i)** Output using style VI for coarse scale. **(e)-(i)** All use style I for fine scale. Colour is preserved in all images using the colour histogram transfer method.



(a) Content (b) Style I (c) Style II/III/IV/V/VI

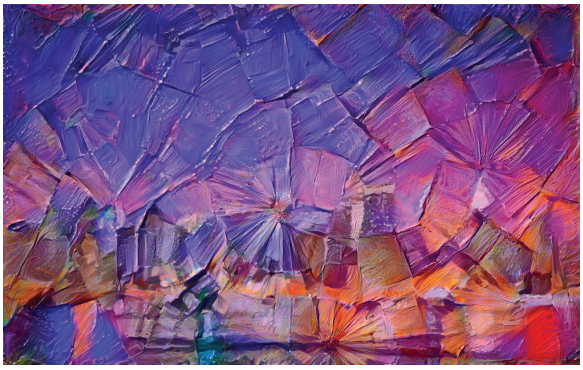


(d) Output using only style I



(e) Output scale mixing style I/style II

(f) Output scale mixing style I/style III



(g) Output scale mixing style I/style IV

(h) Output scale mixing style I/style V

(i) Output scale mixing style I/style VI

Supp. Fig. 9: More examples of new style combinations. **(a)** Content image. **(b)** Style image I. **(c)** Style images II-VI. **(d)** Output using only style I for both fine and coarse scale. **(e)** Output using style II for coarse scale. **(f)** Output using style III for coarse scale. **(g)** Output using style IV for coarse scale. **(h)** Output using style V for coarse scale. **(i)** Output using style VI for coarse scale. **(e)-(i)** All use style I for fine scale. Colour is preserved in all images using the colour histogram transfer method.

5 Parameters used for figures

For Figures 1,2,3 and 5 in the main text and Supp. Figures 2,3,4, we used the original VGG19 network. The content weight for layer conv4_2 was equal to 1 and the style weights for layers conv1_1, conv2_1, conv3_1, conv4_1, conv5_1 were equal to $10^3/N_\ell^2$ where N_ℓ is the number of feature maps in that layer. These settings appeared to work well for a large number of input images. They pronounce the role of layer conv4_1 and thus emphasise stylisation on a mid-level scale.

For Figure 4 in the main text and Supp. Figures 5,6,7,8,9 we used the normalised VGG19 network from [1], the content weight equal to 10^5 and all style weights equal to 2×10^9. These settings increase the transfer of coarse scale image structures and thus emphasise the effect of scale mixing. The large values of the content and style weights for the normalised network are chosen to avoid optimisation issues: The L-BFGS implementation we used is not invariant to rescaling of the loss function and for small loss values the optimisation often gets stuck. Since the normalised network's activations are smaller and so are the loss values compared to the original network, we have to pick larger weights for content and style to increase the scale of the loss values. All stylisations were initialised from the content image, as this generally gives cleaner results. Usually the L-BFGS optimisation was run for 500 iterations in low-resolution and then for another 200 iterations in high-resolution as described in section 6.2 in the main text.

References

[1] L. A. Gatys, A. S. Ecker, and M. Bethge. Image Style Transfer Using Convolutional Neural Networks. In *Proc. CVPR*, 2016.

[2] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance Normalization: The Missing Ingredient for Fast Stylization. *arXiv:1607.08022 [cs]*, July 2016. arXiv: 1607.08022.