MODULAR AND LIGHTWEIGHT NETWORKS FOR BI-SCALE STYLE TRANSFER

Thibault Durand*, Julien Rabin and David Tschumperlé

Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France

ABSTRACT

With the emergence of deep perceptual image features, style transfer has become a popular application that repaints a picture while preserving the geometric patterns and textures from a sample image. Our work is devoted to the combination of perceptual features from multiple style images, taken at different scales, e.g. to mix large-scale structures of a style image with fine-scale textures. Surprisingly, this turns out to be difficult, as most deep neural representations are learned to be robust to scale modifications, so that large structures tend to be tangled with smaller scales. Here a multi-scale convolutional architecture is proposed for bi-scale style transfer. Our solution is based on a modular auto-encoder composed of two lightweight modules that are trained independently to transfer style at specific scales, with control over styles and colors.

Index Terms— Multiple Style Transfer; Color transfer; Lightweight Neural Network; Texture Synthesis/Mixing;

1. INTRODUCTION

Style transfer (ST) usually consists in modifying a “content” image to embed visual characteristics from an example “style” image. Early ST methods have been based on the comparison of local-representations of images, such as patch [2, 3] or wavelets coefficients [4]. Since the seminal work of Gatys et al. [5] for texture synthesis, state-of-the-art ST methods are nowadays based on deep neural networks, in particular to extract “perceptual” features. They are either pre-trained on a subordinate visual recognition task (e.g. texture synthesis [5]), or used to drive the optimization [6]. In addition, generative networks [7, 8] or auto-encoders [9] may be also trained to generate the stylized image.

Here, we deal with the combination of different styles to transfer, each having a different geometric scale. Surprisingly, this topic has been little studied in the literature and mainly for texture interpolation and blending [10, 11]. With these methods, new visual features are synthesized by the interpolation of multiple texture features, rather than simultaneously exhibit them. Unfortunately, as shown for instance in [12], using the original perceptual loss in [5] for multiple styles fails in mixing them and results in images with distinctive styles. A popular approach to circumvent this issue is to use optimal transport framework to compute the average of perceptual features (see e.g. [13, 12]).

In this paper, we propose an original solution for combining geometric features at different scales for ST, which means simultaneously modifying an input image so that its overall geometric structure is preserved while incorporating coarse details from one style image, mixed with the fine details from a second style image, as illustrated in Fig. 1. This has been proved to be a difficult task in [14], as structures at different scales are tangled in deep encoders representations.

An overview of the literature on image ST is presented (Section 2) with a focus on [14] which aims at combining styles at different scales. In Section 3, we propose a new modular

*This work has been supported by Région Normandie under grant RIN.
alternative architecture, composed of two networks that better captures the geometric features of multiple styles for ST. It requires very few parameters (~155k) compared to concurrent methods, enabling fast and independent training. Experiments on bi-scale style transfer and texture synthesis are finally conducted in Section 4.

2. PREVIOUS WORK AND MOTIVATIONS

2.1. Style Transfer With Perceptual Loss

Throughout the paper, we consider the following perceptual loss introduced in [14] for texture synthesis and in [15] for style transfer (in an iterative image data optimization process), and used to train feed-forward networks in [9]:

\[
L_{\omega, \gamma}(C, S, Y) = \sum_{l} \omega_l \|\phi_l(I) - \phi_l(Y)\|^2 + \sum_{l} \gamma_l \|G(\phi_l(S)) - G(\phi_l(Y))\|^2 \tag{1}
\]

where \(\|\cdot\|\) stands as the Frobenius norm, and \(\phi_l(\cdot)\) corresponds to the normalized feature maps at the \(l\)-th layer of VGG-19 [16] (often referred to ReLu_11, ReLu_21, ReLu_31, ReLu_41, ReLu_51). Let us recall that in [15], the first term preserves spatial information from content image \(I\) in the stylized image \(Y\). The second term enforces style features from \(S\) with normalized Gram matrices \(G\). In this work, we consider three different objective functions \((L_A, L_C\) and \(L_F)\) depending on the following choice of coefficients:

- \(L_A\) (All scales): \(\omega_A = [0, 0, 0, 1, 0]\) and \(\gamma_A = [1, 1, 1, 1, 1]\)
- \(L_C\) (Coarse scale): \(\omega_C = [0, 0, 1, 0, 0]\) and \(\gamma_C = [0, 0, 1, 1, 1]\)
- \(L_F\) (Fine scale): \(\omega_F = [0, 0, 0, 1, 0]\) and \(\gamma_F = [1, 1, 0, 0, 0]\)

Note that most methods in the literature exclusively rely on \(L_A\) as originally proposed in [15].

2.2. Controlling Style Transfer

As already mentioned, the problem of controlling style features has been already studied, often by means of training deeper networks on dataset of textures. For example, in [17], explicit parameters are exhibited, allowing the users to select the desired style and its intensity. [18] uses an adversarial loss which incorporates random openings gates to encode a whole collection of styles which can be called independently. [19] introduces a conditional generative network which is trained using the perceptual loss \((1)\), capable of synthesizing mixture from several textures.

A different approach consists in changing the metric. As originally shown in [10], the optimal transport distance allows to define and compute the average of several distributions of style features. This framework can be used to either explicitly compute the barycenter of different styles before synthesis [13], or implicitly drive the optimization itself [12].

All those approaches succeed in blending different styles in different fashion, but none of them allow the user to specifically control the scale at which geometric features should be transferred into the content image.

Fig. 2: Comparison of different techniques for mixing styles at different scales, with the approach of [14], [15] and ours (last row).

2.3. Bi-scale Style Transfer

Multi-scale style transfer has been hardly studied in the literature. As far as we know, only Gatys et al. [14] introduced a method for mixing several styles by preserving some style features at different scales. The reason is likely that such aim is not trivial: even if perceptual features are extracted from several layers in \((1)\), therefore at different resolutions, they are not independent. As a consequence, fine details and colors are still encoded in deep layers. For instance, as illustrated in Fig. 2d, optimizing simultaneously the perceptual loss function \(L_C\) and \(L_F\) results in synthesizing an image where style features are in different locations, but not mixed.

To avoid this issue, [14] introduces a 2-step ST approach. It consists first in combining two styles (Style I, Fig. 2b, and Style II, Fig. 2c) by performing ST with the fine scale loss function \(L_F\). Color transfer from the content image (Fig. 2a) is used as a post-processing. Then, this new image (Fig. 2d) is used to perform ST at coarse scale with \(L_C\).

While achieving the desired result, this method needs to be reconduted for each inference. To this end, we propose an alternative optimization strategy that is illustrated in the last row of Fig. 2 with image data optimization. In Fig. 2g, the content image is first stylized at coarser scale with \(L_C\), and then at finer scale with \(L_F\). This strategy allows for training two separate and independent neural networks (Fig. 2h&i) that are presented in the next section.
3. BI-SCALE NETWORKS

Our modular network is built with the cascading of two complementary multi-scale networks fed with inputs at different scales. An overview of the proposed bi-level architecture is shown in Fig. 3.

![Diagram of bi-level architecture](image)

**Fig. 3:** Overview of the proposed bi-level architecture. The Coarse network (top) synthesizes large geometric features. The Fine network (bottom) adds fine details to the input $I$. The two networks are trained independently and combined during evaluation.

The first network (“Coarse” network $C_u$, $\sim 110k$ parameters, in Fig. 3.1) synthesizes large structures from a first style $S_u$. The second network generates thin textures (“Fine network” $F_v$, $\sim 45k$ parameters, Fig. 3.2) from a second style $S_v$. During evaluation, the two networks are combined by the user to produce the desired style transfer. This modular architecture makes it possible to train each module separately, without requiring to train simultaneously for every possible combination of styles.

The two multi-scale networks are inspired from the Texture Network V1 [7] (originally between $\sim 74k$ for texture synthesis and $\sim 110k$ for style transfer) in which the input data $I$ is decomposed and processed at different resolutions. Roughly speaking, the architecture of network $C$ (respectively $F$) is here mirroring the VGG layers required to compute the coarse $L_C$ (resp. fine $L_F$) loss function. As a result, the Fine network with a low receptive field only acts on the first two scales, while the Coarse network, with a much larger receptive-field, conversely processes the following ones.

3.1. Details of the Architecture

On Fig. 3, the Conv modules are composed of three successive $3 \times 3$ convolution layers, each followed by a batchNorm and a Relu activation. Each Up module is composed of a convolution module, followed by a nearest neighbor upsampling layer ($\times 2$), and a batchNorm. A Light Up module is equivalent to two successive Up modules (leading to a $\times 4$ upsampling). Such module has very few parameters so that most of the parameters are encoded before upsamplings, favoring large structures.

3.2. Independent Training of the Networks

During training, each network is trained independently for a given style image $S$ and a dataset of content images $I$ from DIV2K dataset [20]. The input of the network is composed of a batch of 6 content images, decomposed at different resolutions, starting from $356 \times 356$ pixels. These images are concatenated with random gaussian tensors at the same resolutions, as done in [7]. The Fine network $F$ is trained with the fine scale loss function $L_F(I, S, F(I))$ and the Coarse network $C$ is trained with the coarse scale loss function $L_C(I, S, C(I))$. Parameters from both networks converge in a few thousands iterations using the Adam algorithm (learning rate: $5e^{-2}$).

4. EXPERIMENTS

**Style transfer results** Figure 1 demonstrates the ability of the proposed bi-level network to combine styles at different scales. The approach performs bi-scale style transfer on a content image (top-left) with two large-scale style images $a$ and $b$ and two fine-scale style images $I$ and $2$. All combinations ($F_i \circ C_x(I)$ with $x=a$ or $b$ and $i=1$ or $2$) are shown, including results of single-style transfer (i.e $C_x(I)$ and $F_i(I)$). Observe that an additional color transfer is subsequently performed to preserve the content color distribution with respect to the luminance modifications, as discussed later on. Figures 2h&i respectively show the results of the Coarse network alone $C_i(I)$ and of the bi-scale network $F_i \circ C_x(I)$. Features from style images at $I$ are effectively transferred to the content image at two specific scales, preserving features from each image. The comparison to iterative optimization with the same loss function (Figures 2h) demonstrates that the multi-scale architecture of the network $F$ successfully manages to restrict the style modification to very fine scales yet with limitations due to the low number of parameters involved.
**Bi-scale texture synthesis** In Fig. 4, we illustrate the ability of the proposed method to achieve bi-scale texture mixing. Two large-scale style images (a and b) and two fine-scale style images (1 and 2) are used to train the associated networks for texture synthesis. For such a task, only the random tensor is fed into the network, and weight parameters for the content term in the training loss (1) is set to 0 ($\omega = 0$). It leads to single scale texture synthesis (1, 2, a and b). Note that single scale coarse networks favor large scale (large structure of the bricks rather than its grain from b to b). Inversely, fine networks favor thin details (synthesized strokes in 1 are not as long as in the style image 1). For texture mixing and texture synthesis coarse networks are combined with style transfer fine networks, which generate the 4 possible combination results. Again, observe how different features are preserved while being mixed.

![Fig. 4: Illustration of the proposed modular architecture for single texture synthesis and bi-scale texture mixing. Coarse Networks a,b are combined with Fine Networks 1,2.](image)

**Control on color palette** When training, colors are predicted through perceptual loss (1), which may involve false colors for the coarse network trained with deep features which slightly embody color information. Thus, to control the color distribution of the stylized image, we resort to two simple techniques. To begin with, we add a color transfer affine layer to the end of each network to enforce the color mean and covariance of an image. During evaluation, this simplistic module can be used to impose any first and second order color statistics (12 parameters). See for instance examples in Fig. 4 where different color distributions are enforced.

Also, we may combine the stylized luminance with the chrominance of the content image. To avoid any artefacts, we make use of the NLMR filter from [21] which can be accelerated using guided filtering [22]. On the right part of each combination from Fig. 5 this filter is applied to the chrominance channels and guided by the stylized luminance.

![Fig. 5: Illustration of the control allowed by the color transfer step. The content image Input is stylized with the tiger image (a) and optionally one of the textures (1),(2). Each stylized result is split in two parts, where reference colors are transferred from the style (left) or the content image (right). Last row shows the role of the NLMR filter to avoid artifacts from using chrominance from the content image.](image)

**Ablation study** Images a,b,1,2 from Fig. 1 and Fig. 4 are single neural network independent results (i.e single style transfer and single texture synthesis). Observe how the coarse and fine networks may be used independently but also in combinations between each other, allowing the user to combine styles at chosen scales.

5. **CONCLUSION**

We have presented a new bi-scale neural network architecture to perform image style transfer from several examples by combining features at different scales. We favor fine or coarse features synthesis tuning architecture, VGG features depth and by choosing coherent style image. To the best of our knowledge, this is unique as other neural network methods in the literature focus on mixing features across scales without preserving original characteristics. Also, our method is original since it is based on two scale-complementary modular lightweight architectures (~ 155k total parameters) which are combinable in a plug-and-play manner. Our method which allows for fast and robust training would benefit, during inference and combination, from considering deeper and larger architectures to generate more complex features.
6. REFERENCES


