The Visual Computer manuscript No.
(will be inserted by the editor)

Locally Controllable Neural Style Transfer on Mobile Devices *

Max Reimann · Mandy Klingbeil · Sebastian Pasewaldt · Amir Semmo · Matthias Trapp · Jürgen Döllner

Received: date / Accepted: date

Abstract Mobile expressive rendering gained increasing popularity among users seeking casual creativity by image stylization and supports the development of mobile artists as a new user group. In particular, neural style transfer has advanced as a core technology to emulate characteristics of manifold artistic styles. However, when it comes to creative expression, the technology still faces inherent limitations in providing low-level controls for localized image stylization. In this work, we first propose a problem characterization of interactive style transfer representing a trade-off between visual quality, run-time performance, and user control. We then present *MaeSTrO*, a mobile app for orchestration of neural style transfer techniques using iterative, multi-style generative and adaptive neural networks that can be locally controlled by on-screen painting metaphors. At this, we enhance state-of-the-art neural style transfer techniques by mask-based loss-terms that can be interactively parameterized by a generalized user interface to facilitate a creative and localized editing process. We report on two usability studies and one online survey that demonstrate the ability of our app to transfer styles at improved semantic plausibility.

Keywords non-photorealistic rendering · style transfer · neural networks · mobile devices · interactive control · expressive rendering

Max Reimann, Mandy Klingbeil, Sebastian Pasewaldt
Digital Masterpieces GmbH / Hasso Plattner Institute
E-mail: {first.last}@digitalmasterpieces.com

Amir Semmo, Matthias Trapp, Jürgen Döllner
Hasso Plattner Institute, University of Potsdam
E-mail: {first.last}@hpi.uni-potsdam.de

1 Introduction

Mobile expressive rendering has become a core technology amongst users that seek casual creativity by image stylization [8,45] and is continuously supporting the development of mobile artists as a new user group [18]. Image filtering, in particular, takes an essential part of the mobile photo sharing success [3], since filtered photos are more likely to be viewed and commented on by consumers [2]. Image filters are typically implemented by a feature-level engineering approach that provide casual users and mobile artists with high-level and low-level interactive control over the stylization process [15,35], but also limiting it to prescribed stylization effects [19].

A more generalized approach has been introduced by the architecture engineering approach of deep learning, which activates layers of deep convolutional neural networks (CNNs) [39] to match content and style statistics, and thus perform a neural style transfer (NST) between arbitrary images [10]. This way, it is able to emulate characteristics of manifold artistic styles and media without deep prior knowledge of photo processing or editing, which is practically demonstrated by mobile applications such as *Prisma* and *PicsArt*. However, in the mobile domain, the technology provided by software products still faces inherent limitations in providing low-level controls for localized image stylization [36]—in contrast to image filtering—e. g., with respect to image feature semantics for meaningful abstraction [7] and support of visual interest [32].

The goal of this work is to enhance state-of-the-art adaptive NST techniques, thereby providing a generalized user interface with creativity tool support [38] to facilitate interactive editing [15, 33] on mobile devices via lower-level local control (Figure 1). At this, we make the following contributions:

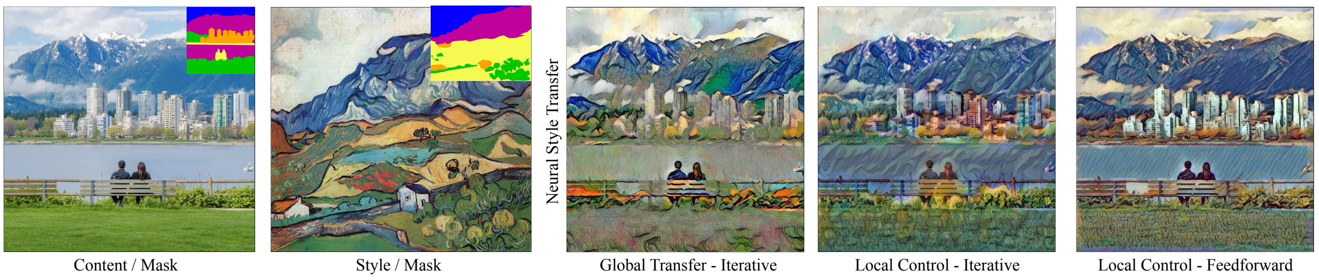

Content / Mask
Style / Mask
Neural Style Transfer
Global Transfer - Iterative
Local Control - Iterative
Local Control - Feedforward

Fig. 1: Comparison of neural style transfer techniques implemented in our app *MaeSTrO*. Interactive location-based filtering is used to stylize the content image, using extensions to global style transfer techniques to provide more expressive results. Content image © karamysh on shutterstock.com, used with permission. The style image by Vincent van Gogh is in the public domain.

1. We provide a problem characterization with user requirements that are mapped to five functional and non-functional requirements for mobile NST.
2. We present *MaeSTrO*, a mobile app for orchestration of three neural style transfer techniques that can be locally controlled by on-screen painting using image masking.
3. We report on two usability studies that show levels of satisfaction reached for the implemented techniques and interactive tools.
4. We propose a mask-based loss-term for semantically-plausible feed-forward style transfer and validate in an online survey its ability to capture artistic styles.

This article is an extended version of our previous paper [29], published at Cyberworlds 2018. As in the original paper, we first provide a background on the problem domain and user requirements that are mapped to functional/non-functional requirements (Section 2). We then give an overview of related work on mobile and controllable style transfer (Section 3). In Section 4, we provide technical details on locally controllable NSTs, with an extended section on direct sub-style matching for semantic learning. We then substantially extend this work by an evaluation section that reports on two qualitative usability studies and one online survey (Section 5). Finally, we conclude this paper and outline opportunities for future work (Section 6).

2 Background

The introduction of mobile expressive rendering had a relevant share in the development of a new user group that discovered the creation of art to be fascinating: *mobile artists*. Their artistic expression is defined through a constant production and discussion of creative imagery, without restriction to a certain skill level or age. As *serious hobbyists* they are always interested in new ways to express their creativity.

We differentiate mobile artists into two groups: Either they start with a blank canvas and their own ideas using low-level drawing or painting techniques, or they use a photograph as input for image processing, e. g., by using the algorithmic support of techniques such as image filtering or example-based rendering via NSTs. We identified the second sub-group as the main target group for *MaeStrO* since they are often early adopters, i. e., they are eager to try new technologies and concepts. To learn more about their requirements, secondary market research, task analysis, user surveys and interviews have been conducted to help design an interactive prototype that fosters creativity, increases productivity and establishes a high user satisfaction when using effects based on NST. Current challenges include the limitation of styles offered in a single app and missing global/local control over an output image, making the usage of such styles unattractive for artists.

User requirements. Table 1 summarizes identified user requirements, i. e., tasks that a system needs to support and the functionalities that need to be provided to complete them. Most of these requirements ground on the need of interactive tools required for art direction [13,31], in particular to widen the interaction spectrum as Isenberg [15] calls for. One of the most important requirements of mobile artists is the *easy exploration of tools and effects*. This includes a self-explanatory interface that enables users to quickly evaluate features, but also a fast image generation process that allows for interactive frame rates (12 frames-per-second or more). Furthermore, *versatile effect presets as well as the possibility to combine different styles* facilitate the development of new ideas. As a limited number of effects also limits creative expression and does not necessarily match a user's stylistic intentions, new ways to explore styles are necessary, e. g., the *possibility to define own styles* through means of experimentation. Beyond that, *artistic control of a style and its parameters* allows for

Table 1: Overview of user requirements mapped to functional and non-functional requirements of NSTs.

User Requirement	SGS	STS	STQ	UC	GMS
Easy exploration of tools and effects		×	×		
Versatile effects or effect presets	×	×	×	×	
Possibility to define own styles	×	×	×	×	
Artistic control of a style/parameters		×		×	
High resolution output					×

creative expression by customizing and therefore personalizing results. To also generate unique results, users want to control the stylistic rendering by modifying the configuration parameters locally and globally, and by adjusting the visibility, e. g., through blending options. Because post-processing the results with other mobile apps is a common practice, a *high resolution output* is essential for keeping up a high-quality photo editing pipeline.

Functional and Non-Functional Requirements. We defined five functional and non-functional requirements of the used NST techniques to meet the expectations and requirements of the target group. The *Style Generation Speed (SGS)* defines the overall time that is needed to configure a neural network based on a style image and a content image in order to generate an output image via style transfer. The *Style Transfer Speed (STS)* defines the overall time to apply a pre-trained style to a content image to generate a final output image. *Style Transfer Quality (STQ)* relates to filtering that considers pictorial semiotics [30, 36], in particular the visual quality with respect to color and features bound to semantics. In the following, we refer to *semantic plausibility* as the fidelity of a NST to stylize image feature classes (e. g., vegetation, buildings, people) in the same spirit as the original artwork. *User Control (UC)* defines the possibility to alter the stylized output globally or locally by modifying how the style image is used for processing. Finally, *GPU-Memory Consumption (GMS)* describes the required memory of the Graphics Processing Unit (GPU). A higher consumption may technically hinder the application of a NST on mobile devices.

3 Related Work

In the following, we give an overview on NST techniques, approaches for their interactive control and their application in the mobile domain. For comprehensive technical overviews we refer to the work by Jing et al. [16] and Semmo et al. [36].

3.1 Neural Style Transfer

Technical Approaches. Neural style transfer was first proposed by Gatys et al. [10], who matched global feature statistics in the layers of a deep CNN using Gram matrices, but which is based on a slow and offline optimization process. Several improvements have been proposed to this method since then, e. g., Li and Wand [20] introduced a patch-based approach also operating on CNN layer outputs to transfer the style more truthfully. To enable interactive performance, Johnson et al. [17] and Ulyanov et al. [43] trained feed-forward neural networks to directly minimize the same loss as the optimization approach. These networks, however, compromise in flexibility as they are pre-trained with respect to a single style. This problem has been further addressed by custom network layers to match multiple styles: Dumoulin et al. [9] use a conditional layer for instance normalization, Li et al. [21] use a network with style selection units, and Zhang and Dana [46] embed a CoMatch layer to match second order statistics. These networks can match tens to several hundred styles. Recently, these approaches have then been further extended to arbitrary styles by determining affine parameters for the instance normalization layers at runtime. In particular, Huang and Belongie [14] compute these parameters directly from the style image in an adaptive instance normalization layer, while Ghiasi et al. [12] use a secondary network to predict them. For *MaeSTrO* we seeked to implement techniques with complementary strengths and weaknesses, thus choosing an optimization-based style transfer [10], a multi-style feed-forward network [46] and an adaptive network [14]. With respect to latter, the approach of Ghiasi et al. [12] may produce results of superior quality but is found to use networks that are too large for a practical usage on mobile devices.

Controllable Neural Style Transfer. Several works proposed methods to improve global and local control over NSTs. Champandard [5] demonstrates the feasibility of a semantic style transfer by using image masks to spatially direct the style application during patch-based transfer [20], an approach that Gatys et al. [11] and Luan et al. [27] similarly used for providing style and content masks for a Gram-matrix based optimization. Gatys et al. [11] also explored controlling the scale of applied styles, where fine details of a style are separated from coarse structures. While all mentioned methods are developed for iterative NSTs, many are also applicable to feed-forward NSTs. For instance, Zhang and Dana [46] demonstrate a brush size control for their networks, Huang and Belongie [14] and Gatys

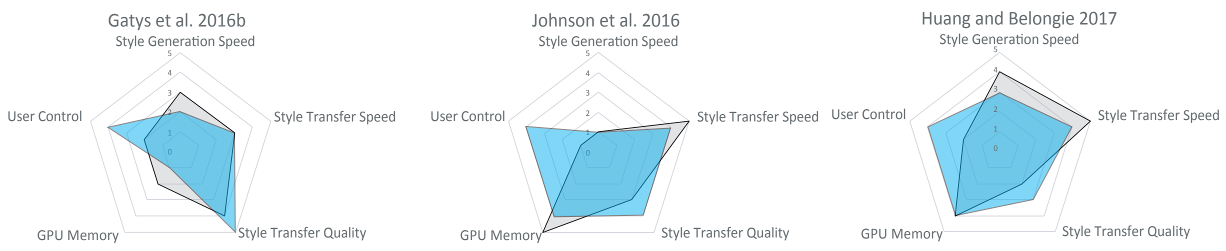


Fig. 2: Subjective comparison of evaluated NST techniques regarding their functional and non-functional requirements (1: worst performance, 5: best performance). Extending existing NST techniques (grey polygon) to facilitate user control over the output leads to higher GPU-memory consumption and a slowed down performance (proposed enhancements: blue polygon).

et al. [11] show localized stylization control, and Liu et al. [25] use depth information while Li et al. [22] use style/content ratio for control of their respective adaptive feed-forward networks. For *MaeSTrO* we follow Luan et al. [27] for mask-based style guides using iterative NSTs and the approach of Huang and Belongie [14] using adaptive NSTs. Further, we train multi-style feed-forward networks with brush size control [14]. At this, we contribute a loss term for mask-guided transfer in multi-style networks and solve transitions between styles by feature-space blending.

3.2 Mobile Applications

Several mobile apps that utilize NSTs have been created and published on app stores (e. g., for iOS, Android). One of the first apps was *Dreamscope*, which uses an iterative NST approach in a client-server environment, i. e., where a server performs the style transfer. A similar functionality has been implemented by `deepart.io` as a web-based application. Both applications allow to define custom styles via style images, but may require multiple minutes/hours for processing.

Feed-forward NSTs were able to tremendously cut down the processing time [17]. In particular, the filtering app *Prisma* was one of the first that successfully used the approach, attracting 30 million users in two months. In its first version, *Prisma* also implemented a client-server-approach, but then opted for an on-device solution for powerful mobile graphics hardware. With the availability of public GPU-based frameworks for executing neural networks (e. g., CoreML), it is even feasible to apply mobile feed-forward NSTs in sub-seconds. Popular examples are *Whisky16* and *Pictory* [37] to interactively apply styles to content images, and *DeepStyleCam* [40] to transfer multiple styles in real-time.

However, none of the mentioned apps enable localized control over the style transfer. As a first global control, image-based post processing has been used to define how the stylized output is blended with the content image (e. g., used in *Painnt*). We follow this idea using mask-based painting as an additional constraint for style transfer. The approach is generalized to enable to choose between a quality-based or performance-oriented NST, which we exemplify for the implemented iterative, feed-forward and adaptive techniques.

4 MaeSTrO

In this section, we first outline a problem characterization with respect to providing interactive tools for parameterizing NSTs, and then propose technical enhancements to state-of-the-art NST techniques that include mask-based loss terms to enable local control.

4.1 Problem Characterization and User Interface

To implement spatial control over the style transfer, the training or configuration of the network can be limited to user-defined regions of the style and content image, as proposed by Luan et al. [27] or Gatys et al. [11]. For *MaeSTrO* we adapted these approaches for all three network types and re-evaluated the most flexible approaches by subjective comparison (Figure 2). Compared to the inflexible approaches, we were able to increase the style transfer quality (Figure 4 and supplemental materials) while causing a higher consumption of GPU memory (GM), which can be crucial due to the limited memory of mobile devices. In addition, the performance initially slowed down significantly (by factor 2-5, depending on the number of sub-styles), hindering an interactive style transfer. Thus, approaches for interactive modifications with a deferred transfer using multi-style feed-forward and adaptive NSTs are proposed, together with a server-sided iterative NST to also synthesize high-quality artistic renditions.

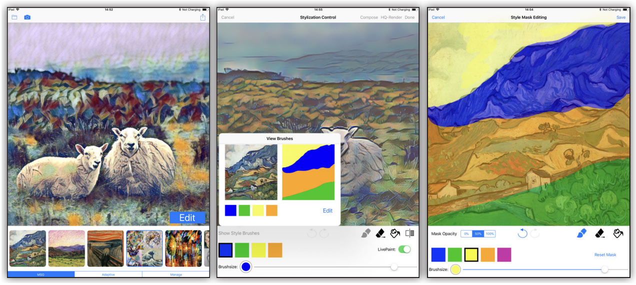


Fig. 3: Overview of important screens in *MaeSTrO*. *Left*: Selection of the NST technique. *Middle*: Locally applying (sub-)styles to a content image with mask preview. *Right*: Editing color-encoded sub-style masks for the adaptive NST. Content image © Rick Barrett on Unsplash.com, used with permission.

Besides the described technical challenges, the increased conceptual complexity has an extensive impact on usability engineering aspects. Giving local control during style application requires new interaction concepts to achieve a high degree of usability within the system. For the first prototype of *MaeSTrO*, we based our design choices on goals like a high learnability of the system and the reduction of error sources, without losing focus on the user requirements of Section 2.

While working on a content image, users gain artistic control by applying multiple styles locally with their finger. We considered three potential approaches to offer them different kinds of styles: (1) Enable users to define and use style brushes, i. e., one style brush is defined through one style image; (2) The usage of detail-controlled style brushes, i. e., applying a style brush with more or less style details through different abstraction levels of the style image; (3) The definition and usage of sub-style brushes. In contrast to (1), one sub-style brush is defined only through a precisely defined part of a style image.

In our first iOS prototype of *MaeSTrO*, we decided to focus on (3), since this approach addresses the most complex challenges that need to be solved to clearly communicate functionalities to end users. These include: (a) how to define sub-style brushes as a user, (b) how to visualize pre-defined and user-defined sub-style brushes of a style image and (c) how to apply different sub-style brushes to a content image. Relevant screens of our prototype are shown in Figure 3.

4.2 Iterative Neural Style Transfer

We adapt the iterative NST of Gatys et al. [10], which defines style transfer as the optimization problem of finding a stylized target image t, whose content is similar to a content image c and whose style is similar to a style image s. At this, a content loss $\mathcal{L}_c$ and a style loss $\mathcal{L}_s$ are minimized using the Gram matrix over activations in a deep neural network. We refer to [10] for formal definitions of the loss terms and optimization. The used Gram matrix sums over the height and width, thus the location of individual features in the style image is lost in the result, and a global texture is transferred.

4.2.1 Spatial control and brush size control

Location-based control over NSTs can be achieved by segmenting the style and content image into different local control masks, as described in [27]. The content image is commonly segmented along semantic borders, while masks in the style image are typically chosen to seperate different textures, colors or shapes. To this end, the style loss term is adjusted to include masks:

$$\mathcal{L}^l_{s+}(t) = \sum_{m=1}^{C} \| \frac{1}{A^{l,m}(t)} \mathcal{G}^{l,m}(t) - \frac{1}{A^{l,m}(s)} \mathcal{G}^{l,m}(s) \|^2_F \quad (1)$$

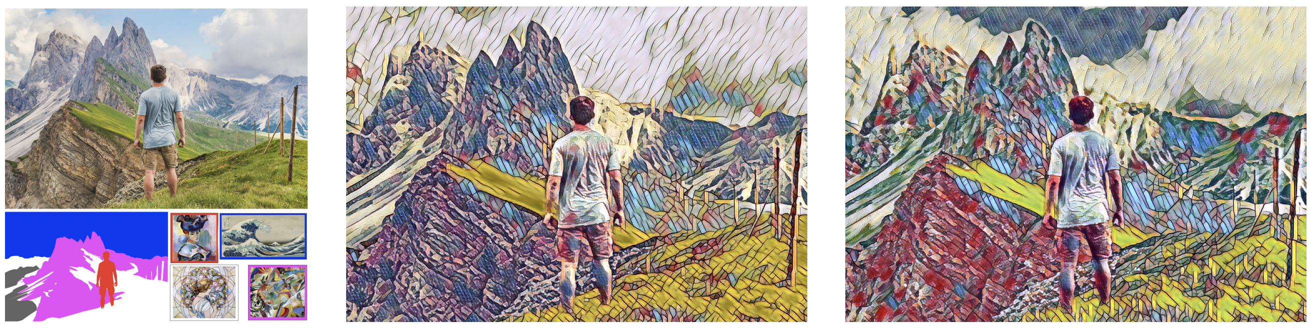
$$\mathcal{G}^{l,m}(t) = R\big(\phi^l(t) M^l_m(t)\big) R\big(\phi^l(t) M^l_m(t)\big)^T \quad (2)$$

where C is the number of style and content masks, $\mathcal{G}^{l,m}(\mathbf{t})$ is the Gram matrix of the target for layer l and mask number m, R is a feature vectorization operation and $M^l_m(t)$ is the content mask m downsampled to the feature map spatial size at layer l, whereas $\mathcal{G}^{l,m}(\mathbf{s})$ conversely yields the Gram matrix for the respective style mask. The area of the masks $A^{l,m}$ is used to normalize magnitude differences in the Gram matrices and reduce intensity artifacts.

In *MaeSTrO*, content masks are either drawn by hand or generated by fully convolutional networks for semantic segmentation (FCN) [26]. The generated masks are grouped by their labels into several pre-defined groups based on textural and semantic similarities. The same FCN is then used as a feature extractor $\phi(t)$ to improve matching between mask and image and to reduce overall memory consumption. The style masks are also drawn by hand and brush size control is achieved by varying the size of the style image with the NST [10].

4.3 Feed-forward Neural Style Transfer

The iterative, spatially controllable approach has a long computation time even for low-resolution images. To speed up the transfer procedure, a feed-forward convolutional neural network, termed the style transfer



(a) Input and style mapping (b) Without $\mathcal{L}_{mask}(t)$ (c) With $\mathcal{L}_{mask}(t)$

Fig. 4: Using the mask loss term $\mathcal{L}_{mask}(t)$ to regularize the decoder for spatially varying styles: (b) *without* masked loss, where dominant style elements (e. g., black mosaic lines) are transferred to differently labeled regions, (c) *with* masked loss, which strictly adheres to the painted style labeling. Content image © Joshua Earle on Unsplash.com, used with permission.

network, can be trained to learn a direct transformation from content to pastiche [17]. To provide smoothly blended sub-styles, we propose the use of networks capable of capturing multiple styles—instead of training multiple networks—, such as the multi-style generative network (*MSG-Net*) proposed by Zhang and Dana [46], who reduce training time and storage space for multiple styles and enable style blending in feature space.

4.3.1 Multi-Style NSTs

The *MSG-Net* [46] learns a generator network $G(c, s)$ that takes both the content and the style targets as inputs, and computes the style Gram matrix at run-time. The main difference to previous work is the introduction of a "CoMatch" layer, which matches second order feature statistics based on the given styles and is able to adapt the output to a set of trainable styles. This layer consists of a learnable weight matrix W, which tunes the content feature map $\phi^l(c)$ based on the target style:

$$\hat{\phi}^l(c) = R^{-1}\left[R(\phi^l(c))^T W \mathcal{G}(\phi^l(s))\right]^T \tag{3}$$

The weight matrix W learns to strike a balance between the style and content loss for a particular style.

4.3.2 Feature-space mask merging

Similar to [6], we combine different styles in the feature-space, rather than the image-space to prevent artifacts at mask contours and speed up model inference. The *MSG* network conceptually consists of an *encoder* part, which is a siamese network used for feature extraction of both content and style image, a *transformer* part, which carries out the style transformation, and a *decoder*, which is an upsampling convolutional layer to the output image. Training the network in prior works is done without requirements for spatial control, the network is thus optimized towards producing global styles. Using input features with spatially varying styles in the decoder can lead to local image artifacts and certain style elements appearing in differently labeled areas (Figure 4b). We add an additional reguralization term to $\mathcal{L}_{total}(t)$ to diminish this effect (Figure 4c), by adding the mean squared error between outputs masked in feature space and outputs masked in image space:

$$\mathcal{L}_{mask}(t) = \left\|D\left[\sum_{s\in S} T(t,s)M_s\right] - \sum_{s\in S} G(t,s)M_s\right\|_F^2, \tag{4}$$

where T is the generator network up to the decoder D, $G(t, s) = D(T(t, s))$ is the complete generator network, and M_s the mask for style s. The loss $\mathcal{L}_{mask}(t)$ is added in the last epoch of training, as it is a fine-tuning operation on the decoder network and is computationally more expensive than the other loss terms. To prevent spatial mask-mapping from affecting the encoding and transformation part of the network, the rest of the network is kept fixed during training.

4.3.3 Network learning

We follow the procedure and parameters described by Zhang and Dana [46] and train networks using either 64 or 128 generator channels on *MS-Coco* [24] (8 epochs, batch size 4). The style is switched every three iterations and the style size is cycled through the sizes $\{256,512,768\}$ for brush size learning. In the final epoch, the masked loss $\mathcal{L}_{mask}(t)$ is added to the total loss term $\mathcal{L}_{total}$ and trained using batch size 1 on *COCO-Stuff* [4], a dataset containing dense segmentation masks, categorizing different "stuff" classes, for classification along material boundaries instead of objects. We group these

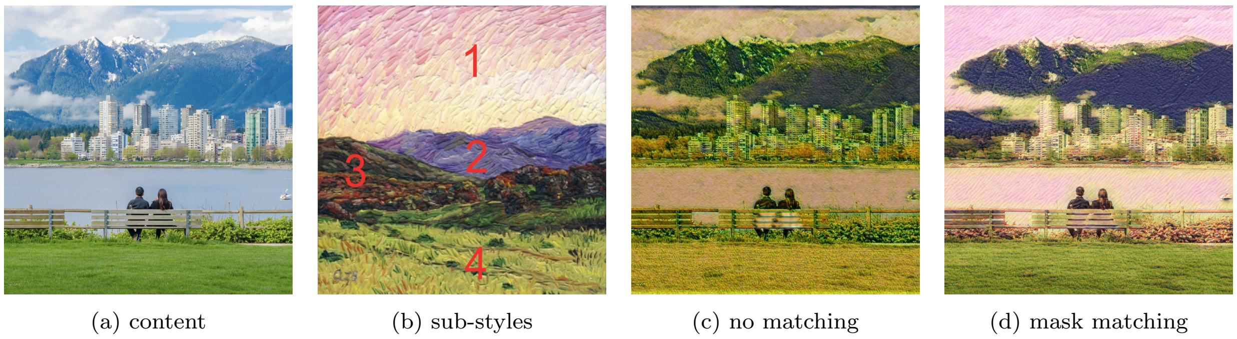


Fig. 5: Networks trained with $\mathcal{L}_{mask}(t)$ to learn a semantic mapping between sub-styles and semantic regions. Content image © karamysh on shutterstock.com, used with permission.

masks to the predefined labels and match them according to strategies described in Section 4.3.4.

4.3.4 Learning semantic mappings

Different matching strategies between style and content masks with $\mathcal{L}_{mask}(t)$ during training can have a significant impact on the learned style representations. Style masks are either matched to content masks with a certain label, or to random masks of any class. In the first case, sub-styles are given explicitly mapped to a semantic by manually assigning classes to sub-styles and only matching them with the corresponding masks in the content images, e. g., a sub-style labeled "sky" will only be learned on regions labeled as sky in the training dataset. By contrast, randomly matching content masks with styles makes no assumption about the semantic meaning of a style, and is desirable for general-purpose stylization tooling, where regions can be depicted with arbitrary styles.

Semantically trained networks can have superior results for styles with a clear semantic mapping and can be used to enhance existing global NST methods by transferring segments of a style to a content image in a meaningful way, as shown in Figure 5. Although the same training losses are used for learning semantic mappings as for training with mask regularization for artifact reduction (Section 4.3.2), there are several differences in the training routine. When training with semantic mappings, $\mathcal{L}_{mask}(t)$ regularization is disabled only in the first epoch, and then used in the following epochs to learn the classification of different semantic concepts. In contrast to random mask mapping, the weights of encoder and stylization layers are not fixed during weight updates, to learn semantic classification in lower layers. The CoMatch-layer style Gram matrix $\mathcal{G}(\phi^l(s))$ is fixed to the global style Gram matrix after learning the sub-styles in the first epoch, as user control is replaced by an automatic classification. Compared to other automatic semantic NSTs, e. g. [23, 47], the transfer is performed in a single feed-forward pass and therefore does not include any runtime overhead.

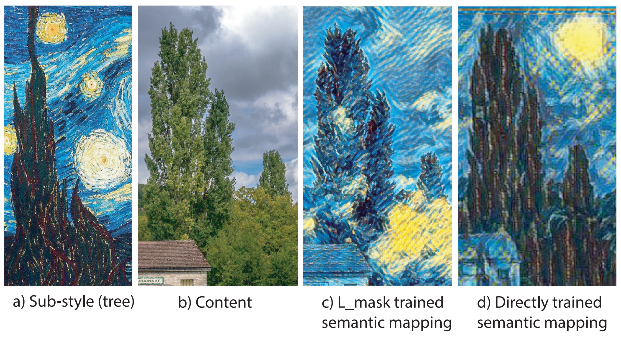

a) Sub-style (tree)
b) Content
c) L_mask trained semantic mapping
d) Directly trained semantic mapping

Fig. 6: Direct, class-based losses can improve structural matching over globally pretrained sub-styles that are matched using $\mathcal{L}_{mask}(t)$. Input and stylization results are *crops* from larger images.

4.3.5 Direct sub-style matching for semantic learning

One drawback of using the mask regularization for semantic training is that sub-styles are first trained globally, where the mask regularization spatially constrains the learned sub-style to certain semantic concepts thereafter. However, the applied stylization is still derived from the global representation. To this end, the learned representation of the sub style globally matches the sub-style's Gram matrix, however it does not necessarily create images in which the stylization of individual objects—corresponding to the learned semantics—matches the Gram matrices of their sub-style counterparts. Put differently, a local sub-style Gram matrix is learned on a global level and then applied to local objects, thus a direct structural matching between individual content masks and sub-styles does not take place during training.

This does not pose a problem for sub styles that are applied to rather large and diverse regions (e. g.sky, ground, mountain; see Figure 5) and thus do not need to establish a object-level structure matching. However, for styles that require an exact structural correspondence, such as matching the cypress tree in van Gogh's *The Starry Night* to trees in the content image, this training approach can lead to less convincing transfer results. In this case, the training losses can be augmented by using the loss formulation for spatial control of iterative style transfer (Equation 1), thereby establishing a direct mapping between sub styles and the semantic concept (Figure 6). This approach is suitable for styles with an unambiguous mapping of style image features to content, such as when transferring a portrait artwork to a portrait image with similar content. In Section 5, we report on a survey which was conducted to compare images generated using learned semantic mappings and images generated using a global stylization.

4.4 Adaptive Neural Style Transfer

Although *MSG-Net* can learn a large amount of styles, the computational burden for training new styles is still high and therefore does not permit end users to add custom styles. A recent work by Huang and Belongie [14] introduces a method to perform a NST on arbitrary, new styles by using an encoder-decoder network containing an adaptive instance normalization layer in the middle. The authors use the insight that instance normalization performs a normalization of feature statistics to a certain style using a small set of affine parameters (scale and shift) [44]. They propose an adaptive instance normalization layer, that directly computes these parameters from the first order style feature statistics, i. e., the mean μ and standard deviation σ:

$$\mathit{AdaIN}(c) = \sigma[\phi(s)]\Big(\frac{\phi(c) - \mu[\phi(c)]}{\sigma[\phi(c)]}\Big) + \mu[\phi(s)] \tag{5}$$

where $\phi(s)$ are the style features and $\phi(c)$ the content features. After normalizing the feature maps to the selected style using *AdaIn*, the decoder transforms the features into the stylized output image. The decoder is trained on a large number of content/style image pairs and learns to reproduce images with arbitrary styles from the style-normalized features.

4.4.1 Spatial control and brush size control

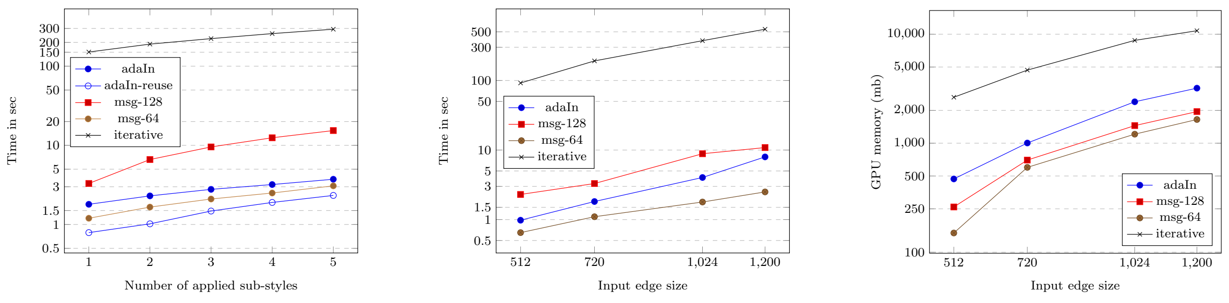
To adapt this method to spatial style control, the *AdaIn* normalization of the input to different styles is performed and then jointly decoded [14]. The decoder, although learned on homogeneous inputs, generalizes to inputs of regions with different styles. Both the features of the style and the *AdaIn* feature statistics are computed for each local-control mask at run-time. Normalization to the masked area, similar to the iterative approach, is not required because mean and variance are locally independent. Brush size control can be achieved by varying the size of the style image input to the encoder, similar to the *MSG-Net* method, but without needing extra training.

4.5 Implementation

Our implementation consists of three components: (1) A *PyTorch* implementation of the *MSG* model training and style control operations, (2) an implementation targeted to mobile Apple devices based on *iOS* and *CoreML* for on-device image processing and (3) a server-based *PyTorch* implementation of the iterative approach. We extended *CoreML* layers for the *CoMatch* layer of the *MSG* network and the *AdaIn* layer of the adaptive network, for which we provide a CPU-based and a GPU-based implementation using *Metal* shaders. The *MSG CoMatch* layer in the formulation of Equation 3 requires the style features and their Gram matrices to be dynamically computed, but which are compute-intensive operations. We thus optimized the performance of the on-device stylization by pre-computing $\hat{W} = \big[W\mathcal{G}(\phi^l(s))\big]^T$ off-device and then computing the *CoMatch* layer output on-device as $\hat{\phi}^l(c) = R^{-1}\big[\hat{W}R(\phi(c))\big]$. The *reshape* operation R only reinterprets the memory layout, thus only one on-device matrix multiplication is required for this layer. The performance of the adaptive style transfer network is optimized by caching the computed content and style features (labeled *AdaIn-reuse* in Figure 7), thus only requiring to re-compute feature means and variances of selected image regions on mask changes. Applying n brushes to one image requires n fast adaptive *InstanceNorm* operations to tune the features to different styles, and one compute-intensive decoder pass.

5 Evaluation

In this section, first outline case studies of the implemented techniques and compare their performance with respect to the requirements discussed in Section 2. Because *MaeSTrO* targets end users, we also report on two usability studies with the aim to improve user interaction, and one online survey that compares global stylizations and learned semantic mappings.



(a) #sub-styles vs. run-time (b) image size vs. run-time (c) image size vs. memory consum.

Fig. 7: Performance of the implemented techniques (feed-forward NST / iPad Pro 10.5", iterative NST / NVidia® GTX-1080TI). The time for the adaptive method was measured with and without re-using content/style features.

5.1 Case Studies

The comparisons in Figure 8 indicate that the NST techniques yield visually different outputs. In particular, the approach of Gatys et al. [10] and the adaptive approach of Huang et al. [14] largely fail to transfer color features. By contrast, our enhancements enable local control over the NST, which facilitates the implementation of a semantic mapping of a style image to a content image, i. e., currently by explicit manual labeling of image regions—as an enhancement this could also be performed using GIST descriptors [41,42]. This leads to a more plausible transfer of texture and color features for the iterative and feed-forward approach. Although the adaptive approach generally lacks to produce results with a striking texture transfer, a clear separation between the different sub-styles could be achieved.

Our approaches have in common that sub-styles are mapped to masked areas of the content image. This enables different application scenarios ranging from a semantics-based, over a separated fore- and background to a completely free stylization. Although the mask creation process can be automated, we find that the manual approach yields better results (Figure 8). This is especially true for regions with sparse color or texture features, which may lead to miss-segmentations. Nevertheless, an automatic segmentation can be useful as a starting point for casual users, since it eases the refinement of the masks.

5.2 Performance

The performance results in Figure 7 show that increasing the number of applied styles has an almost equal effect on all implemented NST techniques. While *AdaIn* is sligthly slower than *MSG-64* when computed on new content and style images, the extracted features and style statistics are reused in both methods in *MaeSTrO*. However, the *AdaIn* method has the encoder as most compute-intensive component, which is applied only once. By contrast, the *MSG* network spends most of its run-time on the style transformation, which is applied per sub-style. Subsequent *AdaIn* stylizations of the same content/style combination (*adaIn-reuse* in Figure 7) are thus considerably faster than the corresponding *MSG* stylization. The *MSG-128* network has a significant performance degradation over the other feed-forward methods, however it also offers, by empirical analysis, the best image quality and can store vastly more styles than *MSG-64*.

Increasing the image size is strongly limited by the GPU memory, where sizes of 1200^2 pixels push the limits of available memory on modern mobile devices (typically around 2-4GB) and, for the iterative method, available memory on high-end desktop GPUs (typically 8-16GB). Producing very high-resolution images is therefore not possible with the given tools and would require either new, smaller network architectures, or image tiling procedures.

5.3 User Control

MaeSTrO gives users the possibility to define sub-styles based on a style image. Sub-styles define specific elements within a style image, e.g. sky, buildings, or hair. They are important to transfer the style to a content image in a semantically correct manner. We argue that users want to transfer a style semantically correct, i. e., they are interested to stylize a specific element of their content image similar to a comparable element in the style image. Therefore, users would also want to define sub-styles with regards to content elements, e. g., define one sub-style for the sky and one for buildings, instead of a fixed, given definition. To this end, we conducted two usability studies to evaluate and improve our app, and gain qualitative feedback from users.

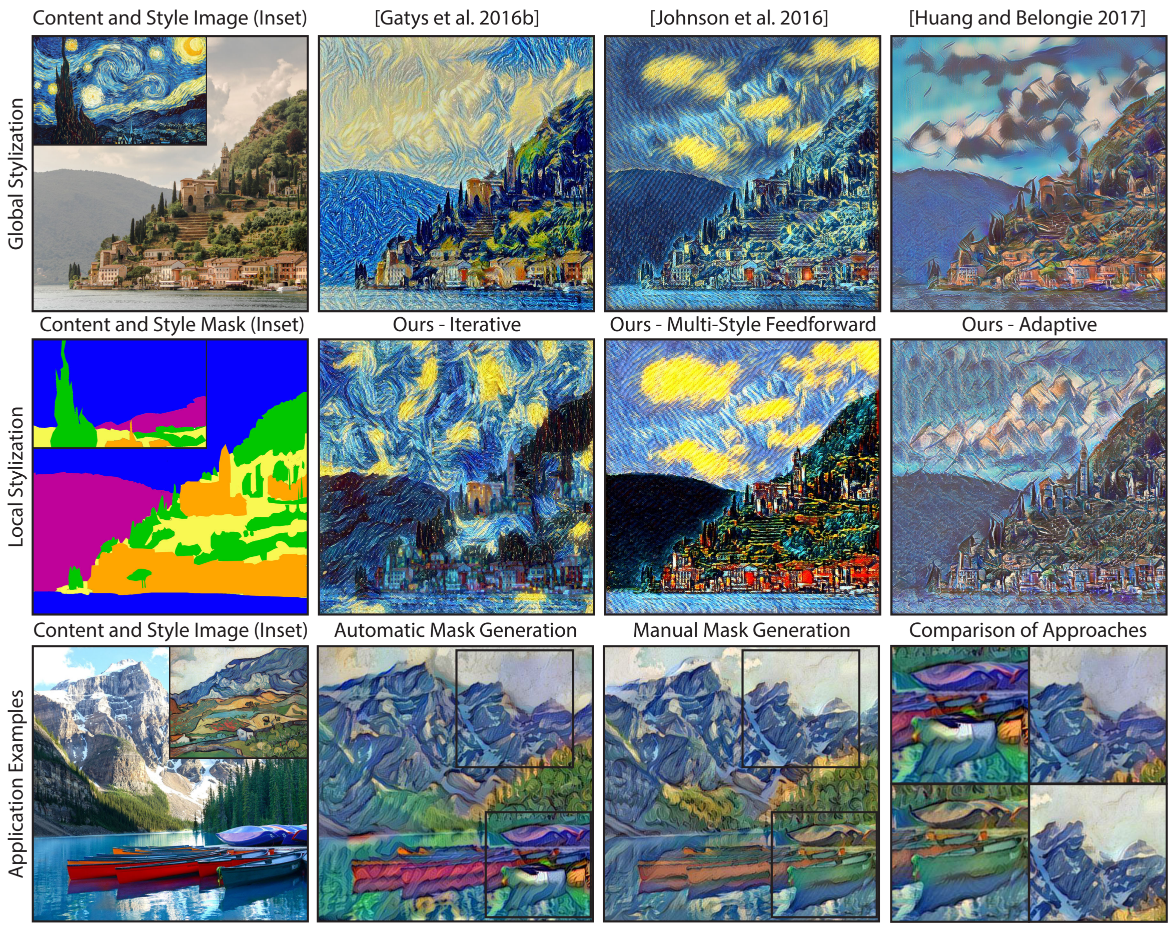


Fig. 8: Outputs created with *MaeSTrO*. Local control enables semantically more plausible results and clearer separations between image regions. Content images © Matthew Fournier and Henrique Ferreira on Unsplash.com, used with permission.

5.3.1 Initial User Study

Our first study aimed to get early insights about usability flaws and was performed on a slightly modified version of our internal research prototype (Figure 3). Users could work with a given content image and apply styles of different artworks, e. g., "The Starry Night" by Vincent van Gogh. When selecting a piece of art, its style is applied globally to the content image, while allowing to edit the stylized content image locally with sub-style brushes afterwards. In this version, we used color masks to define sub-styles of a style image, and to map a defined sub-style locally to a content image (as shown in Figure 3).

When defining sub-styles, users would open a style image and be able to select between five colors. As an example, using blue, users would define the sub-style for the sky by drawing with their finger, applying a blue color to that region to compose a style mask. At a later point, when applying sub-styles to a content image, users would select the color blue and draw on the sky of their content image to map the style, creating the content mask.

Hypothesis. We hypothesized that there is no difference in the time required to understand the concept of color masking whether the user first defined a mask for sub-styles of a style image or applied a predefined mask to a given content image.

Participants. We exposed three participants to the definition of sub-styles first (group 1), and three other participants to the sub-style application first (group 2). All six participants were using the research prototype on an Apple iPad Pro 12.9" (1st generation). The participants (5 male, 1 female) were mainly casual users of photo apps, between 16 and 37 years old, had mixed experiences with photo manipulation apps and no experience with *MaeSTrO*.

Experimental design. We explained the main purpose of the app and asked the participants to perform different tasks to reach a specific goal. Group 1 started with the sub-style definition process right away, where a style image was already selected. They were asked to define at least two sub-styles (using adaptive NSTs). Group 2 started with a given content image that was not stylized yet. The style selection for multi-style feed-forward NSTs had already been opened for them. They received tasks to "stylize the image with van Gogh's Starry Night", to "edit the stylized image", and, looking at a picture that was not transformed semantically correct, to "replace the yellow lawn with something more adequate" using pre-defined sub-styles. All participants filled out a short survey afterwards.

Results. We were not able to accept or reject the hypothesis, since the user interface and the usability exhibited several major flaws. Although, all six participants were convinced that the local control improved the overall style quality, we could observe several challenges when learning the functionalities of the app and utilizing them for their purposes. In the following, we are outlining the major challenges. First, the global application of a style onto a content image caused misleading expectations and hindered the creation of content masks. Two users expected that the app would automatically create content masks based on the semantics used within the style masks. Other speculations included that the app would give recommendations for the application of pre-defined sub-styles to sub-regions of the content image. The majority of testers (five out of six) also missed a direct comparison between the globally stylized image and the image with their local changes. Four would have preferred to work directly on the original content image. Second, the mapping of sub-styles through colors has been misleading when working with images, as it would directly lead to the idea of "colorizing the picture". It was the main reason why none of the test users, neither from group 1 or 2, were able to complete all of their given tasks within a reasonable amount of time i. e.within 10 minutes. As a potential help for group 2, we integrated a small view with the style image and the style mask into the edit screen. The issue of mapping style and content masks overlaps with a third challenge: the fast exploration of local changes in a content image. When editing a content image, users were able to view the content mask, showing the applied colors, or a low-resolution preview with insights about the expected results. Four out of six users preferred the preview, but switched up to 8 times per minute between the views to produce satisfying results. They used the content mask to make sure they did not miss to mask any parts of an edited region, and switched back to the preview to make relevant changes on their artwork.

5.3.2 Follow-up User Study

The feedback we gained from the first user study was used to redesign the app (as shown in Figure 9). We identified the mapping of style and content masks through colors as the main issue. Therefore we exchanged the mapping with something more self-explanatory, while keeping the colors to create a mask. For other observed issues, the participants of the second usability study should also represent a control group for our first tests.

For the definition of sub-styles, we added a table as an overview that opens first (Figure 9). Each row represents one sub-style, showing the style image and the sub-style as highlighted part of this image. Users would edit a sub-style by tapping on it, or create a new one by tapping the plus. We kept the color to indicate the mask area that has been drawn with a finger. Since the color is not used for later mapping purposes and to reduce the user's workload, the system chooses the color by itself.

For the sub-style application, we also separated the sub-style selection from the editing view, forcing users to actively open the sub-style overview to select another one. If users want to edit their stylized content image, the first available sub-style is already selected. Its name and preview—globally applied to the content image—are shown in the toolbox at the bottom of the screen. A tap onto it is necessary to navigate to the overall selection.

Hypothesis. We hypothesized that (1) the user interface changes increase the learnability and, thus, (2) decrease the task completion time.

Participants. We acquired six different participants (3 male, 3 female), grouped again by the experimental conditions. Group 1 started with the sub-style definition, group 2 with the sub-style application. The participants were between 17 and 33 years old, casual users of photo apps and did not have any experience with *MaeSTrO* before.

Results. As it turned out, the new design significantly increased the learnability of the system in general. Thus, the hypotheses 1 could be accepted. In average, group 1 was able to understand the concept of sub-styles and how to create them in under 4 minutes, and group 2 was able to understand how to apply sub-styles semantically correct in under 10 minutes. Thus, hypothesis 2

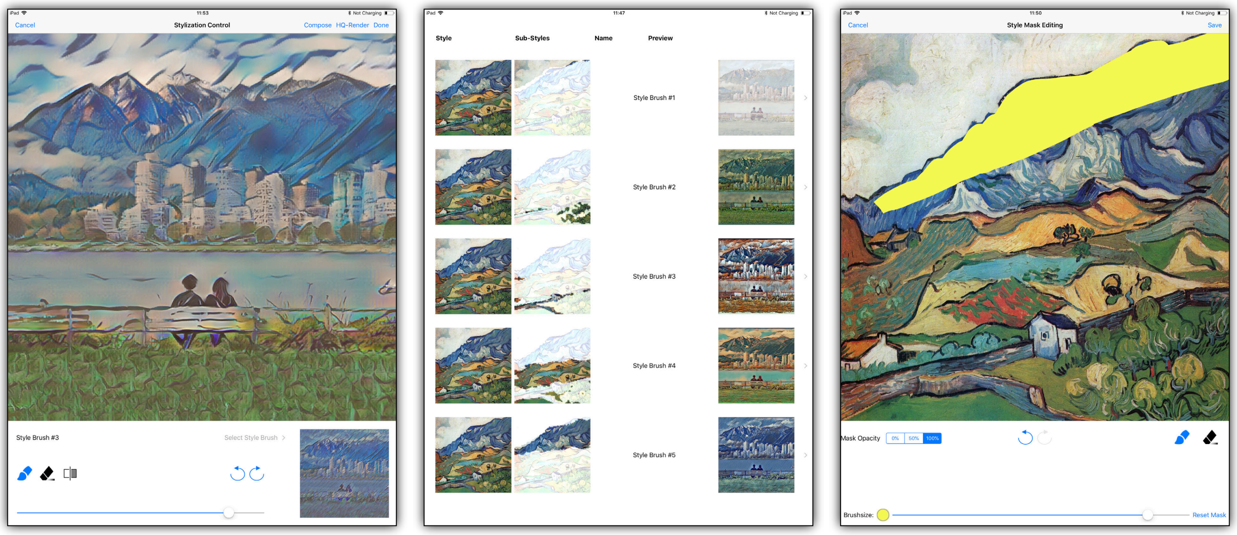


Fig. 9: Overview of important screens in *MaeSTrO* after a redesign. *Left*: Preview of a selected sub-style in the toolbox at the bottom. *Middle*: (Sub-)style selection when editing a stylized content image. *Right*: Definition of one sub-style mask per view for adaptive NST.

could also be accepted. Still, we observed a few challenges when completing the individual tasks that are outlined in the following. First, the style that would be globally applied to a content image was not listed within the (sub-)style selection. Two users thought that a random sub-style was used to globally stylize the image, and therefore that they could select this style to reverse their changes. Further, the global application still hindered the creativity of the users, four out of six would have liked it better to start working on a non-stylized image. Second, editing a stylized content image, users needed some time to figure out the meaning of the thumbnail in their toolbox. It shows the selected sub-style globally applied to the content image as a preview. For further iterations, one participant suggested to let users select a sub-style first before entering the editing view. Third, creating a sub-style, two out of six participants hesitated to save their first mask. Our UI changes forced users to select a (sub-)style first or tap "New", and draw one mask for one sub-style then. To create another sub-style, users needed to tap "Save". It wasn't completely clear to users that they would just save their mask and go back to the (sub-)style overview.

5.4 Semantic NST Survey

Training networks with semantic-aware mappings aims at more closely matching an artwork's style, given an input image with similar content elements. Judging the quality of a NST method remains an open question in the research community, as qualitative assessment is highly subjective and no established quantitative measures for NST methods exist.

Therefore, we conducted an online survey to evaluate if images generated by semantically trained networks are perceived to be closer to the style than the global (semantics-unaware) method of Johnson et al. [17]. Participants complete several *tasks*, consisting of the input image, the artwork and side-by-side a pair of stylizations (A and B), generated by the globally- and the semantically trained network (see Figure 11 and Figure 12). The participants then answer three preference questions (choosing stylization A or B) without knowing which image corresponds to which method. In each task, the participants are asked to choose "which stylized image better reflects the artworks style" (Q1), "which of the two stylized images is more aesthetic" (Q2) and "which stylized image depicts specific content elements more similar to those in the artwork" (Q3). A total of 436 participants[1] (211 female, 208 male, 17 other; aged 17-59) were surveyed, all of them completed the same 10 tasks.

5.4.1 Style representation of semantic networks

The responses to Q1 (Figure 10) show that images generated by semantically trained networks represent the artwork's style significantly better than their global counterparts on all tested images. Only in task 9 the semantic rendition is not strongly preferred over the globally stylized image, which is likely to be caused by the

[1] mainly recruited through volunteers on r/samplesize

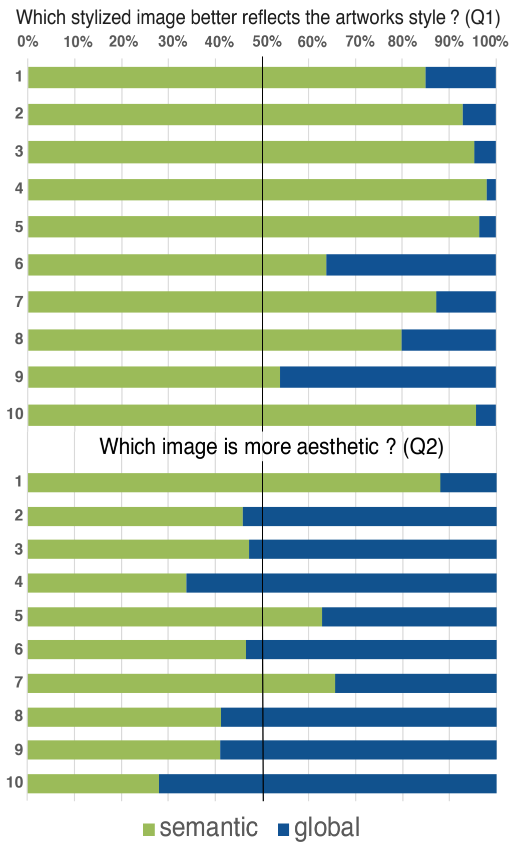


Fig. 10: Responses to Q1 and Q2, per task. The corresponding images are shown in Figure 5 (task 1), Figure 11 (task 2-7) and Figure 12 (task 8-10).

depiction of foreground objects with a low contrast, making different objects hard to distinguish (see Figure 12). Another interesting observation is that significantly more participants think the semantic rendition in task 7 (see Figure 11) is closer to the style than they do in task 6, although both input head portraits are stylized with the same network. A possible explanation is that the portrait in task 7 is very close in terms of content to the artwork, therefore making it easier to establish a one-to-one mapping between style and content (especially for the semantic NST) than for task 6, which contains elements not found in the style image (e.g., beard and short hair).

Q3 acts as a control question for Q1, as overall style perception includes not only semantically plausible matching between style and content but also other factors such as general texturing, brush-strokes, etc. However, we hypothesize that given a direct comparison between stylized image and artwork with similar content, semantically plausible matching will be the dominating factor in style perception. Analysis of responses reveals that Q1 and Q3 are highly dependent: for every task, more than 90% of responses to Q1 and Q3 match and the likelihoods of independence (using χ^2 tests) are $p < 1 \times 10^{-12}$ per task. Therefore, for content-wise similar content/style pairs, the perception of how well an image reflects a style is mostly influenced by the semantic plausibility of the transfer.

5.4.2 Human and computational aesthetics.

For the aesthetic preference question (Q2) there is no significant preference for any method when averaging all task responses. On a per-task level, we define that a >5% difference between both choices marks an unambiguous preference for one method, as at that level the margin of error does not allow a different majority preference using a 95% confidence interval. There is an unambiguous preference for the global method in 4/10 of tasks and for the semantic method in 3/10 of tasks, while for the remaining three tasks, there is only a slight preference for the global method. Portrait style transfers are particularly challenging for semantics based NSTs, as they require pixel-precise segmentation of several facial features, which may not have exactly defined boundaries. Previously, these have been tackled by using hand-drawn masks [34].

Examining these images, there is a preference for stylizations with higher global and local contrasts (acutance) versus their low-contrast counterparts on all tasks, comparing portrait stylizations (Figure 11), the transfer of fine-grained structures also positively seems to influence the aesthetics preference compared to their coarser counterparts. However, this kind of informal assessment is subjective and needs to be formalized by measuring correlations between the aesthetic preference responses and image metrics for aesthetics scoring.

Several studies have been conducted into computational aesthetics judgment of images, however, many of the methods are either black-box machine-learning approaches or rely on common photographic rules (e.g. for image composition), which are mostly not applicable in this case, as the content of the compared stylizations is the same. We choose two image metrics for aesthetics scoring, sharpness ψ_{sh} and tone ψ_{to}, as described by Aydin et al. [1]. They deem image sharpness as the most important metric for image aesthetics, and tone, which is the difference of maximum and mininum gamma-corrected luminance values, is used to describe local and global contrast differences. Other metrics in [1] are either related to image composition, camera focus or colorfulness, and thus not directly relevant in measuring quality of stylizations. The aesthetics score for a stylization is computed as $W = \psi_{sh}\psi_{to}, W \in [0, 1]$. The difference between W(global) and W(semantic) shows the

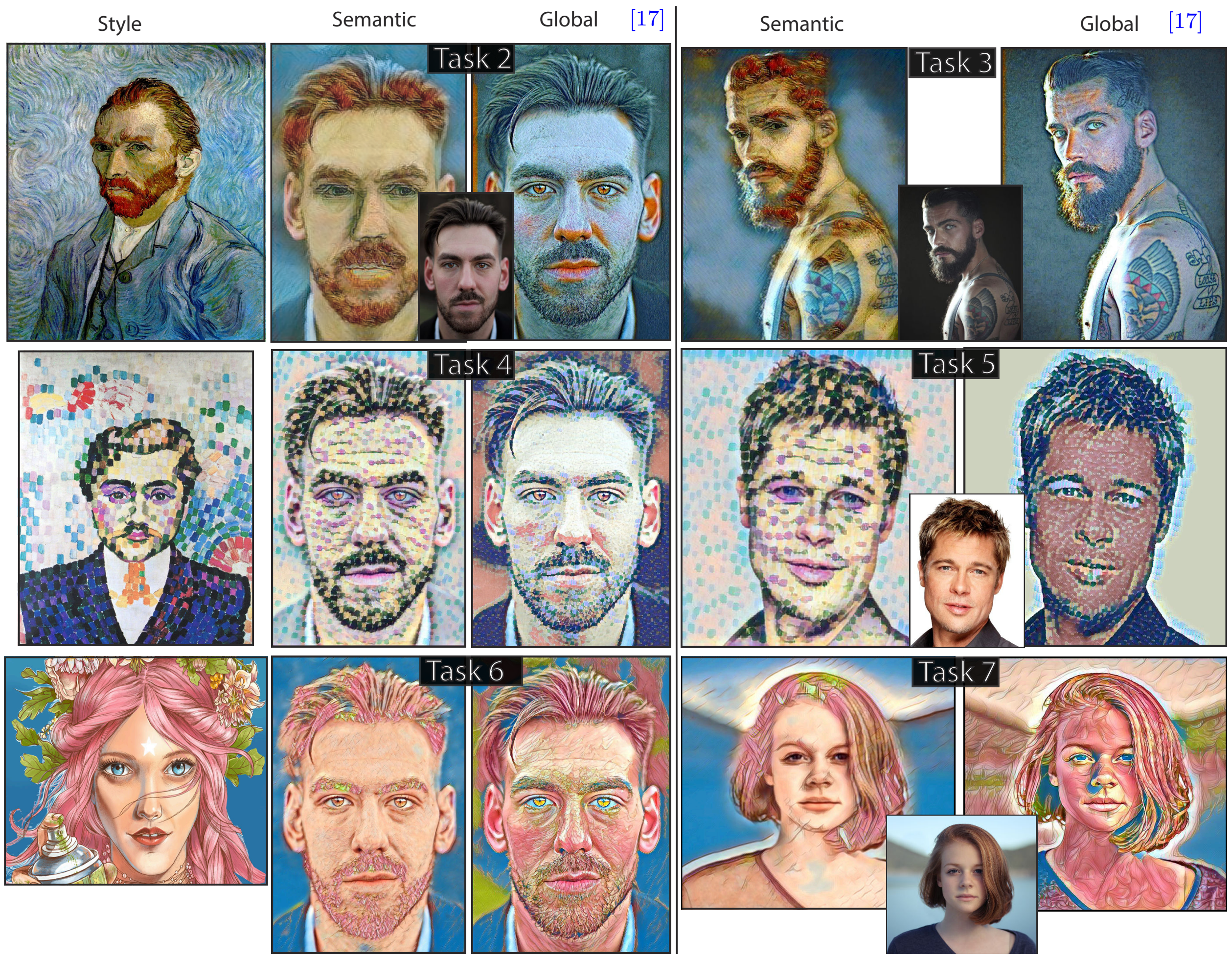


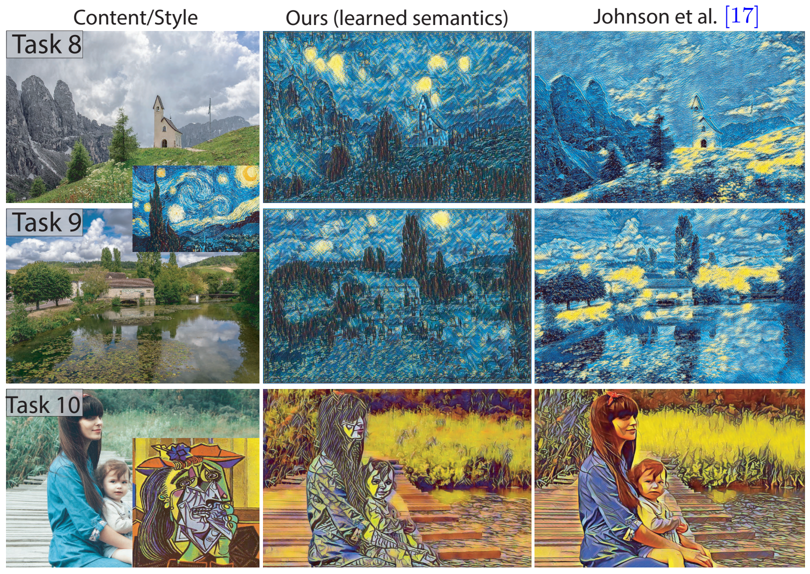
Fig. 11: Survey tasks with global and semantic portrait stylization. Semantic NSTs are trained by segmenting the artwork into three sub-styles (hair, face, background) and matching to the corresponding content masks of the *mut1ny* head segmentation dataset containing 18k faces with pixel-level segmentation masks[2]. Content images © Adam Marcucci, Christopher Campbell on Unsplash.com, used with permission, or public domain.

aesthetics score "preference" for one of the two methods and is compared to the human aesthetic preferences in Figure 12. The comparison shows, that the aesthetics score and human preference correspond in 80% cases (in sign, not in magnitude), and show that a higher sharpness and tone metric correlates with higher aesthetic preference for the stylization. The outlier in task 7 could be explained by the obstructive lines in the woman's face in the semantic stylization, which may negatively affect its perceived aesthetics, as blemishless faces are commonly found to be more attractive and aesthetic.

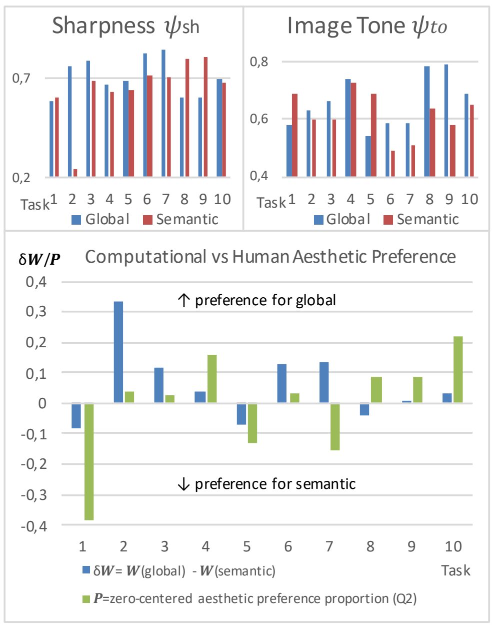
[2] http://www.mut1ny.com/face-headsegmentation-dataset

6 Conclusions

In this work, we implement and extend three techniques for neural style transfer with local control on mobile devices. We show that iterative, multi-style feed-forward and adaptive style transfers are complementary in terms of run-time, user control and visual quality, and conclude that offering all three in a single, end-user oriented mobile app achieves a high degree of flexibility for creating expressive results. We introduce an UI concept which mixes interactive brush-based stylizations with deferred, high-quality stylizations. Evaluating the mobile app on casual users shows that local control methods provide tangible benefits to this user group over global transfer, such as increased artistic freedom and superior stylization quality, even with little to no training. We consider further user studies regarding other



(a) Non-portrait survey tasks.



(b) Aesthetics scores vs. preference responses

Fig. 12: In b) the aesthetics score $W = \psi_{sh}\psi_{to}$ is measured by computing the sharpness ψ_{sh} using the *cpbd* metric [28] and the tone metric ψ_{to}, defined in [1]. The difference δW between global and semantic scores is used for relative comparison. The zero-centered aesthetic preference proportion is computed as $P = 0.5 - \frac{Q2_{global}}{Q2_{global}+Q2_{semantic}}$. Content Images © Giuseppe (cicrico) on flickr.com and Debbie Molle and Sestrjevitovschii Ina on Unsplash.com.

identified challenges, e. g., to start with a non-stylized content image. Another way to present the concept of sub-styles to users would be to create two different apps, one for the definition of (sub-)styles and one for the application to a content image. We introduce a mask-based training loss for feed-forward style transfer networks to map sub-styles to semantic image regions. A survey comparing global and semantically trained networks shows that learned semantic mappings significantly improve the semantic plausibility of the transfer, but also show that the stylization's aesthetics mostly depend on photographic qualities such as sharpness and contrast and less on semantic correspondences.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable feedback. This work was funded by the Federal Ministry of Education and Research (BMBF), Germany, for the AVA project 01IS15041.

References

1. Aydın, T.O., Smolic, A., Gross, M.: Automated aesthetic analysis of photographic images. IEEE Transactions on Visualization and Computer Graphics **21**, 31–42 (2015)
2. Bakhshi, S., Shamma, D.A., Kennedy, L., Gilbert, E.: Why We Filter Our Photos and How It Impacts Engagement. In: Proc. ICWSM, pp. 12–21 (2015)
3. Berry, M.: Re–imagining Place with Filters: More Than Meets the Eye. Journal of Creative Technologies **4** (2014)
4. Caesar, H., Uijlings, J., Ferrari, V.: Coco-stuff: Thing and stuff classes in context. In: Proc. CVPR, pp. 1209–1218 (2018)
5. Champandard, A.J.: Semantic Style Transfer and Turning Two-Bit Doodles into Fine Artworks. arXiv.org report 1612.04337 (2016)
6. Chen, D., Yuan, L., Liao, J., Yu, N., Hua, G.: StyleBank: An Explicit Representation for Neural Image Style Transfer. In: Proc. CVPR (2017)
7. DeCarlo, D., Santella, A.: Stylization and Abstraction of Photographs. ACM Transactions on Graphics **21**(3), 769–776 (2002)
8. Dev, K.: Mobile Expressive Renderings: The State of the Art. IEEE Computer Graphics and Applications **33**(3), 22–31 (2013)
9. Dumoulin, V., Shlens, J., Kudlur, M.: A learned representation for artistic style. In: Proc. ICLR, p. 9 (2017)
10. Gatys, L.A., Ecker, A.S., Bethge, M.: Image Style Transfer Using Convolutional Neural Networks. In: Proc. CVPR (2016)
11. Gatys, L.A., Ecker, A.S., Bethge, M., Hertzmann, A., Shechtman, E.: Controlling Perceptual Factors in Neural Style Transfer. In: Proc. CVPR, pp. 3730–3738 (2017)
12. Ghiasi, G., Lee, H., Kudlur, M., Dumoulin, V., Shlens, J.: Exploring the structure of a real-time, arbitrary neural artistic stylization network. arXiv.org report 1705.06830 (2017)

13. Gooch, A.A., Long, J., Ji, L., Estey, A., Gooch, B.S.: Viewing Progress in Non-Photorealistic Rendering Through Heinlein's Lens. In: Proc. NPAR (2010)
14. Huang, X., Belongie, S.: Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization. arXiv.org report 1703.06868 (2017)
15. Isenberg, T.: Interactive NPAR: What Type of Tools Should We Create? In: Proc. NPAR (2016)
16. Jing, Y., Yang, Y., Feng, Z., Ye, J., Song, M.: Neural Style Transfer: A Review. arXiv.org report 1705.04058 (2018)
17. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In: Proc. ECCV (2016)
18. Keep, D.: Artist with a Camera-Phone: A Decade of Mobile Photography, pp. 14–24. Palgrave Macmillan US, New York (2014)
19. Klingbeil, M., Pasewaldt, S., Semmo, A., Döllner, J.: Challenges in User Experience Design of Image Filtering Apps. In: Proc. MGIA, pp. 22:1–22:6 (2017)
20. Li, C., Wand, M.: Combining Markov Random Fields and Convolutional Neural Networks for Image Synthesis. In: Proc. CVPR (2016)
21. Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Diversified Texture Synthesis with Feed-Forward Networks. In: Proc. CVPR, pp. 266–274 (2017)
22. Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Universal style transfer via feature transforms. In: Advances in Neural Information Processing Systems, pp. 385–395 (2017)
23. Liao, J., Yao, Y., Yuan, L., Hua, G., Kang, S.B.: Visual Attribute Transfer through Deep Image Analogy. arXiv.org report (1705.01088) (2017)
24. Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, L.C.: Microsoft COCO: common objects in context. arXiv.org report 1405.0312 (2014)
25. Liu, X.C., Cheng, M.M., Lai, Y.K., Rosin, P.L.: Depth-aware Neural Style Transfer. In: Proc. NPAR (2017)
26. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proc. CVPR, pp. 3431–3440 (2015)
27. Luan, F., Paris, S., Shechtman, E., Bala, K.: Deep Photo Style Transfer. arXiv.org report 1703.07511 (2017)
28. Narvekar, N.D., Karam, L.J.: A no-reference image blur metric based on the cumulative probability of blur detection (cpbd). pp. 2678–2683 (2011)
29. Reimann, M., Klingbeil, M., Pasewaldt, S., Semmo, A., Döllner, J., Trapp, M.: MaeSTrO: A Mobile App for Style Transfer Orchestration using Neural Networks. In: Proceedings International Conference on Cyberworlds, pp. 9–16. IEEE (2018). DOI 10.1109/CW.2018.00016
30. Rudner, R.: On semiotic aesthetics. The Journal of Aesthetics and Art Criticism **10**(1), 67–77 (1951)
31. Salesin, D.H.: Non-Photorealistic Animation & Rendering: 7 Grand Challenges. Keynote talk at NPAR (2002)
32. Santella, A., DeCarlo, D.: Visual Interest and NPR: An Evaluation and Manifesto. In: Proc. NPAR (2004)
33. Seims, J.: Putting the artist in the loop. ACM SIGGRAPH Computer Graphics **33**(1), 52–53 (1999)
34. Selim, A., Elgharib, M., Doyle, L.: Painting style transfer for head portraits using convolutional neural networks. ACM Transactions on Graphics **35**(4), 129:1–129:18 (2016)
35. Semmo, A., Dürschmid, T., Trapp, M., Klingbeil, M., Döllner, J., Pasewaldt, S.: Interactive Image Filtering with Multiple Levels-of-control on Mobile Devices. In: Proc. MGIA (2016)
36. Semmo, A., Isenberg, T., Döllner, J.: Neural Style Transfer: A Paradigm Shift for Image-based Artistic Rendering? In: Proc. NPAR (2017)
37. Semmo, A., Trapp, M., Döllner, J., Klingbeil, M.: Pictory: Combining Neural Style Transfer and Image Filtering. In: Proc. SIGGRAPH Appy Hour (2017)
38. Shneiderman, B.: Creativity Support Tools: Accelerating Discovery and Innovation. Commun. ACM **50**(12), 20–32 (2007)
39. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv.org report 1409.1556 (2015)
40. Tanno, R., Matsuo, S., Shimoda, W., Yanai, K.: DeepStyleCam: A Real-Time Style Transfer App on iOS. In: Proc. MultiMedia Modeling, pp. 446–449 (2017)
41. Toyoura, M., Abe, N., Mao, X.: Painterly Image Generation Using Scene-Aware Style Transferring. In: Proc. International Conference on Cyberworlds (2016)
42. Toyoura, M., Abe, N., Mao, X.: Scene-Aware Style Transferring Using GIST, pp. 29–49. Springer, Berlin, Heidelberg (2017). DOI http://dx.doi.org/10.1007/978-3-662-56006-8_3
43. Ulyanov, D., Lebedev, V., Vedaldi, A., Lempitsky, V.S.: Texture Networks: Feed-forward Synthesis of Textures and Stylized Images. In: Proc. ICML (2016)
44. Ulyanov, D., Vedaldi, A., Lempitsky, V.S.: Instance Normalization: The Missing Ingredient for Fast Stylization. arXiv.org report 1607.08022 (2016)
45. Winnemöller, H.: NPR in the Wild. In: Image and Video based Artistic Stylisation. Springer (2013)
46. Zhang, H., Dana, K.: Multi-style Generative Network for Real-time Transfer. arXiv.org report 1703.06953 (2017)
47. Zhao, H., Rosin, P.L., Lai, Y.: Automatic semantic style transfer using deep convolutional neural networks and soft masks (1708.09641) (2017)