Computational Aesthetics in Graphics, Visualization, and Imaging
Paul Rosin (Editor)

EXPRESSIVE 2015

Image Stylization by Oil Paint Filtering using Color Palettes

Amir Semmo[1] Daniel Limberger[1] Jan Eric Kyprianidis[2] Jürgen Döllner[1]

[1]Hasso Plattner Institute, University of Potsdam, Germany†
[2]TU Berlin, Germany†

Figure 1: *Example of an automatically processed color image using our stylization technique with 20 derived color tones.*

Abstract

This paper presents an approach for transforming images into an oil paint look. To this end, a color quantization scheme is proposed that performs feature-aware recolorization using the dominant colors of the input image. In addition, an approach for real-time computation of paint textures is presented that builds on the smoothed structure adapted to the main feature contours of the quantized image. Our stylization technique leads to homogeneous outputs in the color domain and enables creative control over the visual output, such as color adjustments and per-pixel parametrizations by means of interactive painting.

Categories and Subject Descriptors (according to ACM CCS): I.4.3 [Computer Graphics]: Image Processing and Computer Vision—Enhancement—Filtering

1. Introduction

Image-based artistic rendering received significant attention in the past decades for visual communication, covering a broad range of techniques to mimic the appeal of artistic media [KCWI13]. Oil paint is considered to be among the most popular of the elementary media because of its qualities for subtle color blending and texturing [Sco05]. Simulating the visual characteristics of oil paint via image filtering, however, is a difficult task with respect to two main issues:

I1 The color distribution should be optimized to conform with a global color palette while preserving main feature contours.
I2 The paint texture should be oriented to the main feature curves to mimic the way an artist might paint with a brush.

In this work we present a technique for image stylization that employs (re-)colorization and non-linear image filtering to devise artistic renditions of 2D images with oil paint characteristics. Rather than attempting to simulate oil paint physically-based [BWL04, LBDF13] or via aligned strokes [Hae90, Her98, HE04, ZZXZ09], this work formulates I1 and I2 as sub-problems of image filtering. The first problem is solved by performing a recolorization, using the

† http://www.hpi3d.de | http://www.cg.tu-berlin.de

DOI: 10.2312/exp.20151188

optimization-based approach of Levin et al. [LLW04], with the dominant colors of the input image for quantization. This approach produces more homogeneous color distributions than local image filtering techniques and gives users more control in refining global color tones. The second problem is solved using the smoothed structure tensor [BBL*06], which is adapted to the feature contours of the quantized output, and bump-mapped to obtain flow-based paint textures in real-time. Furthermore, we propose an interactive painting interface that enables GPU-based per-pixel parametrizations via on-screen painting metaphors, enabling user-defined flow directions and paint texture shading. Finally, the quantized output and paint textures are blended to compose the final image (Figure 1).

The remainder of this paper is structured as follows. Section 2 reviews related work on image stylization and filtering, color quantization, and paint texture synthesis. Section 3 presents the methods used for image stylization with oil paint characteristics. Section 4 presents further results and implementation details, and states ideas for future work. Finally, Section 5 concludes this paper.

2. Related Work

Related work is found in the fields of image stylization and filtering, color quantization, and paint texture synthesis.

Image Stylization and Filtering. For artistic image stylization one can basically distinguish between (1) stroke-based and example-based methods, (2) region-based techniques, and (3) image filtering [KCWI13]. A classical method for stroke-based stylization is to iteratively align brush strokes of varying color, size, and orientation according to the input image [Hae90, Her98, GCS02, HE04, ZZXZ09, ZZ10, LSF10]. For an overview on this topic we refer to the survey by Hegde et al. [HGT13]. An essential building block for region-based stylization is segmentation. Several methods based on a mean shift have been proposed for image abstraction [DS02, WLL*06] and the simulation of artforms and fabrics, such as stained glass [Mou03] and felt [OM06]. However, the rough boundaries of the segmented regions created by these methods would require elaborate post-processing to achieve color blending characteristics of oil paint. To modify areas or image regions more abrasively, local image filtering that operates in the spatial domain may be used, which is often based on anisotropic diffusion [Wei98]. A popular choice is the bilateral filter, which works by weight averaging pixel colors in a local neighborhood according to their distances in space and range [TM98], e.g., for image-based abstraction [WOG06]. Flow fields have been used to adapt bilateral filtering [KD08, KLC09] and particle-based techniques [YLK12] to local image structures. In this work, quantized color outputs are smoothed by flow-based Gaussian filtering to provide smooth interpolations at curved boundaries, however, we restrain from weighting in the color domain to achieve firmer color blendings. Additional filter categories include morphological operations based on dilation and erosion (e.g., for watercolor rendering [BKTS06]), and global optimization schemes for image decomposition, such as weighted least squares [FFLS08], local extrema for edge-preserving smoothing [SSD09], and L_0 gradient minimization [XLXJ11]. However, these techniques are less suited for rendering with a constrained color palette, which instead requires filtering schemes found in color quantization.

Color Image Quantization and Abstraction. The original goal of color quantization is to approximate an image with a relatively small number of colors while minimizing color abbreviations. Popular approaches are based on the median-cut algorithm [Hec82], clustering using octrees [GP88], k-means [KMN*02], and adaptive segmentation via perceptual models [CPMR05] or roughness measures [YMC*14]. However, these algorithms may absorb colors because of their global optimization scheme, or only consider the color space. Other approaches also consider the spatial space via local luminance mapping and thresholding [WOG06], and in its optimization scheme to preserve image details [YLS14], but are mainly self-organizing with respect to the derived color palette. By contrast, we propose a technique that derives colors from local image regions using a scoring system for optimal distribution, and uses the derived color palette for image quantization. At this, the optimization framework of Levin et al. [LLW04] is parametrized to propagate seed pixels (using colors of the derived palette) to the remaining pixels at the premise that pixels in space with similar intensities should have similar colors, with an additional pass for luminance quantization. The output is then post-processed using a flow-based Gaussian filter to provide firm color blendings.

Interactive Paint Texture Synthesis. Contrary to physically-based paint modeling [CBWG10, LBDF13], we separate the paint texture synthesis from color abstraction. Our computation is based on the smoothed structure tensor [BBL*06] and an eigenanalysis to obtain gradient and tangent information for directed smoothing similar to [KD08]. The smoothing pass is adapted to the main feature contours retrieved via the flow-based Laplacian of Gaussian (FLoG) [KK11] to avoid ringing artifacts and provide outputs with adaptive detail. Similar to [Her02], line integral convolution [CL93] and bump mapping via a Phong-based shading [Pho75] are used to synthesize a normal-mapped height texture that is aligned to the local feature orientations. Our computation performs in real-time and is defined as a composition of ordered, parametrizable image processing steps, thus it may also be injected by user-specified motions [HE04, OMG05] via painting [HH90]. In contrast to placing dynamic or static rendering primitives [SIMC07], we extend the concept of specialized local parametrizations [AWB06, TABI07] to a generalized brush-based painting within effect-parameter spaces, which is exemplified for local orientation and shading corrections (e.g., glossiness of water).

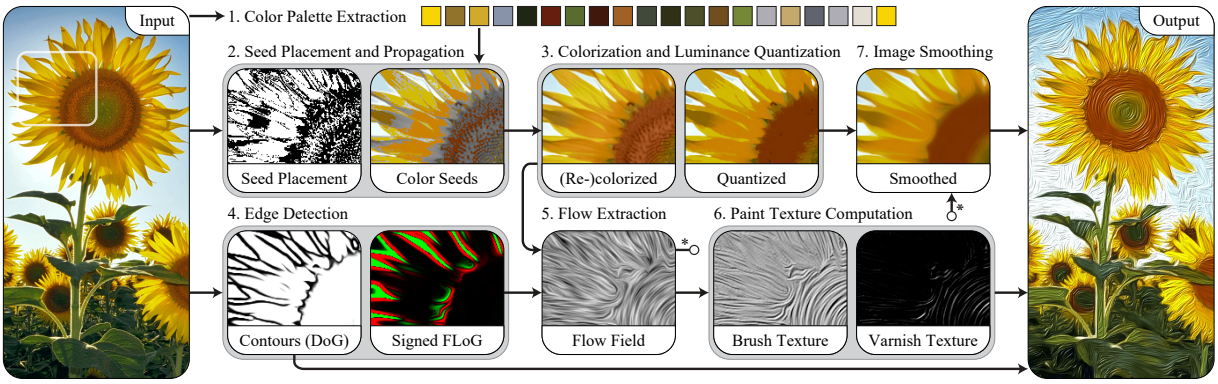

Figure 2: *Overview of the different stages performed for our stylization technique that automatically transforms color images (left) to filtered variants with an oil paint look (right). For a detailed description of the filtering stages the reader is referred to Section 3.*

3. Method

An overview of our stylization technique is shown in Figure 2. The processing starts with extracting a user-defined number of dominant colors from the input image (Section 3.1). Next, the recolorization based on the optimization approach by [LLW04] is performed to quantize the chrominance and luminance channels based on the derived color palette (Section 3.2). In parallel, contour lines are extracted from the input image via difference-of-Gaussians (DoG) and Laplacian-of-Gaussians (LoG) filtering, where the latter is used for flow field computation (Section 3.3). The computation is based on the structure tensor [BBL*06], which is adaptively smoothed according to the derived contour lines to avoid filtering across feature boundaries. An eigenanalysis is used to obtain gradient and tangent information for line integeral convolution [CL93]. The synthesized flow image is then bump-mapped and Phong-shaded to obtain the paint textures. In addition, the local orientation information is used for flow-based Gaussian smoothing of the quantized output. Finally, the color image and paint textures are blended to compose the final result. The filtering stages are presented in more detail in the following sections.

3.1. Dominant Color Extraction

To simulate the way artists paint with a limited number of base colors, dominant (and contemporary) colors need to be derived from the input image. Common approaches use global optimization schemes that cluster similar colors and determine the base colors from the largest clusters, such as the median-cut algorithm [Hec82]. These approaches, however, may produce false colors or absorb contemporary colors.

Our approach computes a color palette $\mathcal{P}$ with colors derived from local image regions of the input image. The main idea is to use a scoring system to classify image regions according to their normalized entropy by penalizing the extraction from features with similar colors. The color extraction is performed incrementally, starting with an empty palette. Extracted colors are inserted after each iteration.

Suppose that a color palette $\mathcal{P}$ is given. To find a new color for the palette, we seek for an image region R with minimal score

$$\mathcal{S}(R) = \frac{\eta(R)}{|R| \cdot \bar{\omega} \cdot \bar{L}}. \tag{1}$$

Here,

$$\eta(R) = -\sum_{c \in R} p_R(c) \log_2 p_R(c) \tag{2}$$

is the the entropy of the image, computed from the probabilities $p_R(c)$ that a certain color c appears in the image region R, and $|R|$ denotes the area of the region. In addition, the region score is weighted according to the average lightness

$$\bar{L} = \frac{\sum_{p \in R} L(p)}{|R|}, \tag{3}$$

to prefer vivid colors. For the lightness L, gamma corrected intensities as defined by the sRGB standard are used. Finally, the region score is weighted according to the average minimum color distance,

$$\bar{\omega} = \frac{\sum_{p \in R} \omega(p)^2}{|R|} \quad \text{with} \quad \omega(p) = \min_{c \in \mathcal{P}} \Delta E\big(I(p), c\big), \tag{4}$$

to colors in the current palette $\mathcal{P}$ to favor colors not yet present, where I corresponds to the input image.

To find a region with minimum score we proceed heuristically, using the following algorithm. The steps are performed iteratively in CIE-Lab color space, n-times in total for a target palette size of n:

1. **Color Difference Mask**: ω is precomputed via the minimal color distance (ΔE) for pixels in I to $\{c \in \mathcal{P}\}$.

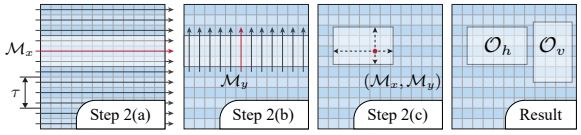

Figure 3: *Overview of the region selection (horizontal pass).*

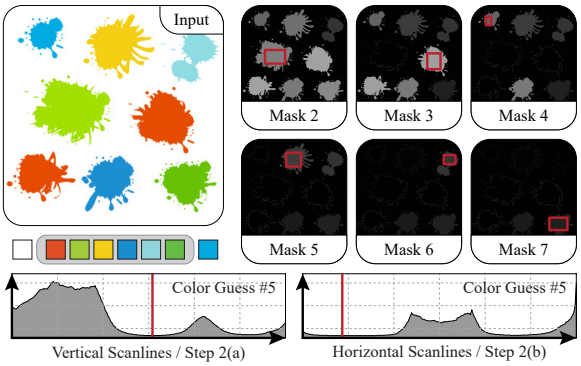

Figure 4: *Color difference masks computed for a test image with 8 colors (including white), overlaid by the color extraction regions. The region scores for the fifth color guess are shown at the bottom.*

Figure 5: *Comparison of the median-cut algorithm to our approach for dominant color extraction (the 10 dominant colors are sorted from left to right).*

2. **Region Search**: Regions are computed for the horizontal and vertical scanlines of the input image. The color is extracted from the region with the better score (Figure 3):
   a. **First Pass**: $\mathcal{S}(R)$ is computed for all regions of width $\tau > 0$ along the horizontal/vertical scanlines. The scanline $\mathcal{M}_x$ with minimal $\mathcal{S}(R)$ is determined.
   b. **Second Pass**: The first pass is repeated for the orthogonal scanlines, bounded by the region defined at $\mathcal{M}_x$. The scanline $\mathcal{M}_y$ with minimal $\mathcal{S}(R)$ is determined.
   c. **Area Computation**: $\mathcal{S}(R)$ is computed iteratively for growing horizontal and vertical boundaries around pixel $(\mathcal{M}_x, \mathcal{M}_y)$ until a minimum value is reached.
3. **Color Extraction**: Once a region R with minimum score has been identified, a representative color is extracted from the region and inserted into the palette $\mathcal{P}$. Thereby, the representative color is computed by finding a mode in the box-filtered histograms of the chrominance channels.

An example of the iterative computation of dominant colors, color difference masks, and optimal region selection is given in Figure 4.

Discussion. For all examples in this paper we set the width for the region search to $\tau = 3$ (step 2a) and sorted the colors in $\mathcal{P}$ by their weighted count in the input image (i.e., if $\Delta E < \xi$) and color difference to previously sorted colors. We empirically determined $\xi = \Delta E^*_{ab} = 7$ as a good default value for thresholding, where $\Delta E^*_{ab} \approx 2.3$ corresponds to a just noticeable color difference in CIE76 [MEO94]. Our algorithm is compared to the median-cut algorithm in Figure 5. Notice how colors of single features are accurately represented and not merged with colors of other features (e.g., the butterfly in Figure 5a, the eagle's beak in Figure 5c, the guitar in Figure 5e, red tones in Figure 5f). Further, we noticed that our approach is more resistant to noise as one can observe by the green background in Figure 5c, where a clustering approach may also derive side tones. Furthermore, we observed that the number of color extractions significantly affects if image features are accurately represented or not. To this end, one could use a metric to control the color coverage, e.g., to derive colors until the maximum and/or mean of the color difference mask (ω) falls below a given threshold. This way, more colors could be derived for colorful images without parameter adjustments. Finally, we believe that the accuracy of our algorithm can be further improved when using a generalized region growing technique to derive colors from feature-aligned (non-rectangular) regions, but which remains subject to future work.

3.2. Image Quantization using Colorization

Our goal is to quantize the input image I using the extracted dominant colors. We formulate this task as an optimization problem, performing a (re-)colorization [LLW04]: Given the intensity of the input image and a number of colored seed pixels, the colors should be propagated to the remaining pixels such that pixels with similar intensities have similar colors.

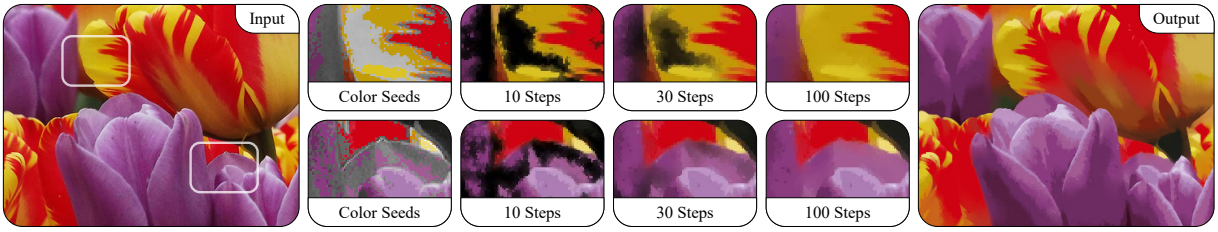

Figure 6: *Image quantization using the algorithm described in Section 3.2 with a palette of 26 colors and $\alpha = 5.5$ for automatic seed placement. The optimization problem was iteratively solved with the "generalized minimum residual" method [SS86].*

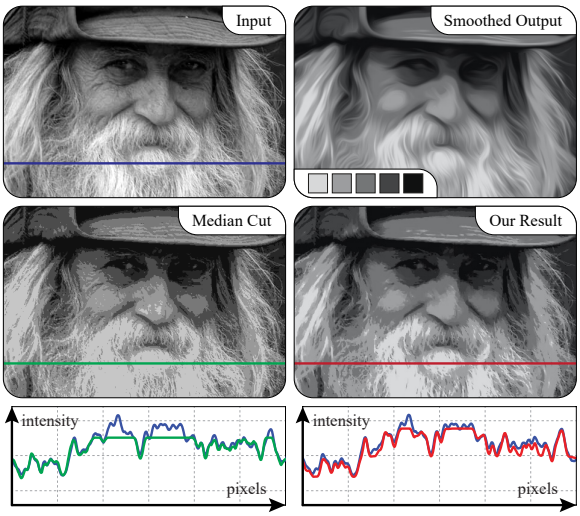

Figure 7: *Comparison of our quantization method with the median-cut algorithm. Notice how our approach preserves feature contrasts at a better scale. The smoothed quantization result is based on a flow-based Gaussian filter [KD08].*

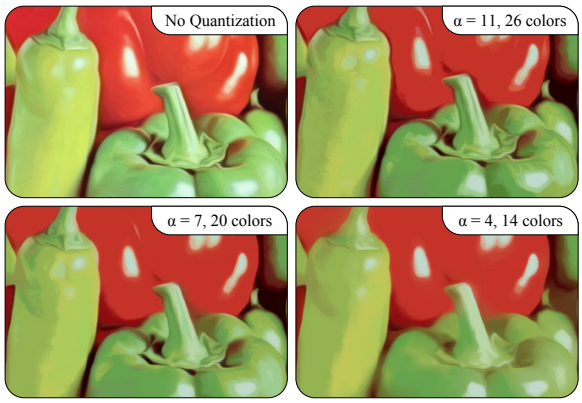

Figure 8: *Example for adjusting the seed placement threshold α and size of the color palette to control the level of abstraction.*

First Pass: Colorization. The optimization is performed with the constraint that *seed pixels* $c(\mathbf{r_i})$ are set to the respective color component of the dominant color c_p in $\mathcal{P}$ with minimal distance:

$$c(\mathbf{r_i}) = \arg\min_{c_p \in \mathcal{P}} \Delta E(c_p, I(\mathbf{r})) \tag{5}$$

if and only if the minimal color distance falls below a threshold $\alpha > 0$. For a given color channel $\mathcal{C}$, the recolorized channel $\mathcal{C}'$ is then computed via the objective function

$$\arg\min_{\mathcal{C}} \sum_{\mathbf{r}} \Big(\mathcal{C}(\mathbf{r}) - \sum_{\mathbf{s} \in \mathcal{N}(\mathbf{r})} w_{\mathbf{rs}}\, \mathcal{C}(\mathbf{s}) \Big)^2, \quad \text{subject to } \mathcal{C}(\mathbf{r}_i) = c(\mathbf{r}_i) \text{ for } i \in \mathit{Seeds} \tag{6}$$

where $w_{\mathbf{rs}}$ denotes the squared difference between the luminance values of the pixels $\mathbf{r}$ and $\mathbf{s}$, and $\mathcal{N}(\mathbf{r})$ being the 8-connected neighborhood of $\mathbf{r}$. The objective function then yields a large sparse system of linear equations that is solved for both chrominance channels of the input image in CIE-Lab color space using the "generalized minimum residual" method [SS86].

Second Pass: Luminance Quantization. We introduce a second pass for luminance quantization, where the objective function is used to recompute the luminance channel. To this end, we reformulate the weighting function $w_{\mathbf{rs}}$ to compute the squared difference between the two *hue* values at pixels $\mathbf{r}$ and $\mathbf{s}$ of the recolorized image in HSV color space. The recomputed luminance channel is then combined with the color channels of the first pass and results in a recolorized image with potential color transitions within regions.

Discussion. Examples using our method are presented in Figure 6 and Figure 7. Contrary to traditional quantization methods, we observed that our approach produces outputs with better color contrasts, primarily because of the scoring system introduced in Section 3.1. The threshold for automatic seed placement should be set greater or equal to the threshold used for the color extraction to use all colors of the derived palette, e.g., $\alpha = \xi = 7$ as a good default value. The threshold may also be set lower, i.e., to initially place fewer color seeds and thus induce more color blendings, which may be

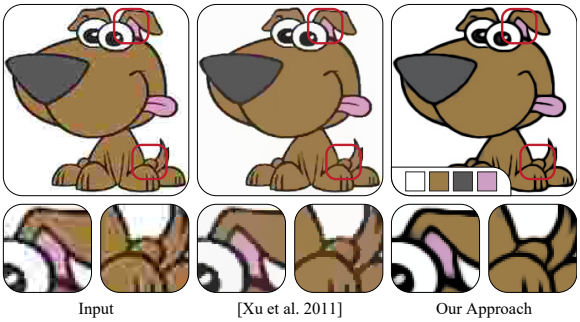

Figure 9: *Comparison of image smoothing via L0 gradient minimization [XLXJ11] with our quantization technique, combined with the output of a DoG filter, for JPEG artifact removal.*

combined with minimizing the sorted color palette to control the level of abstraction (Figure 8).

Finally, we experimented using our method for JPEG artifact removal of clip-arts (Figure 9), where our approach produces accurate results and may also be used to easily redefine single colors.

3.3. Paint Texture

Oil painting is a time-consuming process that often comprises multiple layers of paint and drying phases. During finishing, thin protective layers (*varnish*) may be coated onto the paint for protection against dirt and dust, and to even out its final appearance. This yields two basic characteristics of oil paint textures: (1) reliefs of varying thickness according to the used brushes and applied number of layers with (2) a matt or glossy tint. The first may be perceived as subtle shading or shadowing that is caused by external, off-image illumination, and the second as specular highlighting. To simulate both characteristics, first, a flow image is synthesized using the gradient and tangent information for line integral convolution [CL93].

Adaptively Smoothed Structure Tensor. Local orientation information is derived from an eigenanalysis of the smoothed structure tensor [BBL*06], a method that provides stable estimates and can be computed in real-time [KD08]. Because the quantized image may provide large areas of solid color tones, structure tensors with low gradient magnitudes are replaced by inpainted information for relaxation and to avoid singularities. For a detailed description of the integration and inpainting process we refer to [KK11]. In addition, we perform an adaptive smoothing of the structure tensors to avoid filtering over feature boundaries to obtain more accurate results (Figure 10). The basic idea is to derive the sign of the flow-based Laplacian of Gaussian (FLoG) for thresholding. The Gaussian filter kernel is then adapted to exclude pixel values from weight averaging when the difference in the

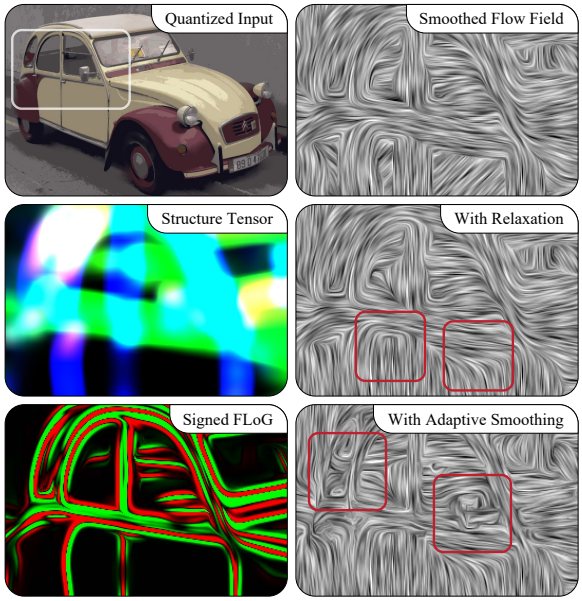

Figure 10: *Enhancements for the smoothed structure tensor (here: $\sigma = 8$) to derive flow fields. Middle: relaxation to avoid singularities, bottom: adaptive smoothing based on the signed FLoG.*

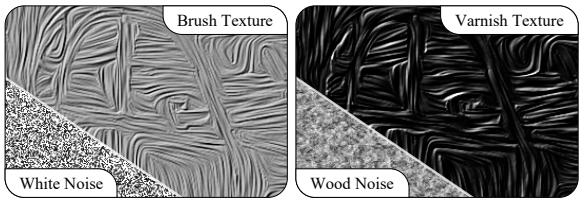

Figure 11: *Paint textures computed for the flow field in Figure 10 and parameters $\mathcal{K} = (8.0, 10.0, 3.0, 8.0)$.*

signed FLoG to the origin pixel reaches a given threshold, e.g., when the sign flips while crossing feature boundaries.

Paint Textures. Line integral convolution [CL93] is performed along the stream lines defined by the minor eigenvector field of the adaptively smoothed structure tensor. Procedural monochromatic noise is blurred in gradient flow direction, which is interpreted as fine-grained height variations with normals N and illuminated using a directional light source L. Principles of Phong shading [Pho75] are then used to render the brush texture $\mathcal{T}_B$ and varnish texture $\mathcal{T}_V$ with pixels $p \in P$:

$$\begin{aligned} \mathcal{T}_B(p) &= 0.5 + N(p) \cdot L, \\ \mathcal{T}_V(p) &= k_{specular} \cdot (N(p) \cdot L)^{k_{shininess}}. \end{aligned} \tag{7}$$

At this, a set $\mathcal{K} = (\sigma_b, k_{scale}, k_{specular}, k_{shininess})$ is used to parametrize the brush and varnish textures (Figure 11), where σ_b and k_{scale} control the smoothing and relief strength of

the brush texture, and the other parameters are used for the varnish texture. We experimented with different noise implementations and observed that high frequency noise with no specular highlights simulate brush characteristics quite naturally, while low frequency, discontinuous noise and subtle specular highlights work well to simulate varnish. The computation of the paint textures is local and fast, and thus enables interactive per-pixel parametrizations (Section 3.4).

Image Composition. The quantized image is post-processed by a Gaussian filter that is adapted to the local orientation to create smoothed outputs at curved boundaries [KD08]. Finally, the brush texture is multiplied with the smoothed color image, and the intermediate result is blended with the varnish texture using *linear dodge* as blend mode. Optionally, contours are enhanced by using a flow-based difference-of-Gaussians filter [KLC09].

3.4. Interactive Painting

When tweaking the output of an image filter, users typically strive for a global parametrization-trade-off that corresponds to multiple visual requirements in different image regions. Recently, consumer photo-editing products started to extend the concept of non-destructive per-pixel parametrizations from alpha masking to a small set of image computations (adjustments) by means of parameter masks. We extended this approach by exposing selected filter parameters as image layers that can be parametrized via painting metaphors. This enables (1) more artistic control over the stylization process, and (2) the modification of intermediate filters outputs with an inadequate local parametrization (Figure 12).

Our method extends the concept of a specialized, locally computed parametrization [TABI07] to a generalized configuration by brush-painting within parameter spaces of local image filters. At this, parameters are encoded as *parameter maps* and adjusted via *virtual brush models* according to well-defined operation sequences. Thereby, we denote the following conceptual objects:

- A *parameter map* is a typed map that substitutes a uniform filter parameter and can be manipulated by actions.

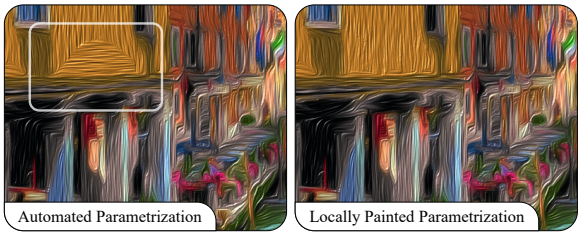

Figure 12: *Failure case of a filtering result and its manual correction using our painting interface for per-pixel parametrization.*

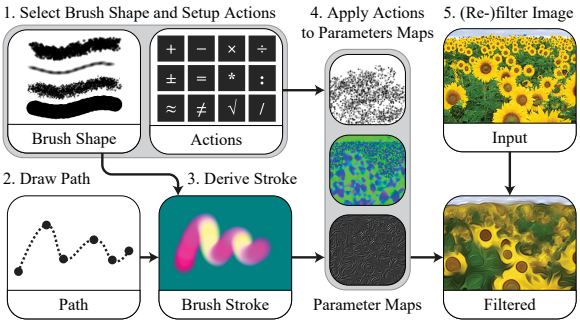

Figure 13: *Schematic overview of our per-pixel parametrization interface: brush strokes are derived from drawn paths and applied to selected parameter maps. The maps' modified local parameters are then used to (re-)filter the image.*

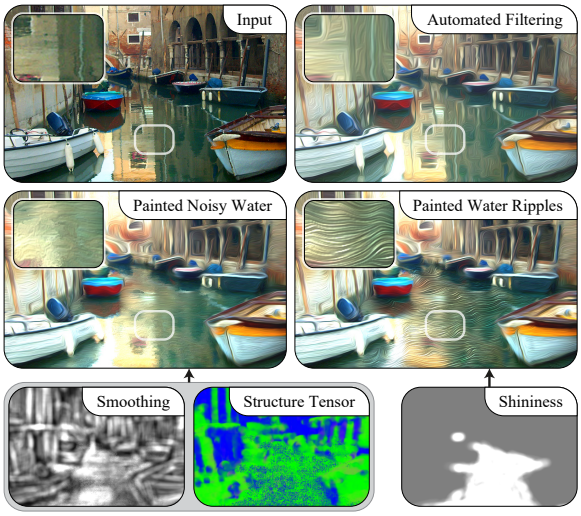

Figure 14: *Example of manual corrections made for parameter layers used as input for the paint texture synthesis: smoothing of the flow field, flow direction, and shininess.*

- A *brush* provides a sequence of actions mapped to parameter maps (map-action pairs) weighted by a brush shape.
- A *brush shape* maps interpolated user input (e.g., position, direction, pressure) to action-compliant input masks (*brush strokes*). Brush shapes can be of arbitrary complexity, e.g., 3D brushes [CBWG10], and are encoded as distance maps for local weighting of actions.
- *Actions* define local computations to modify parameter maps, e.g., *add* or *blur*. An action is filter-independent and can be assigned to any supported parameter map.

This hardware-accelerated process is schematized in Figure 13 and is used to locally adjust the parameters defined by the set $\mathcal{K}$ for the paint texture synthesis. An example is given in Figure 14.

4. Results

We have implemented the dominant color extraction using C++, the colorization and filtering stages on the GPU with CUDA, and the painting interface with Qt. Using an Intel® Xeon™ 4× 3.06 GHz and NVidia® GTX 760, a 800×600 pixel image is processed in 50 seconds for a palette size of 25. The color extraction is currently the limiting stage, followed by the colorization, and the paint texture synthesis that performs in real-time for images with HD resolution. We observed that the color extraction also provides stable estimates when processing downscaled images (up to the second pyramid level) to speed up the processing by a factor of two.

A collection of images processed by our technique is shown in Figure 15. As can be seen, the level of abstraction is adapted to the image contents, e.g., more distinct colors were extracted in colorful images (e.g., *Venice*) and wide filter kernels were used for Gaussian smoothing for soft color blendings (e.g., the *girl's face*). Moreover, noise functions play a major role for stylization, e.g., wood noise is a good basis to simulate thick paint (e.g., the *flowers*) and white noise is quite effective to simulate detail brushes (e.g., the *still life*).

Limitations. An inherent limitation of our technique is that it does not reach the individual painting style diversity of manual or semi-automatic approaches. Yet the proposed painting interface gives users some local control. Further, the current implementation does not provide a high-level control for the level of abstraction, but requires a manual parametrization of color thresholds. Furthermore, image features with smooth color transitions may not be represented as desired but with hard transitions, depending on the seed color placement. Finally, the performance of the colorization stage currently does not enable interactive color refinements, but we believe this issue could be alleviated by visualizing intermediate results of the iterative solver for visual feedback.

Future Work. We see multiple directions for future work. A major strength of our technique is that the stylization is based on global color palettes that may be easily refined interactively. Here, we plan to implement a color transfer functionality to adapt the colors of the derived palette to colors of a target palette or image. Second, the color extraction may be generalized to support feature-aligned (non-rectangular) regions for a more robust extraction, e.g., via region growing. Third, adaptive colorization and filtering approaches could be used to stylize certain image features with more detail than other features. Here, an additional weight for image saliency could be used to direct the color extraction and colorization processes to the more salient features of an image, and thus capturing more detail with respect to the derived color palette. Finally, the extension of our technique to video is of particular interest. Here, the colorization could be extended by a temporal constraint as proposed in [LLW04], and an optical flow could be used to stabilize the paint texture synthesis.

5. Conclusions

In this paper, we have presented an approach for transforming images into filtered variants with an oil paint look. The proposed color extraction and colorization methods enable to quantize color images according to their dominant color palette. Results show that our quantization scheme is able to represent selected image features accurately and provide homogeneous outputs in the color domain. In addition, we proposed an approach for real-time synthesis of paint textures that enables local refinements via interactive, per-pixel parametrizations. Several results demonstrate the manifold application of our approach to different genres of photography.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments and Holger Winnemöller for his support on the *flowpaint* research project. This work was partly funded by the Federal Ministry of Education and Research (BMBF), Germany, within the InnoProfile Transfer research group "4D*n*D-Vis" (`www.4dndvis.de`), and was partly supported by the ERC through grant ERC-2010-StG 259550 (XSHAPE).

References

[AWB06] ANJYO K.-I., WEMLER S., BAXTER W.: Tweakable Light and Shade for Cartoon Animation. In *Proc. NPAR* (2006), pp. 133–139.

[BBL*06] BROX T., BOOMGAARD R., LAUZE F., WEIJER J., WEICKERT J., MRÁZEK P., KORNPROBST P.: Adaptive Structure Tensors and their Applications. *Visualization and Processing of Tensor Fields* (2006), 17–47.

[BKTS06] BOUSSEAU A., KAPLAN M., THOLLOT J., SILLION F. X.: Interactive Watercolor Rendering with Temporal Coherence and Abstraction. In *Proc. NPAR* (2006), pp. 141–149.

[BWL04] BAXTER W., WENDT J., LIN M. C.: IMPaSTo: A Realistic, Interactive Model for Paint. In *Proc. NPAR* (2004), pp. 45–148.

[CBWG10] CHU N., BAXTER W., WEI L.-Y., GOVINDARAJU N.: Detail-preserving Paint Modeling for 3D Brushes. In *Proc. NPAR* (2010), pp. 27–34.

[CL93] CABRAL B., LEEDOM L. C.: Imaging Vector Fields Using Line Integral Convolution. In *Proc. ACM SIGGRAPH* (1993), pp. 263–270.

[CPMR05] CHEN J., PAPPAS T. N., MOJSILOVIC A., ROGOWITZ B.: Adaptive Perceptual Color-Texture Image Segmentation. *IEEE Trans Image Processing 14*, 10 (2005), 1524–1536.

[DS02] DECARLO D., SANTELLA A.: Stylization and Abstraction of Photographs. *ACM Trans. Graph. 21*, 3 (2002), 769–776.

[FFLS08] FARBMAN Z., FATTAL R., LISCHINSKI D., SZELISKI R.: Edge-Preserving Decompositions for Multi-Scale Tone and Detail Manipulation. *ACM Trans. Graph. 27*, 3 (2008), 67:1–67:10.

[GCS02] GOOCH B., COOMBE G., SHIRLEY P.: Artistic Vision: Painterly Rendering Using Computer Vision Techniques. In *Proc. NPAR* (2002), pp. 83–ff.

Figure 15: *Image stylization results produced with our oil paint filtering technique.*

[GP88] GERVAUTZ M., PURGATHOFER W.: A Simple Method for Color Quantization: Octree Quantization. In *New Trends in Computer Graphics*. Springer Berlin Heidelberg, 1988, pp. 219–231.

[Hae90] HAEBERLI P.: Paint by Numbers: Abstract Image Representations. *SIGGRAPH Comput. Graph. 24*, 4 (1990), 207–214.

[HE04] HAYS J., ESSA I.: Image and Video Based Painterly Animation. In *Proc. NPAR* (2004), pp. 113–120.

[Hec82] HECKBERT P.: Color Image Quantization for Frame Buffer Display. *SIGGRAPH Comput. Graph. 16*, 3 (1982), 297–307.

[Her98] HERTZMANN A.: Painterly Rendering with Curved Brush Strokes of Multiple Sizes. In *Proc. ACM SIGGRAPH* (1998), pp. 453–460.

[Her02] HERTZMANN A.: Fast Paint Texture. In *Proc. NPAR* (2002), pp. 91–96.

[HGT13] HEGDE S., GATZIDIS C., TIAN F.: Painterly rendering techniques: a state-of-the-art review of current approaches. *Comp. Anim. Virtual Worlds 24*, 1 (2013), 43–64.

[HH90] HANRAHAN P., HAEBERLI P.: Direct WYSIWYG Painting and Texturing on 3D Shapes. *SIGGRAPH Comput. Graph. 24*, 4 (1990), 215–223.

[KCWI13] KYPRIANIDIS J. E., COLLOMOSSE J., WANG T., ISENBERG T.: State of the 'Art': A Taxonomy of Artistic Stylization Techniques for Images and Video. *IEEE Trans. Vis. Comput. Graphics 19*, 5 (2013), 866–885.

[KD08] KYPRIANIDIS J. E., DÖLLNER J.: Image Abstraction by Structure Adaptive Filtering. In *Proc. EG UK TPCG* (2008), pp. 51–58.

[KK11] KYPRIANIDIS J. E., KANG H.: Image and Video Abstraction by Coherence-Enhancing Filtering. *Comput. Graph. Forum 30*, 2 (2011), 593–602.

[KLC09] KANG H., LEE S., CHUI C. K.: Flow-Based Image Abstraction. *IEEE Trans. Vis. Comput. Graphics 15*, 1 (2009), 62–76.

[KMN*02] KANUNGO T., MOUNT D. M., NETANYAHU N. S., PIATKO C. D., SILVERMAN R., WU A. Y.: An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Trans. Pattern. Anal. Mach. Intell. 24*, 7 (2002), 881–892.

[LBDF13] LU J., BARNES C., DIVERDI S., FINKELSTEIN A.: RealBrush: Painting with Examples of Physical Media. *ACM Trans. Graph. 32*, 4 (2013), 117:1–117:12.

[LLW04] LEVIN A., LISCHINSKI D., WEISS Y.: Colorization Using Optimization. *ACM Trans. Graph. 23*, 3 (2004), 689–694.

[LSF10] LU J., SANDER P. V., FINKELSTEIN A.: Interactive Painterly Stylization of Images, Videos and 3D Animations. In *Proc. ACM I3D* (2010), pp. 127–134.

[MEO94] MAHY M., EYCKEN L., OOSTERLINCK A.: Evaluation of Uniform Color Spaces Developed after the Adoption of CIELAB and CIELUV. *Color Research & Application 19*, 2 (1994), 105–121.

[Mou03] MOULD D.: A Stained Glass Image Filter. In *Proc. EGRW* (2003), pp. 20–25.

[OM06] O'DONOVAN P., MOULD D.: Felt-based Rendering. In *Proc. NPAR* (2006), pp. 55–62.

[OMG05] OLSEN S. C., MAXWELL B. A., GOOCH B.: Interactive Vector Fields for Painterly Rendering. In *Proc. Graphics Interface* (2005), pp. 241–247.

[Pho75] PHONG B. T.: Illumination for Computer Generated Pictures. *Commun. ACM 18*, 6 (1975), 311–317.

[Sco05] SCOTT M.: *Oil Painter's Bible*. Chartwell Books, 2005.

[SIMC07] SCHWARZ M., ISENBERG T., MASON K., CARPENDALE S.: Modeling with Rendering Primitives: An Interactive Non-photorealistic Canvas. In *Proc. NPAR* (2007), pp. 15–22.

[SS86] SAAD Y., SCHULTZ M. H.: GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. and Stat. Comput. 7*, 3 (1986), 856–869.

[SSD09] SUBR K., SOLER C., DURAND F.: Edge-preserving Multiscale Image Decomposition based on Local Extrema. *ACM Trans. Graph. 28*, 5 (2009), 147:1–147:9.

[TABI07] TODO H., ANJYO K.-I., BAXTER W., IGARASHI T.: Locally Controllable Stylized Shading. *ACM Trans. Graph. 26*, 3 (2007), 17:1–17:7.

[TM98] TOMASI C., MANDUCHI R.: Bilateral Filtering for Gray and Color Images. In *Proc. ICCV* (1998), pp. 839–846.

[Wei98] WEICKERT J.: *Anisotropic diffusion in image processing*, vol. 1. Teubner Stuttgart, 1998.

[WLL*06] WEN F., LUAN Q., LIANG L., XU Y.-Q., SHUM H.-Y.: Color Sketch Generation. In *Proc. NPAR* (2006), pp. 47–54.

[WOG06] WINNEMÖLLER H., OLSEN S. C., GOOCH B.: Real-Time Video Abstraction. *ACM Trans. Graph. 25*, 3 (2006), 1221–1226.

[XLXJ11] XU L., LU C., XU Y., JIA J.: Image Smoothing via L_0 Gradient Minimization. *ACM Trans. Graph. 30*, 6 (2011), 174:1–174:12.

[YLK12] YOON J.-C., LEE I.-K., KANG H.: Video Painting Based on a Stabilized Time-Varying Flow Field. *IEEE Trans. Vis. Comput. Graphics 18* (2012), 58–67.

[YLS14] YU J., LU C., SATO Y.: Sparsity-based Color Quantization with Preserved Image Details. In *SIGGRAPH Asia 2014 Posters* (2014), pp. 32:1–32:1.

[YMC*14] YUE X., MIAO D., CAO L., WU Q., CHEN Y.: An efficient color quantization based on generic roughness measure. *Pattern Recognition 47*, 4 (2014), 1777–1789.

[ZZ10] ZHAO M., ZHU S.-C.: Sisley the Abstract Painter. In *Proc. NPAR* (2010), pp. 99–107.

[ZZXZ09] ZENG K., ZHAO M., XIONG C., ZHU S.-C.: From Image Parsing to Painterly Rendering. *ACM Trans. Graph. 29*, 1 (2009), 2:1–2:11.

Original photographs used in Figure 10, Figure 15 (car), and Figure 5a/d/e courtesy Phillip Greenspun. Photographs from flickr.com kindly provided under Creative Commons license by matthiashn (Figure 1), Vincent van der Pas (Figure 2), Gulsen Ozcan (Figure 6), Akaporn Bhothisuwan (Figure 7), Jelto Buurman (Figure 15 / still life), Christophe Chenevier (Figure 15 / girl), and Navdeep Raj (Figure 15 / flowers).