Vision, Modeling, and Visualization (2016)
M. Hullin, M. Stamminger, and T. Weinkauf (Eds.)

Artistic Composition for Painterly Rendering

T. Lindemeier M. Spicker O. Deussen

University of Konstanz, Germany

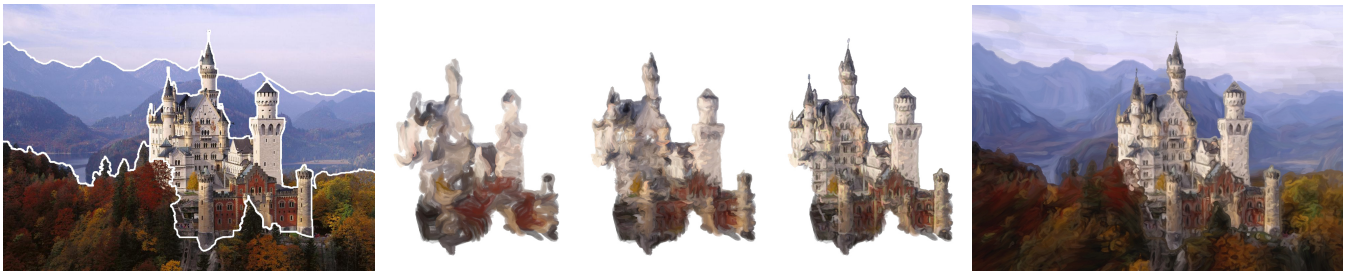


Figure 1: We decompose an input image into regions (left) where each region is separated into detail layers which are then superposed in a composition step (center). This enables us to create painterly rendering results with much less strokes and colors and enhanced artistic freedom. As an example, the castle on the right is painted with more details than its background.

Abstract

We present a technique for painterly renderings that follows a decomposition of the canvas into a set of regions and layers (coarse to fine). The regions reflect the spatial arrangement of the composition and the order in which the painting is to be created (typically back to front), and are produced in a way that new strokes only minimally paint over existing ones. Layers reflect the application of tools and are optimized for certain brush sizes. The number of strokes and colors that are needed to represent an input image are minimized by this decomposition, which is good for software, but essential for hardware-based rendering. Our method allows us to apply different painting styles to different regions as well as layers, and to create painterly renderings with more artistic freedom. We demonstrate our decomposition technique on images that are processed using hierarchical segmentation techniques.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

Painterly rendering creates artistically-looking images using virtual brush strokes. A typical strategy is to start with large brushes and later add details using smaller strokes. In addition to purely software-based rendering systems there exist a number of painting machines that work with real paint and tools [Coh12, Wik12, TFL12, Arm12, Bae13, Gro13, KM13, DLPT12]. To convert an image or photograph to painterly rendering, the input is typically blurred to create coarse strokes [Her98, HE04]. The color of these strokes is usually constant and blurred images allow to compute longer brush strokes. Finer brushes use a less blurred version of the input and are layered on top of each other until the image is sufficiently approximated. This requires a vast amount of strokes and colors to be used during the painting process, which is impracticable for realization in hardware due to material and time constraints.

In contrast to this procedure, artists decompose an image into sharp regions that can include background, far objects, foreground, and sometimes regions within them as well. For each region they apply different painting strategies or even tools (cf. [KKM09]). Each region by itself is filled using coarse to fine layers. In this case over-painting is minimized since artists care about the required painting effort and the fact that the desired colors might interact in an unwanted way if over-painted too often.

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

DOI: 10.2312/vmv.20161350

delivered by EUROGRAPHICS DIGITAL LIBRARY www.eg.org diglib.eg.org

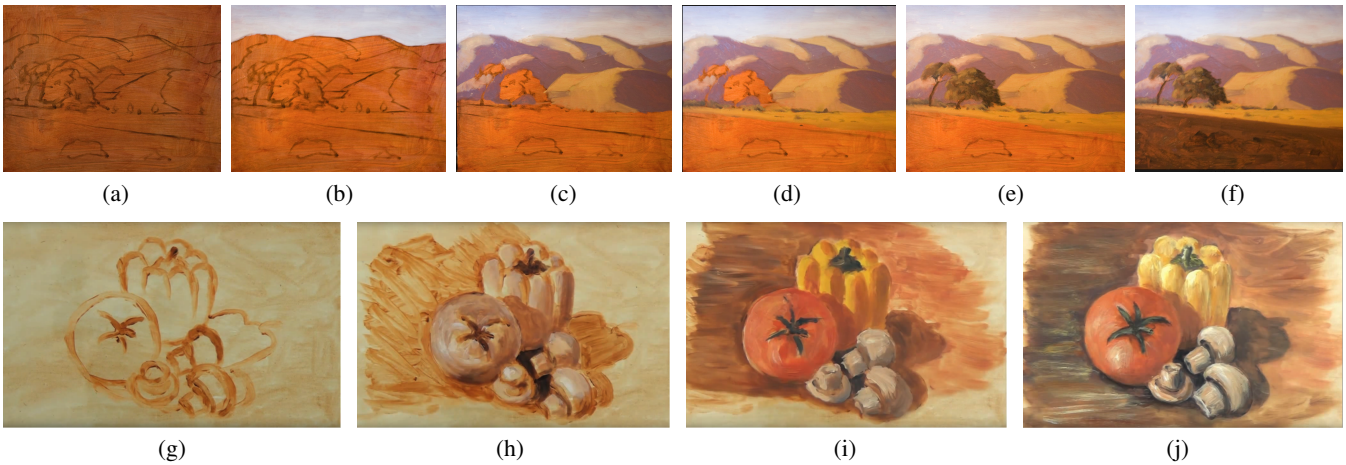


Figure 2: Examples of artists. Upper row: the composition into crisp regions in (a) is followed by painting from back to front in (b)-(f). Lower row: composition into foreground and background using outlines in (g), applying tonal values when translucent paints are used in (h), coloring each region in (i) and refinement with highlights and dark shadows in (j). (images courtesy: (a)-(f) Ross Boewns, paintdrawpint.com, (g)-(j) Kevin Hill).

Figure 2 illustrates this artistic process. In subfigure (a), crisp silhouette lines separate the various parts of the image: background (sky), a number of middle ground regions, and the foreground. Regions are typically painted from back to front (b)-(f), where the newly applied paint can over-paint the previous layers. In another example (g), lines separate the foreground from background and inner structures of objects. First, coarse tonal values are applied (h). This is done when translucent paints are to be applied later (alternatively, the artist can start by painting coarse features in color and add smaller features later). The next layer (i) adds the corresponding color information inside each region. Finally in (j), fine-scale highlights and dark shadows are added.

In this paper, we propose a painting strategy that mimics this kind of decomposition into regions and layers, directly inspired by artistic painting techniques. We apply our concept to painterly rendering of input photographs. For a given input image we use hierarchical segmentation into a number of clear regions. The regions are reorganised according to the painting order that is determined from their spatial arrangement as well as their content. Each region is separated into a number of layers that represent its content from coarse to fine details and can be painted with corresponding tool sizes. This way, blurring the input image is avoided even for larger virtual brushes and the brush strokes follow image content in a natural way.

To our knowledge, the importance of dividing an input image into regions and layers has not been addressed so far. The core message of our work is that only the combination of carefully selected regions and a hierarchical decomposition of layers within these regions can represent the painting process properly. Regions have to be used to determine composition and painting order (e.g. back to front), while layers have to be used to determine details and type of tools (coarse to fine). This combination of regions and layers allows creating high quality painterly renderings with much less colors and strokes. We demonstrate this with a number of examples and by comparing to existing techniques.

2. Related Work

Some aspects of our segmentation and layering strategy have been touched by prior works: Zeng et al. [ZZXZ09] interactively decompose an input photography into regions representing semantic or material classes. They obtain brush stroke examples (textures) for each of these classes from artists and use these to represent regions. However, they do not use a layer mechanism to separate coarse and fine features. Zhao et al. [ZZ10, ZZ11] extended this approach by incorporating additional stroke relationships such as neighborhood contrast and the deformation of regions/objects for achieving certain forms of abstraction.

Painterly Rendering: Hertzmann [Her03] and later Kyprianidis et al. [KCWI13] give an overview of painterly rendering and stroke-based approaches. An inspiration for non-photorealistic rendering techniques was always the somewhat algorithmic painting approach by Bob Ross, which was later directly implemented by Kalaidjian et al. [KKM09] for creating a special Bob Ross representation of synthetic landscapes. Zang et al. [ZHL14] use image processing techniques to emphasize light-dark and also color contrasts. They propose a more artistic approach for image pre-processing that improves painterly rendering approaches. Semmo et al. [SLKD15] create painterly renderings with a reduced number of colors by quantizing the input image. However, this is only applicable to software rendering systems.

Painting Machines: One of the earliest painting machine was developed by Harold Cohen [Coh12]. His system, known as 'AARON' uses techniques of artificial intelligence to produce abstract art. Other machines were later developed by Frieder Nake [Wik12], Ben Grosser [Gro13], Holger Baer [Bae13], Pindar van Arman [Arm12] and Kelly and Marx [KM13]. Deussen et al. [DLPT12, LPD13, LMPD15] propose a painting robot that uses a robot arm and visual feedback to produce drawings and paintings after photographs. In 2016, Andrew Conru started the 'RobotArt' competition for universities [Con16], where each of the participating

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

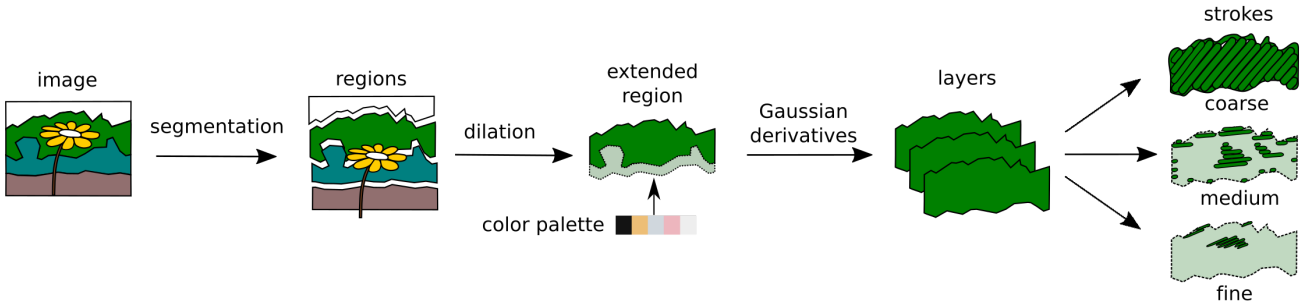


Figure 3: Overview of our processing pipeline: an input image is split into regions, the regions are extended using a dilation filter. A color palette can be added to each region or for the whole painting. Derivatives of Gaussians are used to extract layers from regions. Layers are filled with coarse, medium and fine details and painted with tools of corresponding size.

teams could submit an artwork created by their painting machine. Since then, the field of research gained more publicity and new projects were initiated.

Coarse to fine approaches: Hertzmann [Her98] creates coarse to fine layers by blurring the input images corresponding to the used brush sizes. Hays and Essa [HE04] improve this approach by masking out fine details/frequency at coarse levels. Earlier, Shiraishi and Yamaguchi [SY00] sorted strokes by the area they occupy and paint larger strokes before smaller strokes. Here, the color for strokes is sampled at the center of strokes, therefore no blurring of the input is needed, but the method needs much more strokes to converge. Lu et al. [LSF10] extract three layers (coarse, medium and fine) from images by thresholding the gradient magnitude of the input. While they do not provide artistic results they are still an important inspiration for our method.

Semantics-driven stylization: A number of papers analyze the image content manually or automatically and use this information for visual representation. Lindemeier et al. [LPD13] present drawing styles based on such semantics. The style, however, is assigned by the user and segmentation is done manually. In Lindemeier et al. [LMPD15], style parameters for certain objects are predefined, and erosion filters are applied to regions to avoid boundary artefacts. Some works propose a parse tree of separated objects to enable back to front painting [ZZXZ09, ZZ10, ZZ11]. Segmentation, however, is done manually. During rendering, brush strokes are chosen from a predefined stroke library according to the currently painted object. In [ZZ11] additional style constraints and parameters are proposed to change the contrasts of neighboring strokes. All above approaches divide the input only into neighboring regions but do not have a hierarchical layer structure.

Hierarchical Segmentation: DeCarlo and Santella [DS02] present an interactive abstraction system where images are segmented into a scale-space (Gaussian pyramid) and then abstracted. Eye-tracking is used to determine the importance of segments and to control the degree of abstraction. To our knowledge this is the only hierarchical segmentation approach that comes close to our target. In contrast to their work, we try to find segments automatically that represent the essential information of the input. Song et al. [SPL*13] perform a hierarchical image segmentation and replace segments with certain abstract shapes such as circles, squares and triangles. They rely on methods proposed in [AMFM09, SAH*10] and are able to create abstractions of the input. Other works [ZZXZ09, ZZ10, ZZ11, LMPD15] all use an interactive iterative background-foreground separation. However, there is no segmentation that separates basic low frequency content from high frequency details.

3. Overview

An overview of our method is given in Figure 3. The input is separated into regions, which are extended to allow a proper overpainting. Next, these regions are divided into layers that represent coarse, medium and fine details. We assign a painting style to each region that includes a color palette, the functions for orienting strokes within layers, and the intended stroke parameters. To create brush strokes within layers we use the method first proposed by Hertzmann [Her98] and later refined by Lindemeier et al. [LMPD15].

To decompose images into regions we use the image segmentation proposed by Arbelaez at al. [AMFM09]. Since painterly rendering is performed from back to front, there is a need to order the regions accordingly. Automatic determination of background and foreground for general images is a challenging problem in image analysis, and thus we apply some simple heuristics (top down for landscapes, outside to inside for other subjects), as well as background subtraction and a subsequent manual correction step (see Section 5.2). Given the image segmentation into regions, we extend them slightly towards the neighboring regions using a dilation filter. The missing information in the overlapping parts is created by color diffusion. This way subsequent regions always overpaint existing regions, and visible borders between them are avoided.

Lastly, each region is separated into a number of structure layers. We detect structures of different scales and represent the region by three layers using an extension of the method of Lu et al. [LSF10]. Regions are then realized by painting the layers from coarse to fine. Most of the steps in our processing pipelines are automatic. Artistic choices, such as the selection of a color palette for the painting and style parameters for the brush strokes within regions and layers, are done manually. Sometimes the automatic segmentation does not provide the correct number of regions or ordering between them – this has to be updated manually. In addition, if the user

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

wants to reduce details in some of the layers, he can do so in the style definition. This is an artistic decision that cannot be handled automatically.

In Section 4 we described the decomposition of images into regions and layers. The painting algorithm that uses the decomposed information to paint images from back to front and from coarse to fine is discussed in Section 5.

4. Decomposition of the Input

Our overall decomposition of the image is a two-fold process into first regions and then layers, as mentioned above.

4.1. Division into Regions

A vast amount of works have been published to decompose images in a meaningful way into a set of regions. We use the method of Arbelaez et al. [AMFM09] to divide the image into hierarchical segments, which are stored in the so called *ultrametric contour map (UCM)*. Higher hierarchy levels divide an image into larger regions, while lower levels further subdivide regions. We interactively divide the image into a number of meaningful regions, typically ranging from two to seven (see Figure 4).

4.2. Division into Layers

After defining regions, each region is separated into three layers which store coarse, medium, and fine details. The separation into three structural layers is inspired by the approach of Lu et al. [LSF10] that uses the first derivative of a Gaussian, which is good for detecting edges but has difficulties with features of a given scale. We use a combination of the first and second order derivative of a Gaussian to avoid this problem and choose σ corresponding to the smallest brush radius used in the painting process. Each region of the hierarchical segmentation is filtered using Gaussian derivative kernels $\partial^2 G$ and ∂G. Filtering is done for six orientations:

$$M_i = \sqrt{(\partial G(\Theta_i)^2 + (\partial^2 G(\Theta_i)^2} \qquad \Theta_i = \frac{i\pi}{6}, i = 0, ..., 5.$$

The combined response is the sum of the individual responses M_i and is then separated into three layers using two thresholds. Typical values are $t_m = 0.3$ for medium and $t_f = 0.9$ for fine details. Other values can be defined for different painting styles since they define an intended level of abstraction. The resulting layer masks determine the region that is to be painted with corresponding brush sizes.

5. Creating a Painterly Representation

Our painting algorithm is able to handle continuous colors obtained directly from the source image, as usually done in software-based NPR approaches [Her98, HE04, ZZXZ09, LSF10, ZZ10, ZZ11]. We can, however, also use a limited set of colors (usually about 4-10), which is needed when realizing paintings using a technical system such as e-David [DLPT12, LPD13, LMPD15] due to its hardware limitations. It consists of an industry robot, color repository, tool slots, cleaning station, and a camera for visual feedback (see Figure 10 (a)). Colors are picked from the color repository and cannot be mixed on a palette. This way, only overpainting can be used to create color nuances.

The developers of e-David provide a simple interface that enables access to the functionality of e-David. The usual procedure is to fill desired paints and brushes in the color repository and to mount and register a canvas to the system. The whole painting process can then be controlled by the network-based interface. Pictures of the canvas can be retrieved and are analyzed by our method as described in Section 5.4 to generate brush strokes. These stroke commands are then sent to the machine and realized by it on the canvas. In Figure 10 (b) we show one result created by e-David using our painting method and only four colors.

5.1. Color Palette

For creating color palettes we use the method of [CFL*15]. Their approach automatically creates a palette of a given size from an input image, but also provides an interactive tool to modify this color palette by the user.

5.2. Painting Order

Hierarchical image segmentation creates a hierarchy that is based on decreasing edge contrasts, i.e. the highest split between regions is along the contours with the largest response to contour detections. This kind of division, however, is not the order in which regions should be painted, and it is required to reorder them. Dividing images into background and foreground is a common problem in image processing [BPHV14]. Therefore, there is no general automatic solution for the required re-ordering. We order landscape images top to bottom, since they typically show a vertically stacked order of regions. We use outside-in ordering for all other cases. This initial guess can be refined using manual adjustment. For example, some objects such as the flower in Figure 4 (a) are divided correctly by outside-in ordering, but the castle in Figure 4 (b) would be selected as a second region in order since its content extrudes into the first vertical region on the top of the image.

(a) (b)

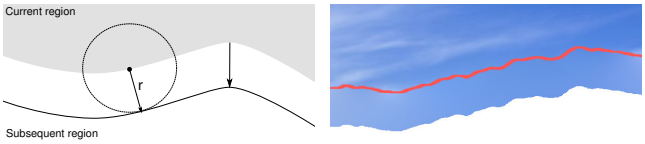
Figure 4: Two examples of region ordering: a) outside-in ordering; b) landscape scene without a clear vertical order.

5.3. Painting Regions

If painted from back to front, content of regions that are painted later have to be placed upon already painted parts of the image. If hard region borders are used, visible gaps between regions can appear. To alleviate this, we extend regions from the back towards

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

subsequent regions. This is a common practice for artists as well. For instance, the sky is usually painted larger than it will appear in the final painting, and is overpainted later with image content such as mountains to create a sharp contrast at the boundary. For extending regions we use a technique proposed by Lindemeier et al. [LMPD15]: a dilation filter is applied to a region and pixel values are copied from the border of the region into the new area using color diffusion described by Orzan et al. [OBW*08]. Since most of the region will be overpainted by subsequent regions, there is no need for a complex texture transfer. Figure 5 illustrates this dilation.



(a) (b)

Figure 5: Dilation of a region: (a) extension of the region towards the subsequent region; (b) filling missing image information by copying pixels from the region boundary.

For a better differentiation of regions we have to create sharp borders to already painted regions at the current outline of the region. Previous works, e.g. [ZZXZ09], halucinate this differentiation by setting the transparency of pixels of strokes that cross region borders to zero and thus blend colors.

For artistically-looking results especially with using painting machines, however, proper outlines are essential. We extract such outlines by morphological operations on the masks of the current and previously painted regions and paint them with the largest brush first. In Figure 6 we compare a boundary region from [ZZXZ09] with ours. The images on the left and in the middle show that the opacity of strokes across boundaries is reduced pixel-wise for coarser (left) and finer (center) strokes which creates blurred boundaries. In contrast to that, our results creates sharp boundaries by painting along the border.

5.4. Painting Layers

Due to the filtering process in the layer extraction step, the coarse and medium layers may have holes where only subsequent finer layers have content. These missing parts are filled by using the diffusion operator as described by Orzan et al. [OBW*08]. This information is later used for painting the individual layers. Results of this layering process for the petal in Figure 4 (a) are shown in Figure 7. The three layers coarse (a) medium (b) and fine (c) are shown on top. On the bottom the same layers are shown after color diffusion (d) - (f).

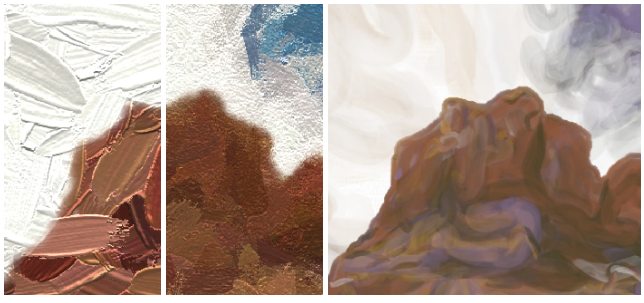
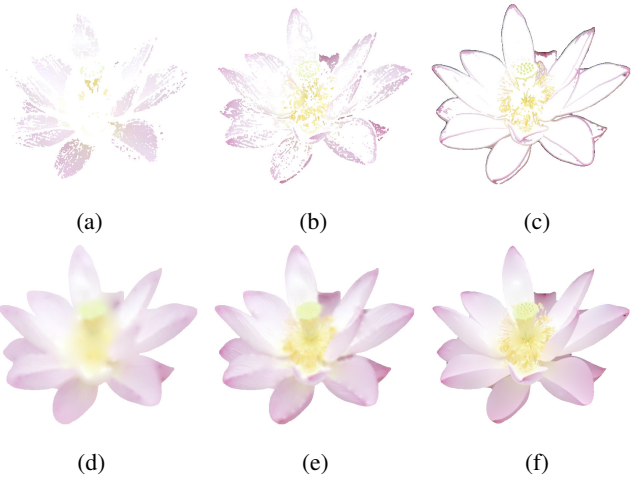


Figure 6: Left and middle: [ZZ10] reduce the opacity of strokes that cross region borders at different scales, this is not possible with real strokes. Right: our result with sharp boundaries.



(a) (b) (c)

(d) (e) (f)

Figure 7: Extracted structure layers from the petal in Figure 4 (a). (a) coarse layer; (b) medium layer; (c) fine layer. (d)-(f): layers after color diffusion.

For each region the layers are painted from coarse to fine. Filling a layer with strokes is controlled by a feedback loop. At the beginning of every iteration we compute a color difference map between canvas C and input image S for all pixels of the current layer. The color difference is calculated in perceptually linear CIELab space and shows where additional brush strokes have to be created. This is done until the layer is represented sufficiently well by strokes. However, since we cannot produce all details with a large brush, we cannot use the overall color distance to determine if we have represented the layer well enough. Instead, we remove details in the color difference map smaller than the current brush size with a morphological opening operation. If the mean color difference is below a given threshold, nothing can be improved with the current brush and we have to move on to the next layer.

Strokes are created by the following process: Potential positions for new strokes are found using the layer masks and color difference map. Strokes are sampled along the layer by grid sampling [Her98], rejection sampling [ZZ11], or by Lloyd Relaxation [LMPD15]. Orientation of strokes is determined by a tensor field computed from the image content. Additionally, relaxation and smoothing is used to reduce noise in the direction field [KK11]. For every seed point from our chosen sampling strategy we set the used brush color directly to the color extracted from the input image, if a continuous color palette is used, or to the closest match in a color palette if a limited set of colors is used (see Section 5.1). The stroke is then integrated, until one of the following conditions is met: the color difference between input and canvas is already small enough, the

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

stroke reaches a maximum length (given by the painting style), the border of a layer is reached, or the brush color applied to the canvas would result in a larger color difference. The effect of applied color is computed using the Kubelka-Munk diffuse reflectance model, cf. [CAS*97, BWL04].

We can stylize each layer by defining various parameters that we define as a painting style for our pipeline. Such parameters include brush radius, stroke length, curvature, as well as desired color palette and opacity (see Section 5.1). The stylization can be further modified by two main characteristics: a sampling strategy which creates stroke seed points, as well as an orientation field that guides the strokes. Sampling strategies employed throughout this paper include random sampling, grid sampling [Her98], density-based sampling [ZZ11], and Lloyd-relaxation [LMPD15]. Orientation fields can be defined by image gradients, from region boundaries, as well as from user-defined patterns. Figure 8 shows the usage of various painting styles defined by the user for each segment.

6. Results

Our painting algorithm can be used with every stroke-based rendering system. We will show a number of results using such systems; please note, however, that the main goal of our approach is to help painting machines such as e-David to create artistically looking paintings.

6.1. Software-based Rendering

In order to show rendered results we implemented a simple renderer based on existing methods. We collected 300 texture samples from real brush strokes for five different brushes, similar to what was done by Zeng et al. [ZZXZ09]. We render the strokes by creating textured polygons along the stroke path similar to Strassman [Str86]. The composition of paint is computed following the approach by Baxter et al. [BWL04]. We simplified their model by using RGB channels instead of the proposed seven channels.

In Figure 9 (a) we create a painterly rendering using the Hertzmann algorithm [Her98]. We divided the input image into three regions and processed each region independently (similar to [ZZXZ09]). In (b) the result of our method is shown that divides the input into regions *and* layers. Our strategy reduces the number of required strokes for the input by fifty percent and the required colors to one third: (a) 47201 strokes, 23909 colors, (b) 20107 strokes, 8065 colors. The time to generate the stroke and the resulting rendering was reduced from 380 to 150 seconds. Please note, that the division into regions does not save many strokes, it is the layering mechanism within the regions that saves most strokes. Due to the additive behaviour of our painting method within layers, strokes are placed only at positions where they are actually necessary, which minimizes over-painting.

6.2. Hardware-based Rendering

Since our method is based on a visual feedback loop and usable with a reduced set of colors, it can be directly applied to a painting machine, such as e-David (see Section 5). A result created with the e-David system can be seen in Figure 10 (b). We used acrylics and mixed a color palette of four colors with lamp black, primary cyan, raw umber, vandyke brown, sand, titanium white, lilac and cadmium red. We thinned out the paints by adding medium.

7. Conclusion

The main point of this paper is to highlight the importance of decomposing an image into *regions* and *layers* for painterly rendering. Such a decomposition allows to create artistically looking results with much less strokes and colors and furthermore drastically enhances artistic freedom. While reducing paint strokes might not be very important for software-only approaches, it is for physical painting using machines such as e-David [LMPD15] or Zanelle [Arm12]. With these realizations the number of needed colors is crucial since only a limited amount of different paints can be applied.

We provide a decomposition pipeline that allows artists as well as untrained users to efficiently decompose an input image into regions and separate these further into layers. Using different degrees of detailing, style parameters such as stroke distributions, and orientations for regions and layers allows to create a variety of different artistic renditions.

We demonstrated the effectiveness of our method by showing a number of painterly results and compared our approach to previous methods. For many inputs we create similarly looking results with only 30% of the strokes and used colors.

In the future we would be interested to extend our framework to include form abstractions (such as shown in [SPL*13]), as well as other abstract rendering methods such as optical blending with brush strokes.

References

[AMFM09] ARBELAEZ P., MAIRE M., FOWLKES C., MALIK J.: From contours to regions: An empirical evaluation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* (June 2009), pp. 2294–2301. doi:10.1109/CVPR.2009.5206707. 3, 4

[Arm12] ARMAN P. V.: Zanelle. http://www.vanarman.com/, 2012. March 13th, 2012. 1, 2, 6

[Bae13] BAER H.:. http://www.holgerbaer.com/, 2013. Januar 3rd, 2013. 1, 2

[BPHV14] BOUWMANS T., PORIKLI F., HÖFERLIN B., VACAVANT A.: *Background Modeling and Foreground Detection for Video Surveillance*. CRC Press, 2014. 4

[BWL04] BAXTER W., WENDT J., LIN M. C.: Impasto: A realistic, interactive model for paint. In *Proceedings of the 3rd International Symposium on Non-photorealistic Animation and Rendering* (New York, NY, USA, 2004), NPAR '04, ACM, pp. 45–148. doi:10.1145/987657.987665. 6

[CAS*97] CURTIS C. J., ANDERSON S. E., SEIMS J. E., FLEISCHER K. W., SALESIN D. H.: Computer-generated watercolor. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., pp. 421–430. doi:10.1145/258734.258896. 6

[CFL*15] CHANG H., FRIED O., LIU Y., DIVERDI S., FINKELSTEIN A.: Palette-based photo recoloring. *ACM Transactions on Graphics (Proc. SIGGRAPH) 34*, 4 (July 2015). 4

[Coh12] COHEN H.: Aaron. http://crca.ucsd.edu/ hcohen/, 2012. March 13th, 2012. 1, 2

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

(a) (b)

Figure 8: Variation of style parameters: (a) abstract foreground with many low curvature strokes. (b) only coarsest layer used for the background.

(a) (b)

Figure 9: Comparison of painterly rendering approaches. (a) traditional painterly rendering using three segments and blurring, (b) our result with regions, layers and no blur. Similar visual quality can be created with only one third of colors and less than half of the strokes (numbers see text).

[Con16] CONRU A.: Robotart - create something beautiful, 2016. URL: http://robotart.org/. 2

[DLPT12] DEUSSEN O., LINDEMEIER T., PIRK S., TAUTZENBERGER M.: Feedback-guided stroke placement for a painting machine. In *Proceedings of the Eighth Annual Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging* (Aire-la-Ville, Switzerland, Switzerland, 2012), CAe '12, Eurographics Association, pp. 25–33. 1, 2, 4

[DS02] DECARLO D., SANTELLA A.: Stylization and abstraction of photographs. *ACM Trans. Graph. 21*, 3 (July 2002), 769–776. doi:10.1145/566654.566650. 3

[Gro13] GROSSER B.:. http://bengrosser.com/, 2013. January 13th, 2013. 1, 2

[HE04] HAYS J., ESSA I.: Image and video based painterly animation. In *Proceedings of the 3rd International Symposium on Non-photorealistic Animation and Rendering* (New York, NY, USA, 2004), NPAR '04, ACM, pp. 113–120. doi:10.1145/987657.987676. 1, 3, 4

[Her98] HERTZMANN A.: Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1998), SIGGRAPH '98, ACM, pp. 453–460. doi:10.1145/280814.280951. 1, 3, 4, 5, 6

[Her03] HERTZMANN A.: A survey of stroke-based rendering. *IEEE Computer Graphics and Applications*, 4 (2003), 70–81. 2

[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 Transactions on Visualization and Computer Graphics 19*, 5 (May 2013), 866–885. doi:10.1109/TVCG.2012.160. 2

[KK11] KYPRIANIDIS J. E., KANG H.: Image and Video Abstraction by Coherence-Enhancing Filtering. *Computer Graphics Forum 30*, 2 (apr 2011), 593–602. doi:10.1111/j.1467-8659.2011.01882.x. 5

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

(a) (b)

Figure 10: (a) *e-David* painting machine. (b) Result with four colors and 15703 strokes created with our method and *e-David* in 15 hours.

[KKM09] KALAIDJIAN A., KAPLAN C. S., MANN S.: Automated landscape painting in the style of bob ross. In *Proceedings of the Fifth Eurographics Conference on Computational Aesthetics in Graphics, Visualization and Imaging* (Aire-la-Ville, Switzerland, Switzerland, 2009), Computational Aesthetics'09, Eurographics Association, pp. 115–122. doi:10.2312/COMPAESTH/COMPAESTH09/115-122. 1, 2

[KM13] KELLY L., MARX D.: Vangobot. http://vangobot.com, 2013. January 10th, 2013. 1, 2

[LMPD15] LINDEMEIER T., METZNER J., POLLAK L., DEUSSEN O.: Hardware-based non-photorealistic rendering using a painting robot. *Computer Graphics Forum 34*, 2 (2015), 311–323. doi:10.1111/cgf.12562. 2, 3, 4, 5, 6

[LPD13] LINDEMEIER T., PIRK S., DEUSSEN O.: Image stylization with a painting machine using semantic hints. *Computers & Graphics 37*, 5 (aug 2013), 293–301. doi:10.1016/j.cag.2013.01.005. 2, 3, 4

[LSF10] LU J., SANDER P. V., FINKELSTEIN A.: Interactive painterly stylization of images, videos and 3d animations. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2010), I3D '10, ACM, pp. 127–134. doi:10.1145/1730804.1730825. 3, 4

[OBW*08] ORZAN A., BOUSSEAU A., WINNEMÖLLER H., BARLA P., THOLLOT J., SALESIN D.: Diffusion curves: A vector representation for smooth-shaded images. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 92:1–92:8. doi:10.1145/1360612.1360691. 5

[SAH*10] SONG Y.-Z., ARBELAEZ P., HALL P., LI C., BALIKAI A.: Finding semantic structures in image hierarchies using laplacian graph energy. In *Computer Vision–ECCV 2010*. Springer, 2010, pp. 694–707. 3

[SLKD15] SEMMO A., LIMBERGER D., KYPRIANIDIS J. E., DÖLLNER J.: Image stylization by oil paint filtering using color palettes. In *Proceedings of the Workshop on Computational Aesthetics* (Aire-la-Ville, Switzerland, Switzerland, 2015), CAE '15, Eurographics Association, pp. 149–158. 2

[SPL*13] SONG Y.-Z., PICKUP D., LI C., ROSIN P., HALL P.: Abstract art by shape classification. *Visualization and Computer Graphics, IEEE Transactions on 19*, 8 (Aug 2013), 1252–1263. doi:10.1109/TVCG.2013.13. 3, 6

[Str86] STRASSMANN S.: Hairy brushes. *SIGGRAPH Comput. Graph. 20*, 4 (Aug. 1986), 225–232. doi:10.1145/15886.15911. 6

[SY00] SHIRAISHI M., YAMAGUCHI Y.: An algorithm for automatic painterly rendering based on local source image approximation. In *Proceedings of the 1st International Symposium on Non-photorealistic Animation and Rendering* (New York, NY, USA, 2000), NPAR '00, ACM, pp. 53–58. doi:10.1145/340916.340923. 3

[TFL12] TRESSET P. A., FOL LEYMARIE F.: Sketches by paul the robot. In *Proceedings of the Eighth Annual Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging* (Aire-la-Ville, Switzerland, Switzerland, 2012), CAe '12, Eurographics Association, pp. 17–24. 1

[Wik12] WIKIPEDIA: Frieder nake. http://de.wikipedia.org/wiki/Frieder_Nake, 2012. December 19th, 2012. 1, 2

[ZHL14] ZANG Y., HUANG H., LI C.-F.: Artistic preprocessing for painterly rendering and image stylization. *Vis. Comput. 30*, 9 (Sept. 2014), 969–979. doi:10.1007/s00371-013-0881-6. 2

[ZZ10] ZHAO M., ZHU S.-C.: Sisley the abstract painter. In *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2010), NPAR '10, ACM, pp. 99–107. doi:10.1145/1809939.1809951. 2, 3, 4, 5

[ZZ11] ZHAO M., ZHU S.-C.: Customizing painterly rendering styles using stroke processes. In *NPAR '11: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2011), ACM, pp. 137–146. doi:http://doi.acm.org/10.1145/2024676.2024698. 2, 3, 4, 5, 6

[ZZXZ09] ZENG K., ZHAO M., XIONG C., ZHU S.-C.: From image parsing to painterly rendering. *ACM Trans. Graph. 29*, 1 (Dec. 2009), 2:1–2:11. doi:10.1145/1640443.1640445. 2, 3, 4, 5, 6

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.