Image and Video Based Painterly Animation

James Hays
School of Computer Science
Carnegie Mellon University

Irfan Essa
GVU Center / College of Computing
Georgia Institute of Technology

http://www.cc.gatech.edu/cpl/projects/artstyling/

Figure 1: Various painterly renderings of a pink flower (top left). Painterly renders representing (top row) "watercolor", "Van Gogh", "Impressionism", (bottom row) "Abstract", "Pointillism", "Flower" and "Abstract" styles. Figure 3 shows some of the brush strokes used.

Abstract

We present techniques for transforming images and videos into painterly animations depicting different artistic styles. Our techniques rely on image and video analysis to compute appearance and motion properties. We also determine and apply motion information from different (user-specified) sources to static and moving images. These properties that encode spatio-temporal variations are then used to render (or paint) effects of selected styles to generate images and videos with a painted look. Painterly animations are generated using a mesh of *brush stroke* objects with dynamic spatio-temporal properties. *Styles* govern the behavior of these brush strokes as well as their rendering to a virtual canvas. We present methods for modifying the properties of these brush strokes according to the input images, videos, or motions. Brush stroke color, length, orientation, opacity, and *motion* are determined and the brush strokes are regenerated to fill the canvas as the video changes. All brush stroke properties are temporally constrained to guarantee temporally coherent non-photorealistic animations.

1 Expressive Painterly Animation

The computer graphics community, in addition to concentrating on photorealistic rendering, strives for automatic methods to generate non-photorealistic and "expressive" renderings [Lansdown and Schofield 1995]. The need for non-photorealistic rendering and animation is obvious to anyone who has marvelled at artistic works where complex scenes and objects are rendered using pen and brush strokes, using lines, colors, and etches. In this paper, we present an automatic image-based rendering approach to generate expressive, artistic, and painterly animations. Our method enables the synthesis of painterly animations by using video analysis to manipulate spatio-temporally coherent brush strokes. Our method also supports merging motions from different sources to generate moving images from static pictures.

To undertake automatic non-photorealistic rendering of moving images, we need to take into account both the spatial (across each frame) and temporal (from one frame to another) information. Towards this end, we propose a framework for generating and manipulating a mesh of dynamic brush strokes by analyzing individual frames as well as motion between frames. Each of these brush strokes has properties such as opacity, color, length, width, orientation, and motion (see Figure 2). We also employ a higher-level concept of *style* to control the behavior of brush strokes and thus produce a spectrum of painterly styles.

In our work we borrow from earlier contributions by Litwinowicz [1997] and Hertzmann et al. [2000]. We introduce several techniques that allow for significantly improved results in transforming video into painterly animations. (1) Each brush stroke property is constrained over time to ensure that smooth, temporally constrained animations are produced. Brush stroke generation and deletion are

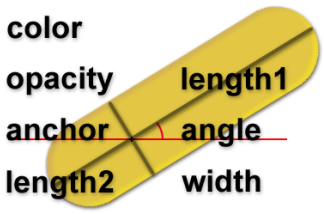


Figure 2: Each Brush stroke is an object with the following properties: Anchor point in image coordinates, angle of orientation in degrees, width in pixels, lengths in both directions from the anchor point in pixels, color (R, G, B) for the current and past several frames, and opacity. In addition the brush stroke knows if it is "New" and/or if it is "Strong." Anchor, angle, width, and lengths are kept in floating point coordinates.

performed smoothly through time by modifying brush stroke opacity. By adding these temporal constraints to the spatial constraints discussed in previous work we create non-photorealistic techniques that animate elegantly across multiple frames without noticeable artifacts. (2) We employ radial basis functions (RBFs) to globally orient brush strokes across time and space. (3) We use edge detection at varying frequencies to guide the creation of new brush strokes and the refinement of fine details. (4) We improve rendering quality beyond previous works by decoupling output resolution from input dimensions and by using real brush stroke textures along with a simple lighting model. (5) Finally, we emphasize the artistic versitility of motion by synthesizing motion information for still images to produce animated stills as well as transplanting motion from video segments onto stills. Our methods are robust, allowing us to generate videos and images of many different user selected artistic styles.

2 Related Work

We are primarily interested in generating videos that show various forms of non-photorealistic moving images. Our work relies on our ability to analyze images and videos to modify the properties of our brush strokes, which are then rendered (painted) to generate a new image or video. Along these lines, our work benefits from all the earlier efforts aimed at using brush strokes as the key elements for rendering style (see the web site maintained by Craig Reynolds [WWW] for an elaborate overview of stylistic depiction in computer graphics).

Our work is closely related to some very important first steps in analyzing imagery to generate non-photorealistic rendering. Early work in this area is that of Litwinowicz [1997], who used image and video analysis to generate Impressionistic effects. This system allows a user to select brush stroke size and style and have the program automatically process a segment of video. Optical flow and edge detection are used manipulate a mesh of persistent brush strokes. Additionally, the user can choose to have the brush strokes follow the contours of an image or fix themselves to a global angle. Advanced versions of this approach, with significant manual intervention, have been used in painterly effects in the feature film "What Dreams May Come" [Ward 1998]. Our approach significantly adds to this approach by more thoroughly addressing temporal coherence.

Hertzmann [1998] introduced a method to automatically paint still images with various stroke sizes and shapes and then extended the technique to video [2000]. This work tries to address the problem of temporal incoherence in Litwinowicz [1997] by only updating the properties of brush strokes that lie in video regions that show

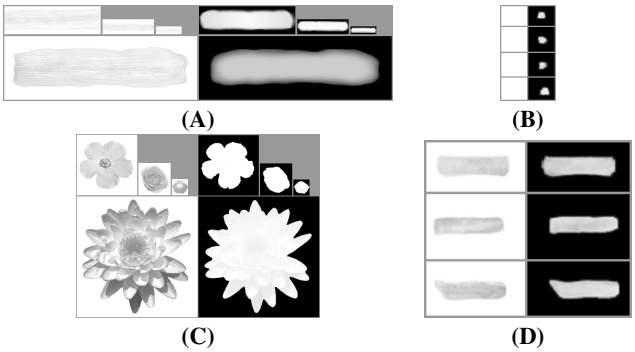


Figure 3: Pairs of brush stroke textures and alpha masks. (A) Four brush strokes used for four different layers of impressionism. (B) Four brush strokes used for pointillism. (C) Four brush strokes for the "flower" style. (D) Three brush strokes used for Van Gogh.

significant change. However, flickering and scintillation of brush strokes remains a problem. A comparison between this technique and our own is contained within the accompanying video. The notion of synthesizing flow fields for painterly animation is touched upon in this work and we build on that.

DeCarlo and Santella [2002] presented an algorithm to stylize photographs by running inputs through (well-known) image preprocessing algorithms to segment the image into regions that could be given a fixed color. Our work differs from this effort as we are more concentrated on video and specifically on brush strokes, while their approach is perhaps the state-of-the-art in shaded styles on still images.

Klein et al. [2002] present a tool to aid in generating non-photorealistic animations from video. Similar to our approach, this tool undertakes a spatio-temporal analysis of video. A novel aspect of their work is the use of a set of "rendering solids" where each rendering solid is a function defined over an interval of time. This allows for the effective and interactive rendering of NPR styles. We believe our approach, inspired largely by the actual process of painting, produces renderings more faithful to the historical art styles we seek to imitate.

Hertzman [2001] presented a versatile technique to learn non-photorealistic transformations based on pairs of unpainted and painted example images. These learned transformations can then be applied to new inputs. This has been extended to video processing only recently [Haro and Essa 2002]. These learning methods hold great promise and can generalize to solve many problems. However, we do not believe they capture certain painterly styles as faithfully as our more domain-specific approach. Our approach also allows more user control of rendering styles.

Recently Hertzmann [2002] introduced a new method for adding a physical appearance to brush strokes. The basic idea is to add height fields to brush strokes to allow for lighting calculations. The resulting highlights and shading give the paint strokes a more realistic appearance. We have adopted a similar approach in our rendering technique.

Finally, our work allows us to extract motion information from one source and apply it to another image (or video) to show moving images. In some instances, we rely on the work of van Wijk [2002] to synthesize flow information and apply it to images and videos. This allows non-photorealistic effects to show motion along the lines of Motion Without Movement. [Freeman et al. 1991].

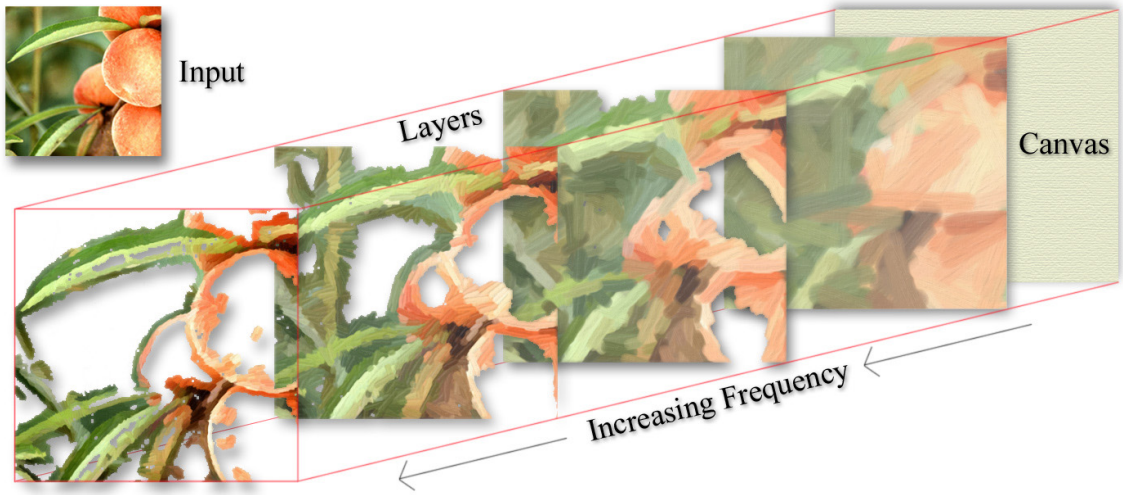


Figure 4: Brush stroke layers and the canvas. The canvas here is twice the size of input image. The Layers are independent meshes of brush strokes which successively refine the input frame.

3 Brush strokes, Styles, Layers, & Canvas

In order to produce a painterly animation one must first pick a *style*. Here a style is an encapsulation of parameters that control the analysis of input frames, the behavior of brush strokes, and the rendering of output. We extend the concept of *style* from [Hertzmann 1998] by adding several parameters shown in Table 1. Styles and the brush stroke textures that accompany them are created by a user and can be saved for reuse.

In order to emulate the coarse to fine painting process customary to styles like Impressionism, we extend the concept of *Layers* (introduced in [Hertzmann 1998]). Layers are disjoint groups of brush strokes representing successive passes of refinement in a painting. As used here, brush strokes are created within a Layer only when perceptually necessary- as determined by the presence of edges at the frequency band corresponding to each Layer.

Each brush stroke has several dynamic properties that are influenced by the analysis of input frames (see Figure 2). Our brush stroke definition most closely resembles that of Litwinowicz [1997]. However, we take special care in modifying brush stroke properties so as to maintain temporal coherency. Previous techniques left brush stroke properties largely unconstrained between frames and produced animations with scintillation and flickering. We add a new brush stroke property, *opacity*, which intuitively corresponds to the confidence a painter would have in creating a refinement stroke. Giving brush strokes dynamic opacity also allows us to smoothly create and delete brush strokes.

Output is rendered onto an arbitrarily large *canvas*. Using a canvas of higher resolution than an input frame is analogous to a painter filling an entire canvas according to a small reference photograph. It allows the output rendering to retain the structural coherency of the input frame while still displaying considerable artistic styling. Painting at the same resolution as your input, as done in previous works, has the effect of blurring details as input pixels are summarized into larger brush strokes. The only way to faithfully reproduce fine input details would be to use degenerately small brush strokes which would add no painterly detail. Layers and canvas are shown in Figure 4.

Table 1: Relevant style parameters used to generate artistic renders of videos and images.

The following parameters are defined for each Layer

brush width, w_b Width of the brush stroke;

regeneration width, w_r The minimum radius of unoccupied space on the canvas for which a new brush stroke will be created to fill. Making it larger than the actual width of a brush stroke will allow the canvas to show through. Making it smaller will make the brush strokes appear more crowded;

brush texture, T_b Which set of brush texture and alpha mask to use in rendering;

derivative of a brush stroke's opacity over time, δ**Opacity** Effectively keeps a brush stroke from changing opacity too quickly. Typically about 10 percent of the range of opacity per frame;

derivative of a brush stroke's length over time, δ**Length** Typically 1 pixel per frame or less for small brush strokes;

derivative of a brush stroke's angle over time $\delta\theta$ Typically 1 degree per frame or less for large brush strokes;

The number of previous frames to average color over for each brush, $C_{s,\max}$ Larger values produce extremely smooth renders in which the colors seem to wash around as brush strokes move to regions of different colors. Smaller values are responsive to color change at the risk of being noisy. A value of about 5 typically works well;

Palette Reduction, P_r The depth of palette to use during color extraction. In some styles it is more appropriate to use a small palette of colors. Brush strokes will still blend together to produce intermediate shades during rendering;

Noise, v The amount of noise to add to the current frame for this layer. Useful for pointillism and Van Gogh.

The following parameters do not vary between layers

contrast stretch Whether or not to stretch the contrast of the output frame in order to utilize the full dynamic range of the color space. Typically set to true, as painters, especially the impressionist, are renowned for seeking to maximize color variation;

histogram equalize Whether or not to equalize each channel of the final rendering. This warps the colors of the output image, making the images look more abstract. It also maximizes the coverage of the color space;

median filter Whether or not to median filter the output frame. This causes the output to stylistically resemble a watercolor painting. Thus each style can have a watercolor variant

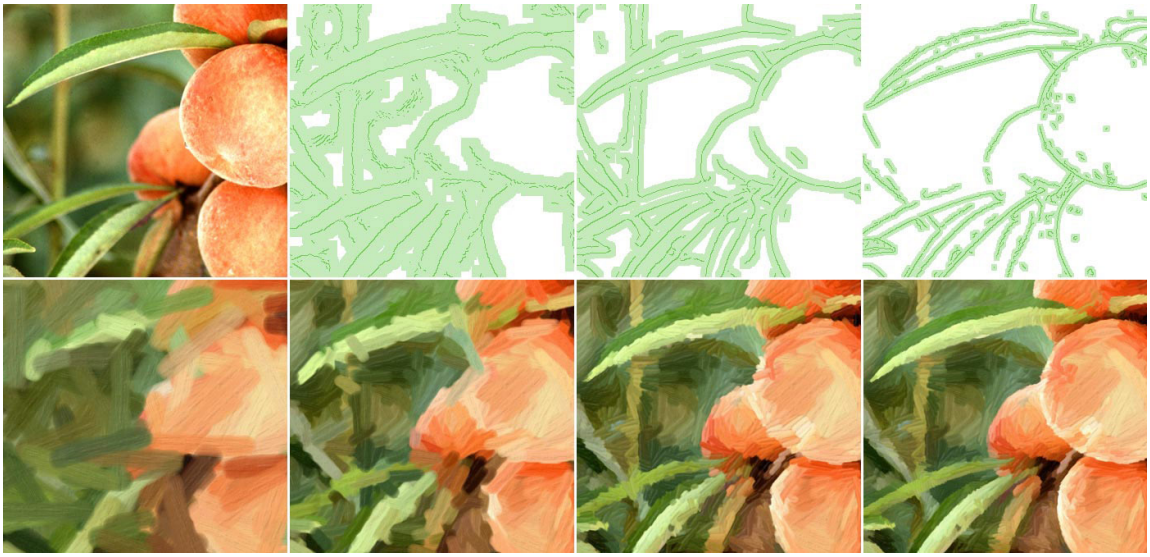


Figure 5: (top row, left to right) Original image, low frequency edges, mid frequency edges, high frequency edges. (bottom row, left to right) Base layer, first layer of refinement, second layer of refinement, third layer of refinement. The dark green in the edge images are the detected edges at each frequency. The light green border is the area, proportional to w_r, that brush strokes will be created to fill. Note how smoothly shaded and out of focus regions are less refined while remaining perceptually similar to the input.

4 Approach

Now we describe in some detail each step in our process to extract information from images and video and then render brush strokes on to a canvas in a painterly manner. The stages of processing that we describe next follow the the initial steps taken by the user to define or choose a style for a given video sequence. The outer loop of our process, containing steps B through F, runs for each input frame.

A. Initialization: First all *layers* are initialized. The number of layers will depend on the *style* selected. Each layer contains an (initially empty) array of brush strokes. The layers will be filled, as necessary, in the regeneration and refinement step.

B. Motion: We are interested in determining the movement, in image space, of each pixel from frame to frame. The goal of this is to give brush strokes motion such that they appear to be sticking to objects in the real world. If brush strokes are not moved from frame to frame in this manner an undesirable "stained-glass" effect results. This observation is a key in Litwinowicz [1997] and Meier [1996]. Following this lead, we also look upon the immense literature in computer vision on measuring pixel motion and use optical flow measurement to estimate the motion between the current and the previous frames. Where no previous frame exists (the first frame in any sequence), this step is omitted. We employed our implementation of Black and Anandan flow [1991]. For each brush stroke, the estimated motion vector at its location is added to its anchor point.

C. Regeneration and Refinement: In video sequences with motion, optical flow will often push the brush strokes far enough apart that the canvas would be visible between them. For this reason, we must search for and fill gaps in our mesh of brush strokes. Brush strokes in higher layers can also "fall off" the features they are meant to be refining because of imperfections in optical flow. In both these cases new brush strokes must be generated to refine the features and prevent the canvas from showing through.

It is also necessary to remove certain brush strokes. Brush strokes that have been pushed on top of each other are redundant and cause visual clutter. Brush strokes which have strayed from high-frequency features must also be removed. If unnecessary high-frequency brush strokes are not removed, the upper layers could eventually fill with brush strokes even if there are no high frequency features needing to be refined.

On the bottom layer, the canvas is searched and new brush strokes are created where gaps are found. Gaps are simply defined as areas w_r across that contain no brush stroke anchor point. The canvas is searched in pseudo-random order (versus scanline) in order to avoid certain mechanical looking artifacts that can result if brush strokes are regenerated in sequence and end up perfectly spaced. New brush strokes are added at the rear of the draw order to maintain temporal coherency. They are also marked as "new."

The same process is executed on the higher layers, but the regeneration is restricted to areas near edges of the appropriate frequency (see Sec. 4E). For example, brush strokes in the highest layer are placed on or near the highest frequency edges in the input frame. These brush strokes appear gradually. They are initially nearly transparent and gain δOpacity each frame they remain on top of an edge of appropriate frequency.

Within each layer, brush strokes centered on the same pixel as another brush stroke higher in the draw order have their opacity reduced by δOpacity, as they are largely redundant. Once their opacity reaches zero they are deleted. Brush strokes pushed off the canvas by optical flow are also removed. Lastly brush strokes which have moved away from edges of the appropriate frequency, either through imperfect optical flow or edges disappearing in the video, are gradually made more transparent until they are removed or they again approach an edge.

We believe proximity to edges is a more perceptually accurate description of how an artist might refine a painting than what previous techniques employed. Hertzmann [1998] creates brush strokes in higher layers when a rendered version of the lower layer differs too much in color from the original image. In many styles, painters worry more about refining the structural accuracy of an image rather than correcting color in otherwise empty regions. (see Figure 6)

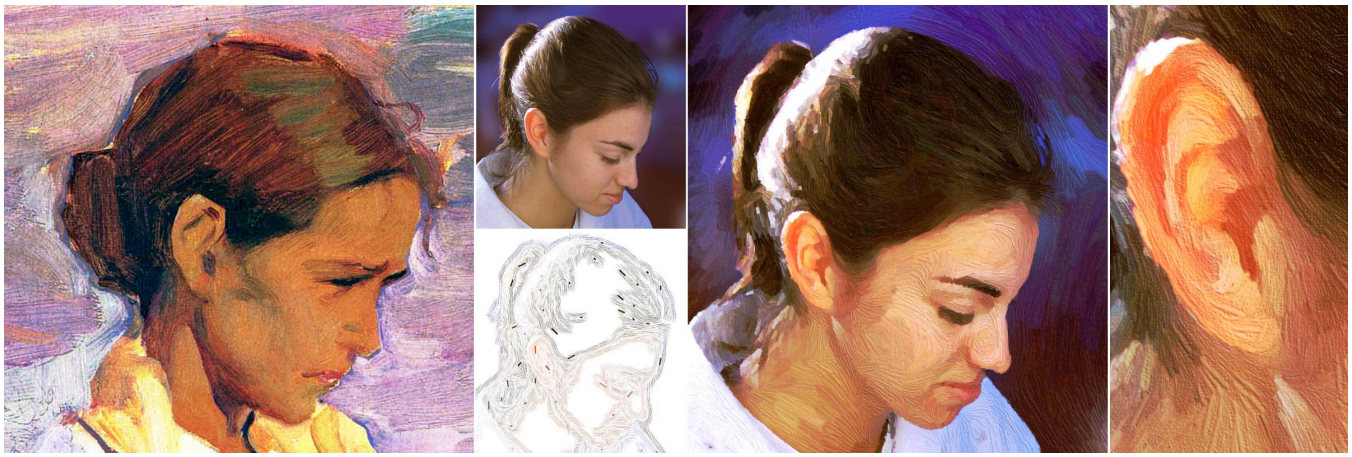


Figure 6: (Left) cropped portion of a painting by Spanish Impressionist Joaquin Sorolla Y Bastida. (Center Top) Input (Center Bottom) Visualization of the RBF. Brush strokes on the third layer are rendered thinly, except for the "Strong" strokes which are wide. (Center Right) Output. Notice how the smaller brush strokes wrap around the shape of the woman's head, as in the painting on the left. Also notice how large strokes are used for the background and fine strokes for the details. The color difference between the original photo and the render is the result of colors being equalized and noise being added during the color extraction step as dictated by our Impressionist (Far Right) Detail view.

D. Orientation: Painters typically paint by following contours of features in an image. Instead of orienting each brush stroke largely based on the local gradient estimates as suggested by [Hertzmann 1998; Litwinowicz 1997], we believe that brush stroke orientation should be globally dictated only by the brush strokes lying on the strongest gradients. We achieve this by using radial basis functions (RBFs) to globally interpolate X and Y gradients from only the strongest gradients [Bors 2001].

For each layer, the input image is blurred by a Gaussian kernel of width proportional to w_b. Gradients on that layer are then estimated across the entire frame with a Sobel filter [Forsyth and Ponce 2002]. "New" brush strokes that happen to fall on gradients of magnitude greater than a stylistically determined threshold and are not near other "Strong" brush strokes are themselves marked as "Strong." Once marked as "Strong," a brush stroke can not lose the designation. This is to avoid the potential temporal inconsistency that would be caused by suddenly removing a basis point from the RBF between frames.

These "Strong" brush strokes, typically numbering 10 to 200 per layer, become the basis points for a radial basis function. In all results shown a linear interpolant was used as we found it produced more aesthetically pleasing gradient estimations than thin-plate or Gaussian bases. Typically, all layers contribute "Strong" brush strokes to the same radial basis function in order to capture the gradients at different frequency bands. The radial basis function is then solved and evaluated at the anchor point of each non-"Strong" brush stroke. Each brush strokes angle is set as the arc tangent of the interpolated Y and X gradients plus 90 degrees.

Brush strokes eligible to become "Strong" are limited to "New" brush strokes to help maintain the distribution of basis points across the canvas. For example, in a video sequence with an upward panning camera, there would be a lack of basis points at the top of the frame because all of the existing basis points have been moved down by the optical flow step described earlier. Thus, the new brush strokes created in step C (at the top of the frame where there were gaps) are the best candidates.

By orienting brush strokes based on only the strongest gradients, gradients at the edges of objects effectively dictate the internal brush stroke orientations. We believe this produces not only more coherent results (see Figure 7), but also more accurately describes many historical painting styles.

As gradient calculations are very sensitive to things like video noise, it is important to constrain the orientation of each brush stroke. Brush strokes ("Strong" or not) are allowed to rotate only $\delta\theta$ each frame, typically about 1 degree. Thus brush strokes are less sensitive to noise but can gradually rotate to different orientations as they move or the video changes.

(A) **(B)**

Figure 7: RBF gradient interpolation. (A) Local Gradients. (B) RBF gradients. Large strokes are Basis points. Notice the curves and swirls that resemble the style of Van Gogh.

E. Edge Clipping: Artists try to maintain the structural coherence of a painting by not allowing brush strokes to cross edges in an image. To emulate this process we first detect edges in the frames and then the brush strokes are clipped to these edges.

For all layers except the highest, the input is blurred by a Gaussian kernel of standard deviation proportional to w_b. Canny edge detection is then run on the blurred images for the lower layers and

on the unblurred input image for the highest layer [Forsyth and Ponce 2002]. We want the highest layer to capture as many high frequency edges as possible. The edge images, used to guide refinement in step C, are used here to clip the brush stroke lengths.

The brush strokes are grown length-wise from their center until they encounter an edge. This step is also similar to [Litwinowicz 1997] except that our brush stroke growth is constrained over time to δLength. Temporal incoherencies between edge images of adjacent frames, often caused by video noise, would otherwise cause the brush strokes to scintillate as they change length rapidly.

F. Color Extraction: Each brush stroke's color is determined by averaging the color of the pixels underneath it in the source frame. Color for each brush stroke is also averaged over the past $C_{s,\max}$ frames, typically five, in order to prevent video noise or interlacing from rapidly changing a brush stroke's color.

Noise can also be added to the input frames during the color extraction step in order to provide more random color variation between brush strokes. This is especially helpful when trying to imitate styles such as pointillism which depend on the perceptual merging of adjacent brush strokes with vastly different pigment to create an image.

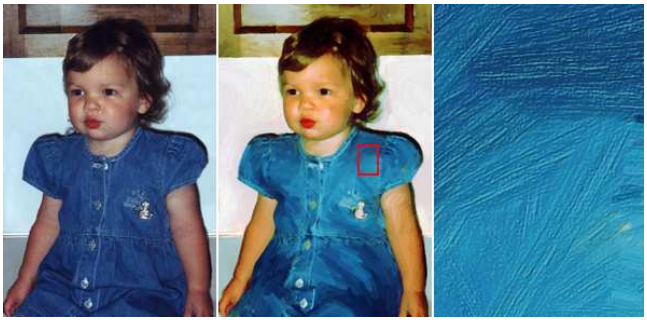


Figure 8: Rendering with brush stroke textures used as height maps. See the zoomed in pattern in the right image.

G Rendering: First the canvas is initialized to either a solid color or a canvas texture. Brush strokes are then rendered according to their draw order from bottom layer to top layer. To render each brush stroke a real brush stroke texture is clipped, rotated, colored, and alpha composited onto the canvas. These brush stroke textures are created by hand by cropping brush strokes out of real paintings of the style being emulated (see Figure 3). They are then encapsulated in a *style* for later reuse.

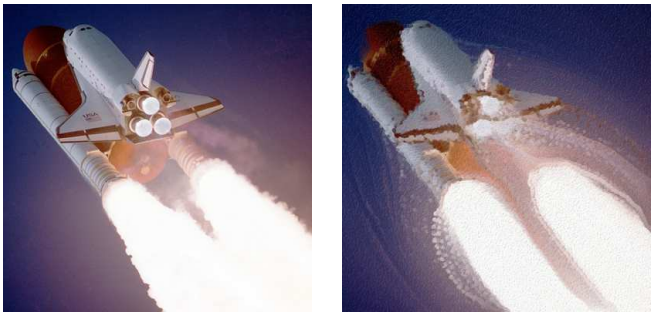
The brush stroke textures (Figure 3) are pairs of gray-scale alpha masks and textures. For each brush stroke to be rendered, the alpha mask is clipped according to the brush strokes lengths by blacking out parts of the alpha mask which should not be rendered. The hard edge that results is then radially blurred to provide soft blending between brush strokes. The texture and alpha mask are then both rotated according to the angle of the brush stroke. The texture is multiplied by the color of the brush stroke. The alpha mask is multiplied by the opacity of the brush stroke. Finally the texture (now colored) and alpha mask are alpha composited onto the canvas on top of the existing brush strokes.

Applying brush textures multiplicatively, as described above, has the effect of darkening an image. The brush texture will also be almost imperceptible in darker regions of a rendering. A better way to handle the textures is to treat them as height maps and then perform lighting calculations based on surface normals estimated from the height map. As in Hertzmann [2002] we use our alpha masks to build a color image. Our brush textures are not colored, but instead alpha composited into a blank image in order to build a height field. With the height field complete, surface normals can be estimated. Lighting calculations are then performed at each pixel, using the color image and surface normals as inputs.

It is not initially intuitive to substitute brush textures cropped from paintings for brush height maps. However, it is sufficient to produce the effects we desire: specular highlights and shading as well as a consistent lighting direction for all brush strokes (Figure 8).

5 Adding User-specified Motions

Non-photorealistic animation is not restricted to video sources. By synthesizing motion information with aid of tools like Image-Based Flow Visualization [van Wijk 2002] a user can create animations from still images that have a compelling sense of motion. Our technique accommodates this by accepting as inputs a still image and any number of optical flow fields. An arbitrarily long sequence can be created by treating the still image as the current frame while successively applying the motion information. All of the *styles* carry over, though it is sometimes interesting to increase $C_{s,\max}$ so that color is averaged over many frames. By doing so a user can create streamlines. A single frame from the resulting animation will then give the impression of both the original image and the motion information that was applied to it.



(Original) **(with Streamlines)**

Figure 9: Shuttle image with non-photorealistic streamlines added to show movement.

Motion information does not need to be synthesized. Motion information can be extracted from one video and then transplanted or added into another video. Likewise motion information from a video can be used to animate a still image. An example image shown in Figure 9).

6 Results: Expressive Painterly Animation

The results of this work are best visualized in video. Video results as well as additional still results can be found on our project web page at http://www.cc.gatech.edu/cpl/projects/artstyling/.

Our technique is very robust. We produce painterly animations from video segments that contain interlacing, are of low resolution, and span multiple scene transitions (see videos). Our technique produces poor results when motion estimation is poor. The most common causes for this are large textureless regions or fluids which optical flow can not track well. When motion estimation fails, brush strokes cease to adhere to objects in the world and the resulting animations can become unpainterly. However, these are rare circumstances. For example, we are able to apply our technique to cell shaded animations where optical flow traditionally has difficulty. Additionally, our method of animating still images is

based on the notion of applying motion information which does not match the scene and we still produce painterly results.

Computational times for our process vary based on style parameters and the complexity of each input frame. Styles that require multiple layers are slower as they tend to "over paint" the output. This happens because successive refinements are placed on top of existing paint and our rendering process is effectively fill rate limited. For the same reason, high frequency images, with many edges, require more refinement and are slower to render. Overall, the processing time for one megapixel output varies from 80 seconds a frame for Pointillism to 300 seconds a frame for Impressionism. This estimate is for video on which motion estimation must be performed, which takes about one minute per frame at DVD resolution. Rendering often takes up the majority of compute cycles and there is much room for optimization there. Our system is implemented on a Intel Pentium IV PC, using OpenCV libraries [Bradksi and Pisarevsky 2000].

Our color plate shows several results from the processing of still images. More images and videos generated using our approach are available from our project web page.

7 Conclusions

We present a new approach that builds on and refines several existing techniques for generating painterly animations. The central element in our work is a brush stroke with dynamic properties. Our brush strokes are arranged in *layers* and their behaviors are governed by user defined and selected *styles* as well information extracted from the input image, video, or motion information. Our primary contributions that extend the state of the art for painterly rendering of images and video are:

- Our use of radial basis functions to globally interpolate brush stroke orientation- This allows us to emulate styles like Van Gogh better than previous techniques have demonstrated. It also aids temporal coherency by preserving through time the basis points that influence orientation.
- Using edges to guide painterly refinement- This grants us greater stylistic freedom than previous methods. Colors in many of our renders are greatly warped, yet the images are still comprehensible because the paintings are structurally refined.
- Handling of real brush stroke textures and lighting during rendering as well as decoupling of canvas space from input space
- The discovery of new, relevant stylistic parameters
- Placing temporal constraints on brush stroke properties- Temporal incoherency, which was in the past the chief detractor of NPR video techniques, is avoided with much care in our approach both by the addition of temporal constraints to previously researched brush stroke properties and by the addition of new brush stroke properties like opacity.
- Flexible handling of motion information- Users may mix and match or synthesize motion information for videos and still images for non-photorealistic effects.

For future work, we are exploring the rendering solid concepts of Klein et al. [2002]. The notion of decoupling input and output frame rates is also interesting. There is also room to improve performance by having the rendering step performed on graphics hardware. Finally, we are looking into studying region-based methods to extend beyond pixels to cell-based renderings.

8 Acknowledgements

We are grateful to Gabriel Brostow and Tony Haro for their suggestions and assistance with this work. Source photographs are from USDA Agricultural Research Service, NASA, and Greg Turk.

References

BLACK, M., AND ANANDAN, P. 1991. Robust dynamic motion estimation over time. In *Proceedings of Computer Vision and Pattern Recognition (CVPR 1991)*, 296–302.

BORS, A. G. 2001. Introduction of the radial basis function (rbf) networks. In *Online Symposium for Electronics Engineers*, vol. 1 of *DSP Algorithms: Multimedia*.

BRADKSI, G., AND PISAREVSKY, V. 2000. Intel's computer vision library: Applications in calibration, stereo, segmentation, tracking, gesture, face, and object recognition. In *In Proceedings of IEEE Computer Vision and Pattern Recognition Conference 2000*, vol. II, II:796–797. Demonstration Paper.

DECARLO, D., AND SANTELLA, A. 2002. Stylization and abstraction of photographs. *ACM Transactions on Graphics 21*, 3 (July), 769–776.

FORSYTH, D., AND PONCE, J. 2002. *Computer Vision: A Modern Approach*. Prentice-Hall.

FREEMAN, W. T., ADELSON, E. H., AND HEEGER, D. J. 1991. Motion without movement. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, ACM Press, 27–30.

HARO, A., AND ESSA, I. 2002. Learning video processing by example. In *Proceedings 16th International Conference on Pattern Recognition*, I:487–491.

HERTZMANN, A., AND PERLIN, K. 2000. Painterly rendering for video and interaction. In *NPAR 2000 : First International Symposium on Non Photorealistic Animation and Rendering*, ACM SIGGRAPH / Eurographics, 7–12.

HERTZMANN, A., JACOBS, C. E., OLIVER, N., CURLESS, B., AND SALESIN, D. H. 2001. Image analogies. In *Proceedings of ACM SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, 327–340.

HERTZMANN, A. 1998. Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of SIGGRAPH 98*, ACM SIGGRAPH / Addison Wesley, Orlando, Florida, Computer Graphics Proceedings, Annual Conference Series, 453–460.

HERTZMANN, A. 2002. Fast paint texture. In *Second International Symposium on Non-Photorealistic Animation and Rendering (NPAR 2002)*, 91–96.

KLEIN, A. W., SLOAN, P.-P. J., FINKELSTEIN, A., AND COHEN, M. F. 2002. Stylized video cubes. In *Proceedings of the ACM SIGGRAPH symposium on Computer animation*, ACM Press, 15–22.

LANSDOWN, J., AND SCHOFIELD, S. 1995. Expressive rendering: A review of nonphotorealistic techniques. *IEEE Computer Graphics and Applications*.

LITWINOWICZ, P. 1997. Processing images and video for an impressionist effect. In *Proceedings of SIGGRAPH 97*, ACM SIGGRAPH / Addison Wesley, Los Angeles, California, Computer Graphics Proceedings, Annual Conference Series, 407–414.

MEIER, B. J. 1996. Painterly rendering for animation. In *Proceedings of SIGGRAPH 96*, vol. 30, 477–484.

REYNOLDS, C., WWW. Stylized depiction in computer graphics non-photorealistic, painterly and 'toon rendering. `http://www.red3d.com/cwr/npr/`.

VAN WIJK, J. J. 2002. Image based flow visualization. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, 745–754.

WARD, V., 1998. What dreams may come. Feature Film.

Image and Video Based Painterly Animation
James Hays and Irfan Essa