Detail-Preserving Paint Modeling for 3D Brushes

Nelson Chu William Baxter Li-Yi Wei Naga Govindaraju
Microsoft Research

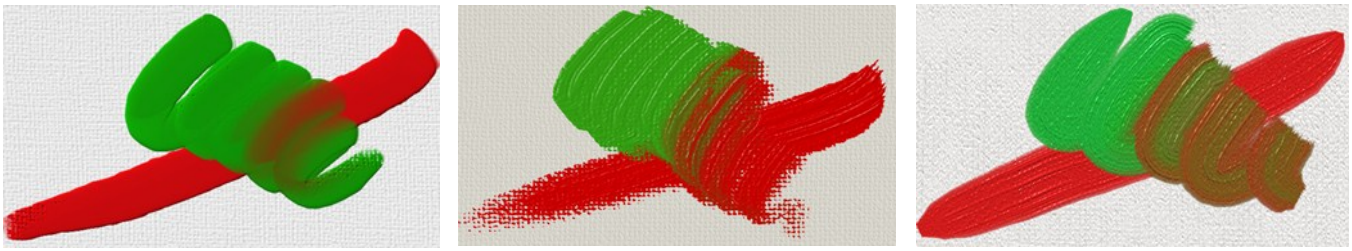

Figure 1: *Oil paint simulated by an implementation of the system presented in [Baxter et al. 2001] (left), by our method (middle) and by the commercial package ArtRage 3 (right). Compared with [Baxter et al. 2001], our technique produces more organic color streaks and much less color blurring. Compared with the ArtRage strokes, ours appear more natural since the ArtRage method of sweeping a 1D texture to simulate strokes give less variation and control than the 3D deformable brushes offered by our system.*

Abstract

Recent years have witnessed significant advances in 3D brush modeling and simulation in digital paint tools. Compared with traditional 2D brushes, a 3D brush can be both more intuitive and more expressive by offering an experience closer to wielding a real, physical brush. To support popular media types such as oil and pastel, most previous 3D brush models have implemented paint smearing and mixing. This is generally accomplished by a simple repeated exchange of paint between the 3D brush and 2D canvas, with the paint picked up by the brush typically mapped directly onto the brush surface. In this paper we demonstrate that both repeated exchanges and direct mapping of paint onto brush surfaces are sub-optimal choices, leading to excessive loss of color detail and computational inefficiencies. We present new techniques to solve both problems, first by using a *canvas snapshot buffer* to prevent repeated paint exchange, and second by mapping brush paint onto a 2D, *resolution-matched pickup map* that sits underneath the brush, instead of mapping onto the 3D brush itself. Together, these act to minimize resampling artifacts, helping to preserve fine streaks and color details in strokes, while at the same time yielding improved efficiency by never sampling the brush more densely than necessary. We demonstrate the effectiveness of our method in a real-time paint system implemented on the GPU that simulates pastel and oil paint. Our method is simple and effective, and achieves a level of realism for these two media not attained by any previous work.

Keywords: digital painting, virtual tools, resampling, oil painting, pastel, natural media modeling

1. Introduction

Digital painting has been an active subject of study for over 30 years [Smith 2001] and has become an indispensible component of common graphic tool sets (e.g. the Adobe Creative Suite). Unlike physical painting with real brush and canvas, digital painting offers several unique capabilities such as easy undo and saving the artwork at different stages, as well as lossless copy and reproduction. As a result, digital painting has become a popular way to create illustrations for everything from books, to advertising, to background mattes in movie production.

However there are still some nuances of real art media and positive attributes of real art tools which are not found in current digital systems. Specifically, marks made by real-world tools have an organic richness of detail and expressive variability that have yet to be captured adequately by any digital painting system. In this work, we focus in particular on oil and pastel, two media in which the interactions of tools and media are critical and give rise to both detail and variability. Researchers have shown that the lack of expressive variability can be addressed by moving from simple 2D bitmap "brushes" to more realistic models of real-world tools, such as deformable 3D paint brush models [Baxter *et al.* 2001; Chu and Tai 2002; Adams *et al.* 2004; Baxter and Lin 2004; Okaichi *et al.* 2008; Baxter and Govindaraju 2010]. However, basically none of these previous techniques is able to recreate the fine details seen in real oil or pastel marks due to how the smearing and mixing of paint are implemented. We find that even with currently available commercial tools that specialize in natural art media (see Figure 3), stroke variation and/or color detail are lacking, making a recreation of effects like in Figure 2 nearly impossible.

Figure 2: *Close-up of a real painting showing subtle streaks (top, oil painting © Scott Burdick; used with permisson) and smearing (bottom, pastel).*

For many artistic styles involving media such as oil paint and pastel, smearing and the detailed interactions that arise from it are essential components. We can think of smearing as a bidirectional and simultaneous process consisting of depositing paint from the brush to the canvas as well as picking up paint from the canvas onto the brush. To produce a smearing effect digitally, the simplest and most direct approach is to approximate the interaction as a sequence of repeated discrete transfers of paint between the brush and the canvas. This is the approach taken by *e.g.* [Baxter *et al.* 2001; Baxter *et al.* 2004a; Baxter *et al.* 2004b; Van Haevre *et al.* 2007; Van Laerhoven and Van Reeth 2007].

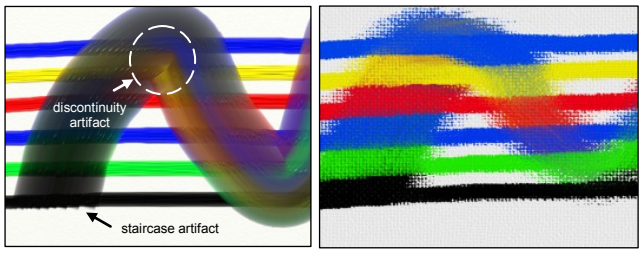

Figure 3: *A comparison of smearing effects by ArtRage 3 (left) and by our system (right). In these tests, several horizontal strokes are smudged in a zigzag motion. Note that in the former only certain colors are picked up (dominated by black with other colors mostly ignored). Sweeping a 1D-texture in the former also creates discontinuity artifacts at acute corners. Corel Painter's latest "Artists'Oils" produces results similar to Artrage's.*

In an effort to support such smearing effects, prior 3D brush models *e.g.* [Baxter *et al.* 2001; Baxter and Lin 2004; Van Haevre *et al.* 2007; Van Laerhoven and Van Reeth 2007] have allowed the 3D brush to pick up paint on its surface. However, since the brush samples are generally not aligned with those of the canvas, some loss of detail due to resampling is inevitable, along with inefficiencies similar to those that motivate the use of mip-mapping for texture rendering. This alone might not be catastrophic; however, in previous work the one-time resampling error is compounded many fold by repeatedly resampling paint that was just deposited. Typically, paint is transferred between the canvas and brush hundreds of times during a stroke, with resampling occurring on each transfer. This rapidly iterated resampling results in excessive blurring of the paint color as can be seen in Figure 1 (left) and Figure 16. We observe that it is essential to minimize this resampling in order to produce high-quality smearing results exhibiting sharp detail like that seen in Figure 2.

Contributions: In this paper, we first analyze the key sources of blurring associated with digital painting algorithms for 3D brushes. Based on our analysis, we propose two novel techniques to drastically reduce this blurring. First, we propose a new brush representation that uses a 3D brush plus a 2D, *resolution-matched pickup map* that sits underneath the brush. The pickup map is used for storing paint picked up from the canvas, instead of mapping this data directly onto the brush geometry as in previous work. This eliminates the problems associated with resolution mismatch between brush and canvas. Second, we introduce a special *canvas snapshot buffer* to avoid repeated resampling of the paint by preventing pickup of paint just deposited. Importantly, with our snapshot buffer technique, picking up paint less recently deposited *is* allowed, leading to more realistic behavior for self-intersecting strokes. All of our techniques leverage the hardware acceleration capabilities of GPUs.

Our new techniques have the advantage of minimizing resampling artifacts associated with the use of a 3D brush model, allowing us to simulate high-quality paint smearing efficiently. Using these techniques, we developed a real-time paint system that simulates pastel and oil paint, two art media that rely heavily on smearing. We performed quality and performance comparisons with prior algorithms [Baxter *et al.* 2001; Baxter and Lin 2004]. In practice, we observed our algorithms to achieve much higher quality than prior hardware-accelerated implementations without increasing the computational cost.

2. Background

Many traditional artists prefer not to use digital paint media because the existing digital tools lack the complexity and organic appearance of real media (compare *e.g.* Figures 2 and 19 with Figure 3 (left)). The complex nature of brushes and fluid paint is inherently difficult to reproduce digitally in real time. With viscous media like oil, or dry particulate media like chalk pastel, reproducing detailed smearing effects is particularly challenging.

In oil painting, *wet-in-wet* [Appellof 1993] is a common technique that artists rely on. Stroking wet-in-wet means applying colors over and into one another while still wet. Technically, this process is just smearing wet paint on the canvas while depositing. Wet-in-wet stroking is a key feature in the *alla prima* style, in which colors are laid down more or less as they would appear in the finished painting. Reworking is kept to a minimum to give a spontaneous effect. When paints are smeared, one should see delicate color streaks like those shown in Figure 2 or Figure 19 (left). Artists also invented a technique called *broken color* [Appellof 1993; Flattmann 2007], which is to juxtapose different colors on the canvas and let the viewer's eye mix them so that the colors stay vibrant. Often, broken colors are made with quick wet-in-wet strokes so that traces of previously deposited colors peek through newly deposited ones (Figure 2). The broken color technique also applies to other art media like pastel.

To improve the digital painting experience, researchers have devised computational models of various art media and tools in the past few decades. Recent advances show a trend towards using 3D deformable brushes instead of 2D ones to better capture complex brush deformation to give more varied and controllable brush marks (*e.g.* [Baxter *et al.* 2001; Chu and Tai 2002; Adams *et al.* 2004; Baxter and Lin 2004; Okaichi *et al.* 2008]). There is also a trend towards physically-based methods to better simulate the organic behavior of real paint (e.g. [Curtis *et al.* 1997; Baxter *et al.* 2004a; Baxter *et al.* 2004b; Rudolf *et al.* 2005; Chu and Tai 2005]). The state of the art in watercolor simulation is quite good [Curtis *et al.* 1997; Chu and Tai 2005; Van Laerhoven and Van Reeth 2007]. However, watercolor simulation is based on modeling low-viscosity fluid, which tends to be very diffusive when wet and fixed when dry, so detailed smearing effects (crucial in pastel and oil) do not arise. We next briefly review the state-of-the-art in the simulation of pastel and oil paint media.

Pastel Simulation: Van Haevre *et al.* [2007] simulate pastel painting with a pastel pencil. Deposition and smearing are implemented by performing two-way paint transfer between pencil and paper. A height-field is used to represent the 3D geometry of the pencil tip and a 2D texture stores the paint pickup and is mapped onto the pencil tip. Paper weathering is also considered, in which the paper can be dented if the user presses too hard. Rudolf *et al.* [2005] also simulated wax crayon similarly with paper and crayon tip modelled as height-fields. However, they assumed wax would not be carried a long distance and modelled smearing by redistributing paint from a paper cell to its 8 neighbors using a 3×3 mask rather than using a separate texture that stores picked up paint. Both Van Haevre *et al.* [2007] and Rudolf *et al.* [2005] simulated crayon weathering by trimming the crayon height-field.

Oil Paint Simulation: Baxter *et al.* [2001; 2004a; 2004b] simulated oil paint with various levels of complexity. Their first method [Baxter *et al.* 2001] was the simplest and closest to ours: two-way transfer of paint between canvas and brush was performed via a few texture buffers. Their second method [Baxter *et al.* 2004a] uses a conservative advection scheme in addition to the two-way transfer to simulate the paint dynamics, and a full-spectrum rendering method for higher color accuracy. Their third method [Baxter *et al.* 2004b] models paint motion with the Stokes' equations, which are solved in 3D to allow a true volumetric modeling of oil paint. However, this gives too low a performance for practical use on current systems due to large computational requirements. All these paint models are coupled with a 3D brush model [Baxter *et al.* 2001; Baxter and Lin 2004]. Note that none of these three models was capable of preserving fine details. Okaichi *et al.* [2008] also simulated thick oil paint with a two-way transfer model similar to [Baxter *et al.* 2004a] along with a 3D painting knife model.

Commercial Packages: Popular digital paint packages like *Corel Painter 11* or *Ambient Design ArtRage 3* feature pastel and oil paint models. To perform smearing, they appear to sample the paint on the canvas coarsely when they simulate paint pickup because the results lack detail (Figures 1 right and 3 left). Digital painters also use *Adobe Photoshop* to paint, but its tool set is not specifically designed for simulating real paint media, so the results tend more towards photorealistic styles. As far as we can tell, all these packages render the strokes either by dabbing a 2D footprint or sweeping a 1D texture along a stroke path.

3. Overview

In this section, we first describe a common paint simulation pipeline. We then explain how the issue of resampling arises and how our proposed techniques solve the problem.

3.1 Paint Simulation Pipeline

An interactive painting application obtains user input via an input device such as a graphics tablet, simulates the paint deposition, and then displays the deposited paint onto the screen. We specifically focus on interactive techniques for the simulation of the paint medium using a graphics processing unit (GPU). These algorithms assume the canvas resolution is fixed during a simulation and specified a priori by the user. A 3D brush drives the underlying paint simulation. The brush geometry can deform based on the orientation of the brush handle and pressure. Moreover, the brush geometry can transfer paint to and pick up paint from the canvas, *i.e.* a bidirectional transfer for paint smearing and mixing effects. The core steps involved in implementing the painting simulation in both our system and previous work are shown in Figure 4.

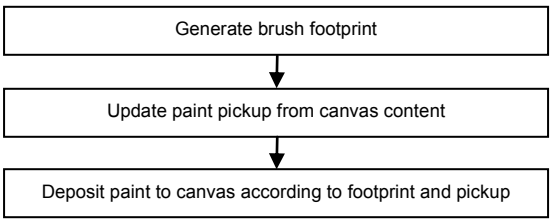

Figure 4: *The core steps for one brush footprint impression.*

To generate a footprint, we render the depth value of the brush geometry into a height-field and modulate this height-field with the canvas tooth to produce the footprint (Figure 5) similar to [Baxter *et al.* 2001; Chu and Tai 2002; Adams *et al.* 2004; Van Haevre *et al.* 2007]. The user input is sampled at discrete points, and a stroke is generated by repeatedly imprinting the 2D brush footprint between the two successive input points (Figure 6). The spacing between adjacent imprints has to be no larger than a single pixel on the canvas if the stroke is to be free of artifacts along the edges. While performing the stroke generation, the system needs to simulate paint being picked up by the brush from the canvas in addition to depositing the paint from the brush. This is performed during each imprint step.

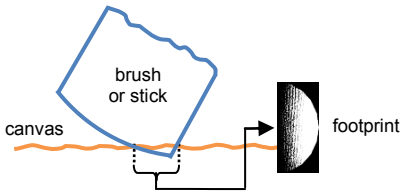

Figure 5: *Intersecting brush geometry to get a footprint.*

Figure 6: *Imprinting a brush footprint along a stroke trajectory.*

3.2 Sampling and Resampling Issues

Previous researchers [Baxter *et al.* 2001; Baxter *et al.* 2004a; Baxter *et al.* 2004b; Okaichi *et al.* 2008] have used the above pipeline to simulate paint deposition with a smearing effect. However, the smearing of multiple colors in these works displays significantly more color diffusion than in real paint. Specifically, one expects to see streaks of different colors when paint is only gently mixed (compare Figure 2 (top) and Figure 17 with Figure 1 (left)). We identify two causes for this loss of detail. The first is that the effective simulation resolution is limited by the lower of the canvas resolution and the brush pickup resolution. Often, the canvas and/or the brush resolutions are limited by the real-time requirement of the application. The other cause is resampling that occurs during paint transfer. We next examine these causes more closely.

In general, numerical diffusion is common in digital reproductions of various natural phenomena. This is due to the finite resolution and the repeated resampling used in the computation. Researchers in fluid simulation have come up with various techniques to tackle this issue (e.g. the classic Riemann problem solutions for shock preservation in fluid dynamics [LeVeque 1992] or more recent examples in the graphics literature [Weiskopf 2004; Kim *et al.* 2005]). In image processing, various methods have also been applied to preserve edge details while resampling [Barash 2000]. These techniques, however, add significant computational overhead to the simulation and to our knowledge have not been implemented in any digital paint programs.

In previous 3D brush (or pastel stick or palette knife) models [Baxter *et al.* 2001; Adams *et al.* 2004; Baxter and Lin 2004; Van Haevre *et al.* 2007; Okaichi *et al.* 2008], paint pickup information is mapped to the brush geometry. Specifically, pickup is stored either as textures mapped on to the brush surface [Baxter *et al.* 2001; Baxter and Lin 2004; Van Haevre *et al.* 2007], as vertex data on the brush surface mesh [Okaichi *et al.* 2008], or as samples scattered on the brush surface [Adams *et al.* 2004]. Paint on the canvas, on the other hand, is stored as textures mapped to the canvas [Baxter *et al.* 2001; Baxter and Lin 2004; Van Haevre *et al.* 2007; Okaichi *et al.* 2008] or as samples on the canvas

surface [Adams *et al.* 2004]. In general, using a deformable 3D brush model makes it hard to ensure we have matched resolutions on the brush and canvas surfaces, as projecting the brush surface down to the canvas changes the relative sampling densities. The brush deformation can also be significant (e.g. the brush tip may spread). Most previous models (with rare exceptions like [Adams *et al.* 2004]) simply use a static resolution for the brush (Figure 7). This causes paint data to be up-sampled or down-sampled when we perform paint exchanges between the two surfaces. This implies the fidelity of the higher resolution data is wasted when the resolutions do not match.

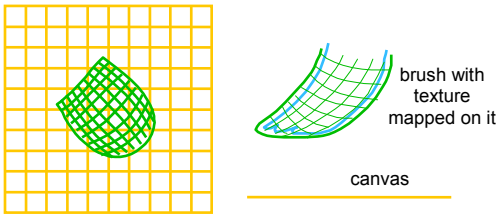

Figure 7: *Canvas and brush with different resolution and alignment.*

The problems of resolution mismatch and sampling are encountered frequently in computer graphics. In some ways the problem of establishing a mapping between a 3D brush and the canvas is exactly the problem of texture filtering. Transferring paint from the brush to the canvas is essentially a matter of rendering a textured brush under an orthographic projection. As such, mip-mapping the brush data would help solve the problems of either too much or too little resolution on the brush; however, it becomes expensive to update all the mip levels, and furthermore mip-mapping is inherently isotropic, whereas the sampling mismatch between a projected brush and canvas really requires anisotropic filtering, which is even more expensive.

On the other hand, the problem of mapping from the canvas back to a 3D brush is in some ways similar to the problem of shadow mapping or projective texturing. In this case, large stretch in the some parts of parameterization of the canvas projected onto the brush is unavoidable. This is the case for portions of the brush geometry nearly perpendicular to the canvas, which means that for some portions of the brush, the canvas-to-brush mapping will always be undersampled.

Even if resolutions were roughly matched, there is another perhaps even more serious sampling issue. With previous implementations [Baxter *et al.* 2001; Van Haevre *et al.* 2007; Okaichi *et al.* 2008], paint just deposited is immediately picked up again, and thus resampled again, to implement smearing. This creates a tight feedback loop that magnifies the blurring. For a brush footprint that is N pixels wide, stroking will resample each pixel along the stroke about N times, essentially applying an N-iterated blurring filter to the color data.

In the following discussion of our solutions to these problems, we adopt the texture-based pickup storage method and call the texture used to represent a layer of paint picked up by the brush the *pickup map*. We denote the texture that stores paint deposited on the canvas by *canvas map*.

4. Our Method

The key issue we have identified with previous work which attempts to simulate media like oil paint and pastel is the inability to preserve fine-scale color detail during smearing and mixing, particularly in conjunction with 3D brush models. We propose two new techniques that eliminate almost all of this blurring, explained in the next two subsections.

4.1 Resolution-Matched Pickup Map

To address the resampling problem related to differing resolutions of brush paint and canvas paint, we propose the use of a *resolution-matched pickup map* (Figure 8). Conceptually, this is a bitmap with the same resolution as the canvas, sitting under the 3D brush. We call our proposition a hybrid approach because the brush geometry is 3D but the pickup information is represented on a flat 2D surface. Instead of performing bidirectional paint transfer between the canvas and the 3D brush, we now perform the same operations between the canvas and this 2D surface. To correlate the 3D brush motion/deformation with the 2D pickup map, we have the current footprint of the former act as a mask for the latter when performing the paint transfer.

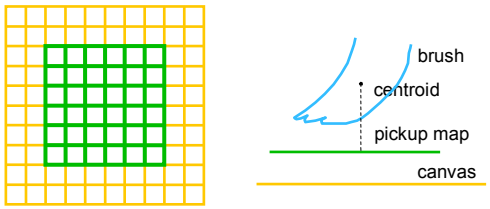

Figure 8: *A flat pickup map sitting under the 3D brush with resolution matched to that of the canvas.*

We align the center of this pickup map to an anchor point on the 3D brush, which can be the centroid of the bristle geometry for a brush (Figure 8 right), or the center of the base for a pastel stick. Given a certain brush size, we use the diagonal of the 3D bounding-box of the brush in rest position to bound the pickup map size so it is big enough to cover the brush for all possible brush transformation. If the user scales the brush, we scale the pickup map also to maintain matched resolutions. As the resolution of this pickup map matches that of the canvas, there is no issue of resampling due to a mismatch of resolutions.

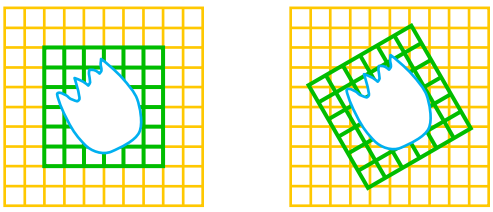

Figure 9: *Restricting the pickup map to align to the pixel grid of the canvas irrespective to brush orientation (left) and rotating the map according the brush orientation (right).*

If we restrict the pickup map to align with the canvas pixel grid irrespective to how the brush is rotated relative to the canvas (Figure 9 left), we can eliminate resampling due to geometry misalignment too. However, this also means that paint picked up on one particular part of the brush does not stick to the same spot if the brush is rotated. Rotating the pickup map about its center so that the colors stay close to their original 3D positions (Figure 9 right) largely alleviates this problem - the only dislocation is then due to the local bristle deformation and twisting of the brush. Given that even with a real brush it is not always clear exactly what paint has been picked up where, we find this to be acceptable for a practical user painting experience. Note that rotating the pickup map does cause resampling, but this does not significantly

degrade quality since the resolutions between canvas and pickup maps are matched and the resampling does not happen repeatedly.

4.2 Canvas Snapshot Buffer

The second issue responsible for loss of detail in previous work is the repeated resampling of paint. To greatly reduce this, we propose the use of a *canvas snapshot buffer*, Ω. As previously explained, in a typical paint pipeline, paint is transferred between the brush and canvas at every impression along a stroke. We modify that typical pipeline, using our snapshot buffer, as follows (see Figure 10). Before the first imprint of a stroke, Ω is initialized to be identical to the current canvas map. Then, before every subsequent imprint, Ω is updated to contain the latest version of the canvas map *except* for the region covered by the pickup map at the current brush position (green rectangle, Figure 10). By using Ω as the input canvas map to our paint pickup update algorithm instead of the canvas itself, we avoid the tight feedback loop during the bidirectional paint transfer, and thus effectively prevent the blurring that has plagued previous systems. The use of Ω also helps avoid quickly saturating to the brush color when blending with canvas paint (see Section 5, Deposition). For comparisons with and without a snapshot buffer see Figure 14 and Figure 16.

Figure 10: *The canvas snapshot buffer* Ω *is all up-to-date except for the area covered by the pickup map, preventing repeated resampling and the attending blurring.*

The above simple update scheme works well in allowing the effect of self-overlapping except for acute stroke corners (Figure 11 middle) and a few special cases like drawing a loop smaller than the pickup map size. These issues can be fixed by keeping track of the recent addition of the current stroke to produce a mask with which we decide if we use the current canvas map in our paint transfer operations. However, in our current system we only tackle the acute stroke issue by updating Ω with a complete snapshot when we detect a large change in the stroke direction, because the other occasions needing a fix are rare in practice. Finally, should no self-overlapping (Figure 11 left) be preferred, we can simply update Ω only at the beginning of a stroke.

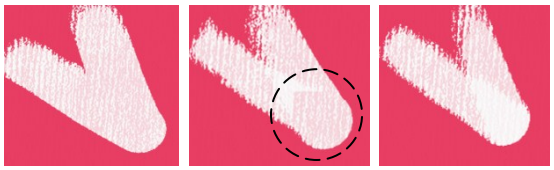

Figure 11: *No self-overlapping (left), self-overlapping with artifact (middle) and correct self-overlapping (right).*

5. Paint Model Details

We have implemented a paint system (Figure 12) that uses the proposed techniques to simulate pastel and oil paint. In certain parts of our paint model, we also made the simulation not strictly physical in favor of artistic control. For pastel simulation, we use a non-deforming 3D stick model as the paint applicator. We render pastel artwork with a flat and matte appearance. For oil painting, we use a deforming brush model as in [Baxter and Govindaraju 2010]. We also simulate ridges in oil paint formed by the brush bristles and render the paint glossy. Other parts of the simulation are essentially the same for both media. Although we use 3D brush models in our system, we note that our paint transfer pipeline is not tied to any particular 3D brush simulation technique. Furthermore, our method also integrates seamlessly with a traditional 2D brush model by simply treating the brush footprint as static—the 3D portion of our 3D/2D hybrid then just goes away. For instance, our system switches to a 2D brush when we have touch input (Figure 12 inset). In the rest of this section, we describe the details of our paint modeling. All operations described are performed in a per-pixel fashion using DirectX 10 HLSL shader programs on a GPU.

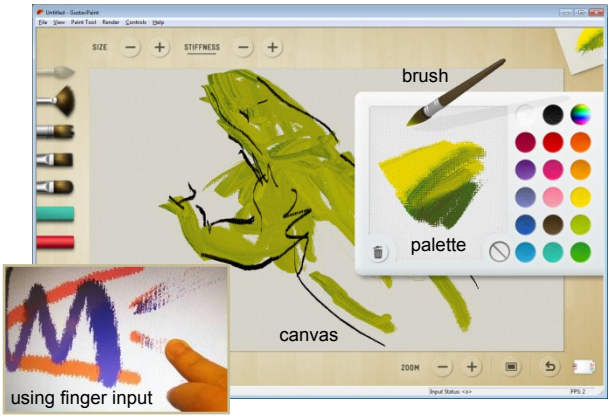

Figure 12: *User interface of our paint system.*

Deposition: If we simply copy the brush source color onto the canvas as we imprint with the footprint without picking up existing canvas paint, we obtain solid strokes (Figure 13 left). To simulate bidirectional transfer, we render the strokes as a blend of the source color and the canvas map, depending on the brush footprint height value. To simulate re-deposition of paint pickup (Figure 13 middle and right), we determine the source color as:

$$C_S = lerp(C_I, C_P, A_P),$$

where C_S is the source color, C_I is the intrinsic color of the pastel stick or the brush loading, C_P is the color on the pickup map, A_P is the thickness of the pickup layer, and *lerp(a, b, t)* is the linear interpolation function defined as $a \cdot (1-t) + b \cdot t$.

Figure 13: *Depositing paint simply as solid color (left), and with textural detail giving a broken-color quality (right).*

If we use the deposition algorithm described so far without our snapshot buffer, the overlapping imprints quickly saturate the stroke to the source color giving a solid stroke almost like Figure 13 (left) because the pickup map is immediately filled with paint just deposited. Thus, the use of our canvas snapshot buffer (Section 4.2) not only helps with resampling issues, but also helps avoid this quick saturation of color, while keeping the deposition algorithm simple.

Note that in reality, the canvas tooth grips pastel pigments only up to a certain extent (amount of pickup is also limited likewise). Further deposition results in pastel dust that does not stay on the

surface. We could model this by adding a step in our deposition algorithm to check for a maximum deposition limit. However, we choose not to because this makes it impossible to continue overlaying strokes with nice textural blending (Figure 13 right).

Ω as Artistic Control: We can actually use the presence or absence of our snapshot buffer, Ω, as an artistic control. Should we want the paint to blend more, we can use the current canvas map instead of Ω in our algorithm (Figure 14 left). The increased blurring gives the impression of wet blending (when solvent is added to the paint) where the pigments mix more thoroughly. To control the amount of blending, we can simply use an interpolation of the canvas map and Ω as input to our algorithm.

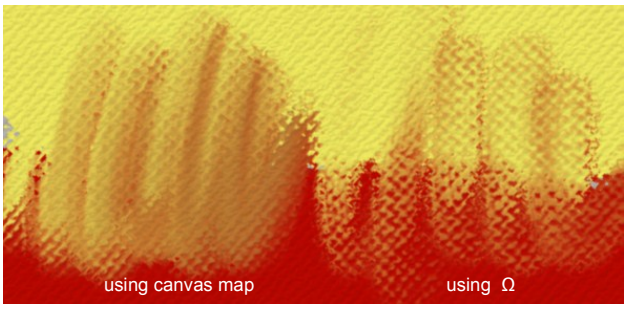

Figure 14: *A comparison of smearing using different buffers as input to our paint pickup update algorithm.*

Color Streaks and Ridges: To create color streaks, we use a bump texture to modulate the brush footprint. To simulate ridges, we use a separate texture to drive paint thickness accumulation and removal at various spots touched by the brush. Both color streaks and ridges are shown in Figures 15 and 19.

Figure 15: *Close-up of an oil painting simulated with our system showing subtle ridges and color streaks.*

6. Results and Discussion

Quality Verification: To confirm that our method of using a combination of a resolution-matched pickup map and the canvas snapshot, Ω, minimizes blurring, we conducted simple tests in which we smear one-pixel grid lines of paint strictly horizontally (Figure 16). When we use a brush of 60 × 60 pixels dab size with a brush-mapped texture at 128 × 128 resolution (using an implementation of [Baxter *et al.* 2001]), we observe prominent blurring (Figure 16 first row). We also used a 1024 × 1024 brush-mapped texture to ensure the brush had sufficient resolution, but we obtained similar results (not shown), indicating brush resolution was already matching or exceeding the canvas resolution. The rest of the entries in Figure 16 were obtained using a 60 × 60 size brush with a resolution-matched pickup map and with different options in our simulation toggled. Note that since we are smearing horizontally, we should see the blue horizontal lines stay nearly unchanged. We see that the use of resolution-matched pickup map alone does not remove excessive blurring because the blue lines diffuse vertically in the second row, although less severely than in the first row. The blue lines in the last three rows stay much sharper. With the pickup map restricted to be canvas-aligned, we see the blue lines stay perfectly sharp (rows 3 and 5). We conclude, however, that the fourth row with rotatable pickup map is the best choice overall because it minimizes blurring while allowing the paint to stick roughly to the correct location on the brush. Furthermore, the small amount of resampling due to map rotation can actually be beneficial for its anti-aliasing effect.

Resolution -matched pickup map	Pickup map canvas-aligned	Using Ω	Smearing results
N/A	N/A	No	
Yes	No	No	
Yes	Yes	No	
Yes	No	Yes	
Yes	Yes	Yes	

Figure 16: *Smearing test results. The top row result was obtained with previous method of using a brush-mapped texture [Baxter et al. 2001], while the rest were done with our system.*

Performance: Our system is quite responsive even on a low-end laptop – with a brush of dab size 60 × 60 pixels, it runs at 145 frames per second (FPS) on a tabletPC with an AMD Turion X2 Dual-Core Mobile RM-75 2.2GHz CPU and an ATI Mobility Radeon HD3200 GPU.

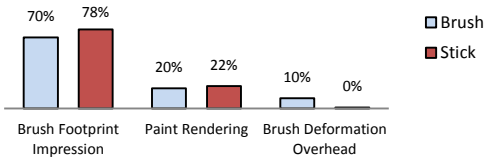

Figure 17: *A breakdown of GPU processing cost in our system.*

We perform paint simulation entirely on the GPU and brush dynamics simulation [Baxter and Govindaraju 2010] using a combination of CPU and GPU. In most cases, our system is GPU bound. Figure 17 shows a breakdown of GPU processing cost of our simulation in a typical scenario where the artwork is rendered at one-to-one zoom level and a brush of dab size 60 × 60 pixels is dragged at a moderate speed while maintaining a frame-rate of 145 FPS. Note that the brush footprint impression is a major expense taking 70% or 78% of the total cost for the two cases of using a deformable brush and a non-deforming stick as the paint applicator, respectively. For an accurate evaluation of our paint transfer performance, we vary the brush size within a reasonable range and plot a graph showing the number of brush footprint

impressions per second (Figure 18). Our data indicates a clear decrease in number of impressions with increase in brush coverage, confirming that the paint transfer performance is largely determined by the number of processed pixels.

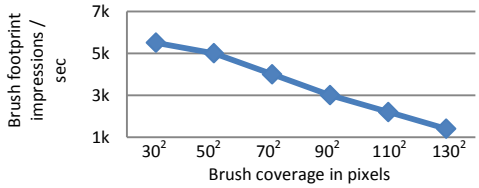

Figure 18: *Performance data of our paint simulation.*

For a system using brush-mapped texture to get a smearing quality on par with our use of pickup map, one may try to increase the canvas resolution (although the result would not be as good because canvas pixels would be blurred). Figure 16 indicates that at least an increase of 64x higher resolution is needed. This would have a major impact on the paint transfer performance and would lead to lower interactivity as demonstrated in Figures 17 and 18.

Sample Artwork: To demonstrate the usability of our system, we show a few sample artworks, which were all done in a resolution of 2k × 1k pixels. Figure 19 left is a close-up of a real oil painting by artist Scott Burdick while Figure 19 right shows a painting made with our system to mimic that painting. Note that the simulated ridges and color streaks approach the same level of organic quality as the real painting. Figure 20 is another painting made using our oil paint model. Figure 21 shows a pastel painting created using our system. The background was made by smearing black and shades of blue. Because of the variations in the footprint, we have bits of black showing in the blues, which gives a pleasing, natural look (Figure 21 left inset). Figure 22 is another sample pastel painting made with our system. Note that our model is able to give the powdery feel of real soft pastel as colors are smeared (Figure 22 inset). We encourage the readers to watch our supplementary video to appreciate the effectiveness of our simulation.

6. Conclusion and Future Work

We have presented a new simple but effective brush representation scheme for digital painting that preserves desirable nuances in the brush marks. Achieving the same level of detail would otherwise require complex algorithms to reduce data diffusion or greater computation to maintain a higher effective simulation resolution. We showed how we apply our scheme in a simplified simulation pipeline for pastel and oil paint. We are able to produce high quality artwork in real-time even on a low-end PC. Our techniques for modeling the brush representation and paint transfer are general and can be applied to other paint simulation algorithms including systems with 2D brushes, but they are particularly effective in minimizing the blurring artifacts that have been characteristic of previous 3D brush systems.

One major insight we garnered from this research is that storing paint pickup information in a flat 2D surface would actually produce better results than doing so on a 3D brush geometry. This is somehow counter-intuitive, as most people might consider the latter a more natural choice. Nevertheless, through our analysis and experiments, we have found that even though using a 2D pickup map would not attach paints as precisely as mapping the information onto a 3D brush model, the overall quality still improves significantly due to much reduced bidirectional resampling issues with the canvas.

There are several other avenues for future work. To further reduce the effect of paint picked up not sticking to the same spot on the brush (Section 4.1), we can shift the pickup map to account for brush twisting. For oil painting, currently we only simulate shallow ridges. It would be nice to add thick paint support [Baxter *et al*. 2004a; Baxter *et al.* 2004b; Okaichi *et al.* 2008] so that the user can paint in the impasto style [Appellof 1993]. Furthermore, we have yet to simulate oil paint being thinned to give a semi-transparent quality. Finally, the current system uses additive RGB color space; a subtractive color model, or more sophisticated models [Baxter *et al.* 2004a; Xu *et al.* 2007], can be employed instead should we want the colors to behave more realistically. For pastel, we can add more realism by modeling tool and surface weathering [Rudolf *et al.* 2005; Van Haevre *et al.* 2007].

Acknowledgments

We would like to thank Craig Mundie, Dan Reed, and John Manferdelli for their support of this work. We thank Avneesh Sud for his help in interfacing with multi-touch and pen input in our prototype system. We are grateful to artists Scott Burdick and Stéphanie Valentin for allowing us to show their work in Figures 2 and 19 and the accompanying video. Finally, we thank Nicholas Kamuda for painting Figure 20.

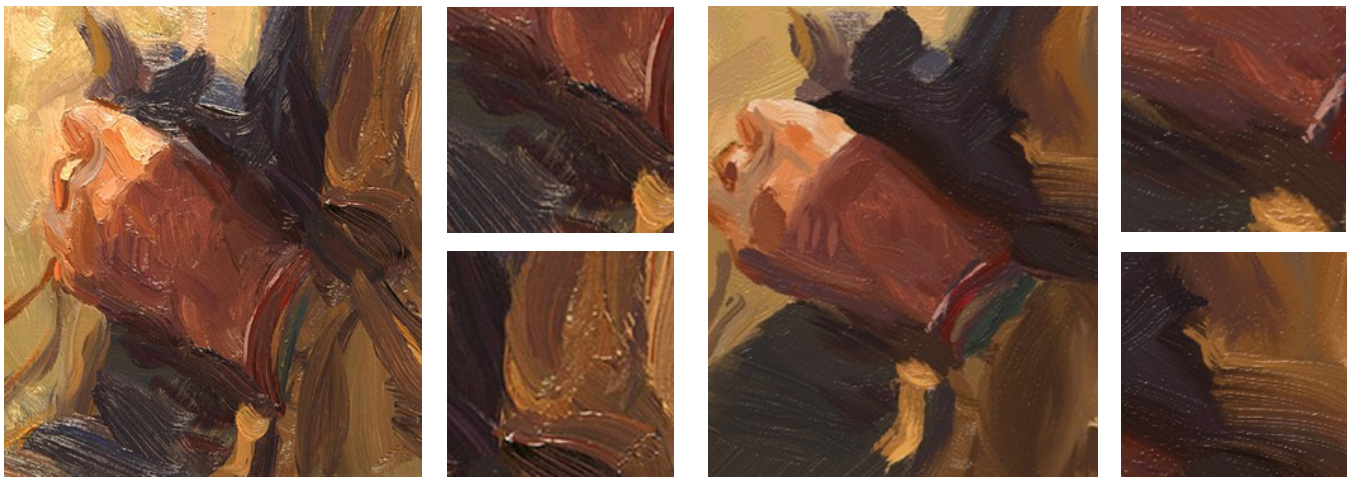

Figure 19: *Part of a real oil painting (left) and a similar result created by an artist with our system (right). The real painting was done by Scott Burdick (© Scott Burdick; used with permission).*

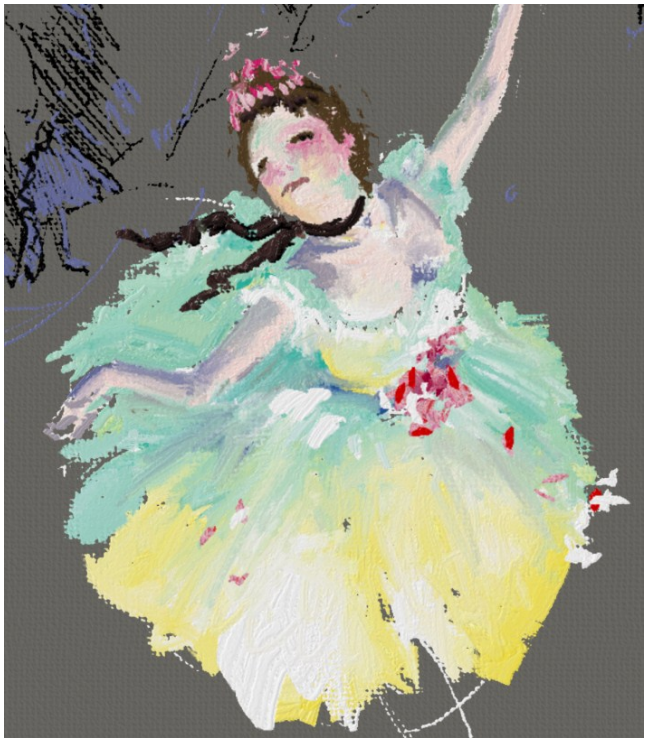

Figure 20: *An oil painting created with our system.*

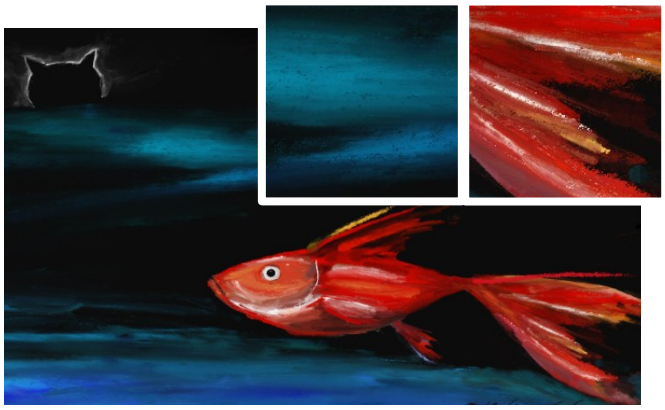

Figure 21: *A pastel painting made with our system.*

Figure 22: Close-up of a pastel painting made with our system.

References

ADAMS, B., WICKE, M., DUTRÉ, P., GROSS, M., PAULY, M. AND TESCHNER, M. 2004. Interactive 3D Painting on Point-Sampled Objects. Eurographics Sym. on Point-Based Graphics, 2004.

APPELLOF, M. 1993. Everything You Ever Wanted to Know about Oil Painting. Watson-Guptill.

BARASH, D. 2000. Bilateral filtering and anisotropic diffusion: Towards a unified viewpoint. HP Laboratories Technical Report, 2000.

BAXTER, W. AND GOVINDARAJU, N. 2010. Simple data-driven modeling of brushes. In Proceedings of the 2010 ACM SIGGRAPH Symposium on interactive 3D Graphics and Games.

BAXTER, W. AND LIN, M. 2004. A Versatile Interactive 3D Brush Model. In Proc. of Pacific Graphics 2004.

BAXTER, W., LIU, Y., AND LIN, M. C. 2004. A viscous paint model for interactive applications. J. Comput. Animat. Virtual Worlds 15, 3-4 (Jul. 2004), pp. 433-441.

BAXTER, W., SCHEIB, V., LIN, M., AND MANOCHA, D. 2001. DAB: Interactive Haptic Painting with 3D Virtual Brushes. In Proc. of ACM SIGGRAPH 2001.

BAXTER, W., WENDT, J., AND LIN, M. 2004. IMPaSTo: a realistic, interactive model for paint. In Proc. of the 3rd int'l Sym. on Non-Photorealistic Animation and Rendering, June, 2004.

CHU, S. H. AND TAI, C.-L. 2002. An efficient brush model for physically-based 3D painting. In Proc. of Pacific Graphics, 2002.

CHU, S. H. AND TAI, C.-L. 2005. MoXi: Real-Time Ink Dispersion in Absorbent Paper. In Proc. of ACM SIGGRAPH 2005.

CURTIS, C. J., ANDERSON, S. E., SEIMS, J. E., FLEISCHER, K. W., AND SALESIN, D. H. 1997. Computer-generated watercolor. In Proc. of ACM SIGGRAPH 1997.

DOCIU, D., DAHLING, M., SEEGMILLER, D., AND DELON, M. 2008. d'artiste - Digital Painting 2, Ballistic Publishing.

FLATTMANN, A. 2007. The Art of Pastel Painting. Pelican Publishing.

LEVEQUE, R. 1992. Numerical Methods for Conservation Laws. Birkhäuser, Berlin.

KIM, B. M., LIU, Y., LLAMAS, I., AND ROSIGNAC, J. 2005. Flowfixer: Using BFECC for fluid simulation. In Eurographics Workshop on Natural Phenomena 2005, pp. 51-56.

OKAICHI, N., JOHAN, H., IMAGIRE, T., AND NISHITA, T. 2008. A virtual painting knife. Vis. Comput. 24, 7 (Jul. 2008), 753-763.

RUDOLF, D., MOULD, D., AND NEUFELD, E. 2005. A bidirectional deposition model of wax crayons. Computer Graphics Forum 24(1), pp. 27-39, Mar. 2005.

SMITH, A. R. 2001. Digital Paint Systems: An Anecdotal and Historical Overview. IEEE Ann. Hist. Comput. 23, 2 (Apr. 2001), pp. 4-30.

VAN HAEVRE, W., VAN LAERHOVEN, T., DI FIORE, F., AND VAN REETH, F. 2007. From Dust till Drawn: A Real-time Bidirectional Pastel Simulation. The Visual Computer, pp. 925-934, 2007.

VAN LAERHOVEN, T., AND VAN REETH, F. 2007. Brush up your painting skills: Realistic brush design for interactive painting applications. The Visual Computer, pp. 763-771, 2007.

WEISKOPF, D. 2004. Dye Advection without the Blur: A Level-Set Approach for Texture-Based Visualization of Unsteady Flow, Computer Graphics Forum 23(3), pp. 479-488, 2004.

XU, S., TAN, H., JIAO, X., LAU, F.C.M., AND PAN, Y. 2007. A generic pigment model for digital painting. Comput. Graph. Forum (EG 2007), Vol. 26, 2007.