Computers & Graphics 65 (2017) 60–72

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on Expressive 2016

Art-directed watercolor stylization of 3D animations in real-time

S.E. Montesdeoca [a,b,c,*], H.S. Seah [b], H.-M. Rall [c], D. Benvenuti [c]

[a] Multi-plAtform Game Innovation Centre, Interdisciplinary Graduate School, Nanyang Technological University, Singapore
[b] School of Computer Science and Engineering, Nanyang Technological University, Singapore
[c] School of Art, Design and Media, Nanyang Technological University, Singapore

ARTICLE INFO

Article history:
Received 29 October 2016
Revised 19 March 2017
Accepted 20 March 2017
Available online 30 March 2017

Keywords:
Watercolor
NPR
Expressive rendering
Real-time
Direct stylization
Object-space

ABSTRACT

This paper presents a direct stylization system to render 3D animated geometry as watercolor painted animation. Featuring low-level art-direction in real-time, our approach focuses on letting users paint custom stylization parameters in the 3D scene. These painted parameters drive watercolor effects in object-space, managing localized control and emulating the characteristic appearance of traditional watercolor. For this purpose, the parameters alter the object-space geometric representations and are rasterized, to coherently control and enhance further image-space effects. The watercolor effects are simulated through improved and novel algorithms to recreate hand tremors, pigment turbulence, color bleeding, edge darkening, paper distortion and granulation. All these represent essential characteristic effects of traditional watercolor. The proposed direct stylization system scales well with scene complexity, can be implemented in most rendering pipelines and can be adapted to simulate a wide range of watercolor looks. The simulation is compared to previous approaches and is evaluated through a user study, involving professional CG artists spending over 50 h stylizing their own assets and sharing their feedback about the watercolor stylization, the direct stylization system and their needs as artists using Non-photorealistic Rendering (NPR).

1. Introduction

The aesthetic quality of watercolor paintings and its characteristic effects make watercolor a unique traditional medium, which is extensively used for multiple coloring purposes. From early topological maps to fashion design illustrations and fine art – as Leslie Dutcher describes it: “watercolor paint has proven to be as elegant and academic as its counterpart [oil] while retaining its own ebullience and congeniality” [1]. However, while regarded as the traditional medium by itself is, it is rarely seen in animated form. The reason for this is arguably the accompanying difficulty to control the medium. Even acclaimed watercolor artists admit its capricious nature, which makes it difficult to master [2–4]. This peculiarity, together with the need for at least 10–12 frames per second to entail a sense of motion [5], makes traditional watercolor animation a complex and laborious endeavor to pursue.

Various systems to digitally recreate watercolor have been proposed, which facilitate the production of watercolor paintings and animations. These systems can be organized into three main categories according to their respective input data, namely: stroke-space, image-space and object-space. Each category has its own specialized application, however, they often tend to interrelate in order to enhance the final watercolor simulation. The system proposed in this paper is conceived to render 3D animated geometry as watercolor simulated animations, which falls into the third category, an object-space system. However, it contributes and differentiates itself from other object-space systems by two key elements.

First, the proposed system features a direct stylization pipe-line that is art-directed in object-space by the end user, who can paint the desired simulated effects with real-time performance. This enables immediate and coherent localized control over the watercolor look. And second, the proposed system extends and improves the palette of simulated watercolor effects. These simulated effects can be incorporated into most conventional rendering pipelines and feature the following:

- Object-space shader with a watercolor reflectance model, art-directed pigment turbulence, hand tremors and custom control features for further image-space filtering.
- Color bleeding through a hybrid solution involving object- and image-space processing (art-directed).
- Edge darkening with gradual pigment accumulation through a difference of Gaussian feature enhancement (art-directed).
- Surface distortion through UV displacement following surface normal inclinations (art-directed).

* Corresponding author.
E-mail addresses: santiago002@e.ntu.edu.sg, santymontesdeoca@gmail.com (S.E. Montesdeoca).

http://dx.doi.org/10.1016/j.cag.2017.03.002
0097-8493/

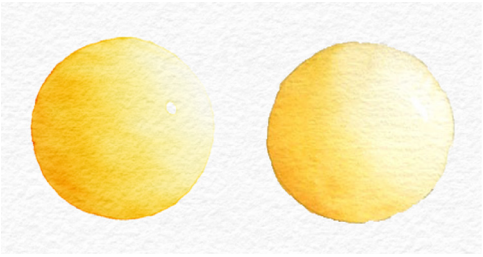


Fig. 1. Computer generated using our system (left) and hand-painted (right).

- Split surface granulation with dedicated control over paper roughness and pigment accumulation at the valleys of the paper (art-directed).

Rendered results can be found throughout the paper and an implementation in Autodesk Maya 2016 Ext. 2 for Windows is included in the Appendix A. For animated outcomes, please refer to the demonstration video. An abstracted pipeline schematic can be found in Fig. 4.

This paper embodies an extended journal version of the NPAR 2016 paper by Montesdeoca et al. [6]. Compared to the original version, the major additional contributions are threefold:

- A discussion on real-time art direction in object-space and the immediate benefits of direct stylization pipelines, compared to off-line decoupled stylization pipelines.
- A new approach towards color bleeding using a 4-dimensional joint-bilateral bleeding algorithm.
- A user study involving professional CG artists stylizing their own assets using our system, providing valuable feedback, insight on their specific needs and reinforcing this work.

In the remainder of this paper, related work will be introduced in Section 2. The proposed system with its individual contributions is presented in Section 3, the results are presented and discussed in Section 4 and evaluated in Section 5. The paper concludes with Section 6, proposing opportunities for future work.

2. Related work

Recreating traditional painting media has been one of the main challenges of non-photorealistic rendering since its inception. Remarkable examples came early on through physical approximation of the medium which analyzed and calculated the spread of pigment in simulated paper [7,8].

Stroke-space simulation systems especially benefited from accurately modeled physical simulations towards watercolor. Each brush-stroke can be intricately controlled to achieve most desired effects and a characteristic watercolor look [9–11]. Nonetheless, it can be difficult for an artist to understand and interpret the physical attributes, which these simulations offer. Alternatives can be found through procedural and example based approaches [12,13]. Stroke-space watercolor simulations provide extensive artistic control over digital paintings. However, in animation, it still presents itself as a complex and laborious alternative where each frame requires to be hand-painted.

For the proposed watercolor system, stroke-space simulations can be used to paint watercolor-looking textures, which are then mapped onto 3D objects. However, these textures would remain static and are prone to perspective and geometric distortion without further processing. Image-space and object-space simulation techniques are better suited for our approach and are presented next.

2.1. Image-space simulation

A traditional watercolor painting is a composition of a variety of complex interactions of pigment and water, layered together to create a body of work which interacts and speaks for itself with unique attributes, all chosen by the artist. Image-space simulation attempts to recreate this interaction and considers the image as a whole – finding and simulating characteristic effects and features which the artist might have chosen with the given subject.

Early attempts used image processing algorithms to automatically convert raster images to watercolor simulated paintings. Johan et al. [14] focused on recreating watercolor brush strokes by following a vector field created along the boundaries of the image elements. Bousseau et al. [15] focused instead on recreating the wet-on-dry (lavis) technique using a series of filters. Characteristic effects such as pigment turbulent accumulation, dispersion, edge darkening, paper surface structure and its distortion on the painted subject were considered. However, both of these methods process the image as a whole, limiting control of the outcome to variables and predefined rules. More recent work considers this limitation and involves computer vision algorithms to automatically locate the area which artists might emphasize [16,17]. The system proposed by Wang et al. [16] implements saliency detection to locate the areas of interest and stylize them differently. Additionally, the input image is colorized according to colors found on a training set of real watercolor paintings. The palette of simulated watercolor effects was also extended to include color bleeding found on wet-in-wet techniques and hand tremors, which are common in most watercolor paintings. However, the previously introduced image-space simulation systems rely heavily on image segmentation to extract detail from the input data. Segmentation is expensive to compute and over time introduces significant flickering issues, which are not a desirable effect in watercolor animations. Image-time-space simulation of watercolor has been proposed to address this issue by Bousseau et al. [18] but it still produces unnatural artifacts and is limited in the amount of effects it can simulate.

The use of neural networks to stylize images [19] is also gaining attention within image-space stylization approaches and watercolor simulation. They feature high-level control and infinite possibilities, depending on the trained neural network and desired source image-style. Through this, users can take advantage of an automated stylization, which can produce believable watercolorized results. Unfortunately, images stylized through neural networks are expensive to compute, difficult to art-direct and present unexpected and undesired artifacts, especially in animated form.

Finally, example-based approaches have been proposed to simulate watercolor in animation using image analogies. These image-space approaches have also been successfully applied in object-space rendered imagery, taking advantage from the additional object-space information for better visual results. Fišer et al. [20] proposed a system to simulate noise from static watercolor textures placed in image-space and object-space. Bénard et al. [21] implements image analogies to interpolate painted keyframes done by artists, with the help of velocity and orientation fields gathered from object-space data. This is probably the only image-space system that could embed watercolor stylization with art-direction, in which the user has localized control over the final look. However, the stylization will depend on the previous stroke-space watercolor simulation and textures are required to be painted every few frames. This makes the approach unsuitable for interactive applications where the point of view is not previously defined, such as games and virtual reality. Fišer et al. [22] takes advantage of image analogies to extend stylized shading from a single painted reference into a 3D model. Their system produces believable interactive results for simple and single imagery, but presents artifacts and temporal coherence problems when animated.

2.2. Object-space simulation

Object-space simulation approaches watercolor simulation and its characteristic effects in a more localized manner, as each 3D object can be individually defined and stylized. However, most object-space simulations are conjugated with image-space simulations once rendered, to enhance the outcome. The synergy between image-space and object-space approaches manages to create complex intertwined systems that can potentially offer extensive control and stylization possibilities.

The system proposed by Lum and Ma [23] was one of the first to simulate watercolor effects in object-space by incorporating pigment turbulence through the use of filtered noise along the curvature of the geometry. Burgess et al. [24] extends this work by simplifying the noise filter used for pigment turbulence. In addition, object identifiers and depth information were used to simulate darkened edges, together with additional noise to emulate edge distortion. However, results of these initial watercolor simulation systems offer limited control and resemblance to the traditional counterpart.

Lei and Chang [25] proposed a programmable vertex and fragment shader system which included a custom reflectance model based on a simulated color ramp. Image-space simulations, such as edge darkening through edge detection and paper effects to recreate granulation and distortion, were performed afterwards. Luft and Deussen [26] proposed a complex layered system where each object with a common identifier is rendered in multiple passes and each pass is processed by image-space simulations to later be composed into a final image – together with all other objects. Further particles are also attached to the geometry in the case of trees to emulate leaves. The system improves on art-directed control – as each object group gets simulated in image-space independently – and improves on edge darkening effects. However, the system does not scale well with scene complexity and still misses fundamental characteristic effects like pigment turbulence. Finally, Luft et al. [27] implemented and integrated watercolor stylization into a CAD software, proposing a tone-based lighting model and stylistic means of abstraction. The stylization relies extensively on spatial geometric relationship data, modulating their look through the use of an ambient occlusion pass. However, the system is not versatile and limited to a specific watercolorized look.

In general, existing object-space watercolor simulation approaches feature rudimentary art-direction and a limited reproduction of characteristic effects. The work presented in this paper contributes to the field by improving on these two key elements. A direct stylization pipeline is proposed, which enables immediate art-direction, scales well with scene complexity, extends the reproduction of characteristic watercolor effects and can be implemented in most rendering pipelines.

3. Real-time art-directed watercolor system

The aspiration of our system is to create a versatile and user-friendly object-space watercolor simulation tool, that not only focuses on modeling the characteristic effects of watercolor, but also considers real-time performance and art-direction as a necessary requirement. An immediate feedback is crucial as it could drastically reduce the required time for art-direction, facilitating and motivating the creative flow of the users.

Common stylization pipelines for film and television are based on a two-step process, see Fig. 2. First, the required source images are rendered off-line from a 3D application, including all necessary stylization intermediates such as custom masks and additional rasterized object-space data. These images are then gathered and processed in a compositing software to form the final stylized image. This decoupled creative process, while flexible and modular,

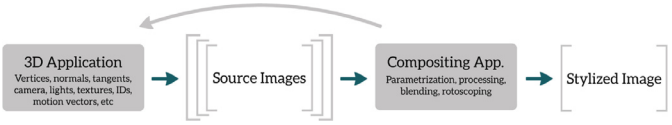


Fig. 2. Decoupled stylization pipeline.

slows down and even hinders art-direction. In stylized rendering, art-directed changes tend to occur often, but these changes can only be determined once the stylized image is seen and evaluated. If changes are required, some source images usually need to be altered, forcing to return up along the pipeline, back to the object-space application. In the 3D application, the changes are made and the new images are re-rendered and reconfigured back in compositing. Unfortunately, making stylization changes in the 3D application, without immediately viewing the end result, is not intuitive. This makes the process difficult and prone to several repetitions, until the final stylization is achieved. Additionally, the workflow at each stage involves different artists who might also have different visions and interpretations of the final desired stylization.

Taking inspiration from real-time technologies and game-engine rendering pipelines, we propose a more direct object-space stylization pipeline, which facilitates art-direction through immediate visual feedback, see Fig. 3. This pipeline, attached to a digital content creation software, enables the user to see the stylization results in real-time at every step – from asset creation, to the final art-directed tweaks of the resulting image. This dramatically increases the stylization workflow and helps artists focus directly on the desired result. The stylized image can be taken as is, however, the source images created internally can potentially be rendered off-line, taking advantage of ray traced fidelity. This way, the stylization is art-directed in real-time through the direct stylization pipeline, but the final off-line render can be processed through the decoupled stylization pipeline, enabling a compositing workflow with a feature-rich, art-directed and stylized image.

Following next, the direct watercolor stylization pipeline will be presented. First, our implementation on art-directed object-space control over the watercolor stylization is explained (Section 3.1). Then, the system portrayed in Fig. 4 is deconstructed and the modeled effects presented. The supported effects include hand tremors, a watercolor reflectance model, pigment turbulence, color bleeding (wet-in-wet), edge darkening and paper effects such as distortion and granulation.

3.1. Art-direction

Artists will usually not paint an entire painting with an even treatment. They will emphasize, abstract, and add custom effects locally, following their own style and expression. This is especially true with the aesthetic quality of watercolors, which permits a wide variety of expressions through their volatile effects. To control and drive these effects, low-level control over the stylization is needed. This control should also be embedded in object-space, so as to guarantee a coherent placement under camera motion. To this end, we rasterize parameter masks from object-space data. Masking is commonly used for image-space manipulation and compositing, but recent image-space stylization systems have also began adopting them as simulation drivers [28,29]. However, our mask is not created from painted pixels, but by vertices (points) in 3D space that are rasterized into masks. Alex Harvill [30] has also implemented this idea through a decoupled stylization pipeline, rendering a mask of weighted points to control a moving 2D illustration. From custom texture maps to point weights and vertex colors, these scalar field data can be used in the form of

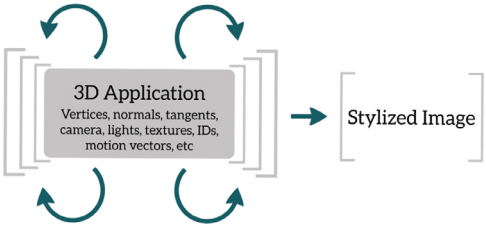


Fig. 3. Direct stylization pipeline.

intermediate render passes containing masking parameters for further image-space stylization.

In our system, we have chosen the use of vertex colors to create parameter masks for multiple reasons. They provide four parameter control channels (RGBA), are natively supported by vertex shaders (compute fast), can be animated for further temporal control and are supported by most 3D applications. For the implementation presented in this paper, we have assigned the control of effects to the following channels:

- Red - Controls paper distortion.
- Green - Controls paper granulation.
- Blue - Controls edge darkening and color bleeding.
- Alpha - Controls pigment turbulence and pigment fading.

Deconstructed examples of the art-directed influences and the dramatic change it has over the 3D rendered outcome can be seen in Figs. 5 and 6. Following the schematic in Fig. 4, these rendered images have undergone stylization through a direct stylization pipeline, which will be introduced next.

3.2. Controlled object-space effects

The first step in the direct stylization pipeline consists of processing the application data at the vertex shader stage. At this stage, vertices are usually just transformed from object space to their respective projection space. However, this stage can also serve to displace existing vertices to help future stylization. Our simulation involves two distinct vertex displacements, which aid in the recreation of the wet-in-wet technique and hand tremors.

Wet-in-wet and hand tremor deformation

The Wet-in-Wet technique, often referred in this paper as color bleeding, is a technique where pigment is applied to an already wet surface, to allow the color to spread (bleed) outside of its placement area. For this purpose, the vertices, which contain embedded bleeding parameters in their vertex colors, are translated along their normal vectors. This is done to pull the vertices outside of their original geometry, simulating the pigment bleeding outside its contained subject area.

Hand tremors are an essential characteristic of hand painted media as acknowledged by Wang et al. [16]. They are observable in the small irregularities found mostly on the edges of a painting and are caused by involuntary fluctuations in the human nervous system. These fluctuations can be simulated by minimally offsetting the vertices from their original positions. For best results, the vertices are displaced once they have been transformed to their projection space. This way, the offset will distribute evenly and will not be affected by any previous perspective projection. We chose a sinusoidal noise function as it can additionally simulate watery deformation, if desired.

$$V_o = sin(T \times s + V \times f) \times t \times P_p \quad (1)$$

The vertex offset V_o is modulated by the original vertex position V, time T and relative pixel size of the projection space P_p. Adjustment variables for speed s, frequency f and tremor amount t can be changed according to the desired tremor effect. However, hand tremors occur mostly in edges where precise control is required by the artist. Therefore, the vertex tremor V_t is modulated by the dot product of the view direction $\vec{V}$ and vertex normal $\vec{N}$.

$$V_t = V + V_o(1 - a(\vec{V} \cdot \vec{N})) \quad (2)$$

Once the vertices have been transformed and displaced per the desired stylization, they are sent down the pipeline towards the pixel shader stage, which, in our implementation, renders to multiple render targets (MRT), creating the color and control images.

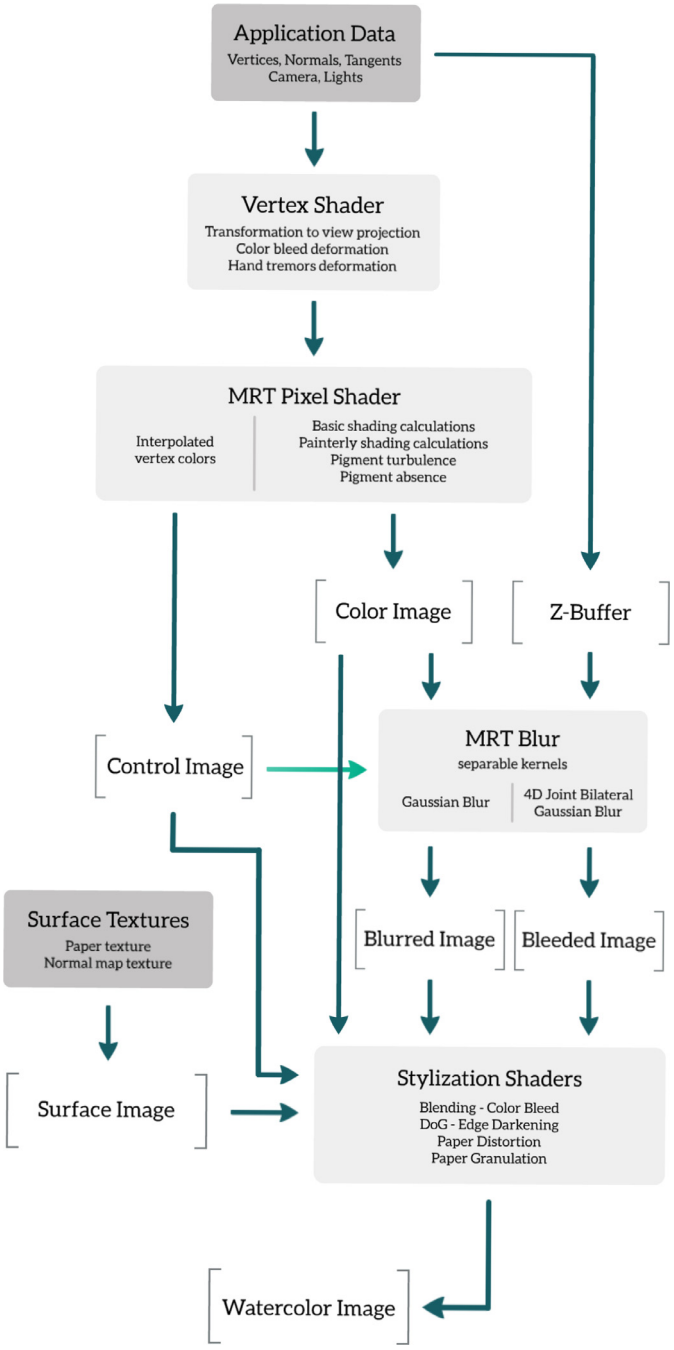


Fig. 4. System schematic of the rendering pipeline. Dark gray elements represent the input data, gray elements represent the different processing stages and the bracketed elements represent the computed images in the frame buffers.

Fig. 5. Breakdown of a watercolor stylized character. Henry © Oculus Story Studio.

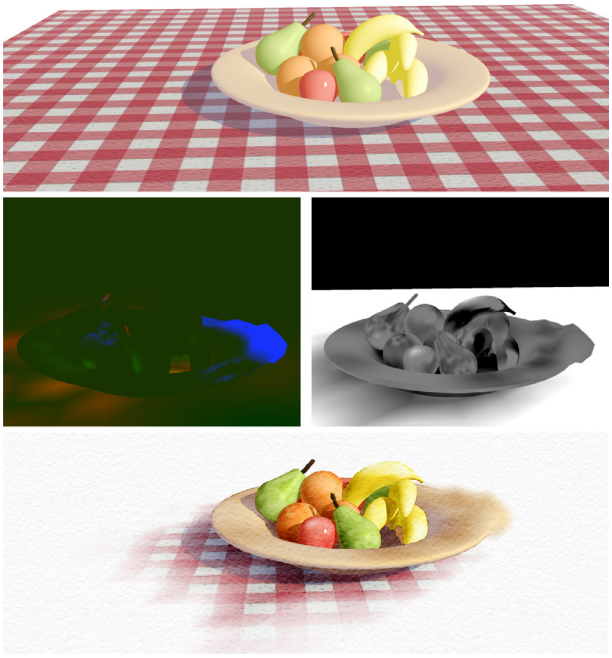


Fig. 6. Top to bottom: color image rendered without stylization; control image showing the rasterized control masks [vertex colors] (Left: RGB channels | Right: Alpha channel); art-directed watercolor simulation. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article).

Watercolor reflectance model and pigment turbulence

The characteristic translucency and turbulence found in watercolors play a pivotal role in the way motifs are colored, but is not found in common shading primitives. Lei and Chang [25] proposed the use of a one-dimensional color ramp to define reflected colors throughout the surface. However, this presents problems once the user intends to use texture maps. Therefore, we propose a custom watercolor reflectance model extending common shading primitives. The shading model features watercolor characteristics such as pigment dilution, cangiante illumination (change in the highlights to a brighter hue) and art-directed pigment turbulence.

To simulate dilution, we first calculate the area of effect D_A by modulating the dot product of the fragment normal $\vec{N}$ and the light direction $\vec{L}$, with a dilute area variable $d_A \in (0, 1]$.

$$D_A = \frac{\vec{L} \cdot \vec{N} + (d_A - 1)}{d_A} \tag{3}$$

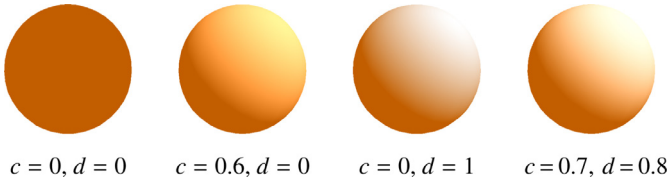


Fig. 7. Watercolor reflectance model with different cangiante and dilution attributes ($d_A = 1$; $C = (194, 94, 0)$).

With the computed area, the surface color may undergo the changes of a cangiante illumination, which shifts the hue of the surface away from a primary color. Through color theory in art and human perception, it is known that highlights do not necessarily shift perceptively towards white. The same rules apply to shadows which also do not necessarily shift perceptively towards black [31]. To achieve this, the cangiante color C_c is calculated by adding the dilute area multiplied by the cangiante variable c.

$$C_c = C + (D_A \times c) \tag{4}$$

Once the hue at the highlight has shifted, the dilution is performed by linearly interpolating the cangiante color towards the paper color C_p through the dilution variable d. Results of the shader with different dilution attributes are shown in Fig. 7 and the final interpolation is shown below.

$$C_d = d \times D_A(C_p - C_c) + C_c \tag{5}$$

Pigment turbulence is a low-frequency noise, which happens when the pigment settles unevenly on the flat surface of the paper, creating areas of varying pigment density. This effect is characteristic for watercolors – due to the fluidity of the water – and is incorporated within the object-space shader. Image-space simulations of pigment turbulence have been implemented before, however they are overlaid across the entire image. Therefore, it does not distinguish between different painted areas and present "shower-door" artifacts.

In our approach, we let the user art-direct the pigment density variations to suit the desired amount and pattern. This is achieved by letting the user directly paint the parameters on objects within the 3D application – *Ctrl* $\in [0, 1]$. The density parameters can also be generated by automatically assigning noise values into the vertex control channel. We find that this solution helps to give initial density variations, but art-direction is required to refine the result. An art-directed rendered example with different pigment densities can be seen in Fig. 8. The accumulation of pigments is simulated by altering the source colors through the color transmittance modification model, proposed by Bousseau et al. [15]. In this model,

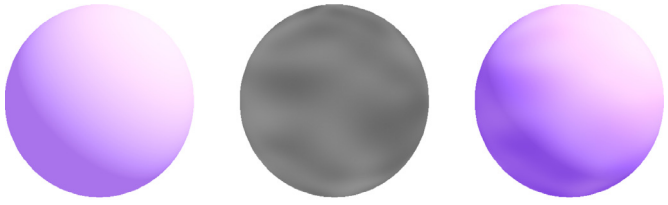


Fig. 8. Shader turbulence breakdown ($C = (169, 115, 235)$).

the observed color is the result of the exponential attenuation of reflected light, which has been transmitted through the density of the pigment. We additionally use the same parameters in the control channel to fade the rendered color C towards the paper color C_p. This is a common stylistic choice found in watercolor illustrations to attract attention to a desired subject. For this purpose, the painter avoids unnecessary scene elements by simply fading the color and leaving the paper as is. An example of this effect can be seen in Fig. 5 on the tablecloth. Both algorithms can be found in the equation below.

$$C_t = \begin{cases} C^{3-(Ctrl\times 4)} & \text{if } Ctrl < 0.5 \\ (Ctrl - 0.5) \times 2(C_p - C) + C & \text{if } Ctrl \geq 0.5 \end{cases} \quad (6)$$

To enhance the look of the shader, we support basic shading requirements such as the ability to use texture, normal and specular maps. Additionally, painterly reflectance attributes were also implemented, such as a custom diffuse shading factor, custom shade color, shade wrap, hard highlights and atmospheric fade. However, the deconstruction of these features have been omitted as they are not necessary characteristics of watercolors.

After the watercolor reflectance is calculated, the pixel shader shader also interpolates the vertex colors and renders to two different render targets. The watercolor reflectance is saved as a color image and the vertex colors are saved as a control image. These two images, stored in the frame buffers, are used extensively down the pipeline by image-space simulations, to obtain the final watercolor image.

3.3. Controlled image-space effects

Before entering the image-space stylization stage, three intermediate images need to be created. The first image involves the surface textures. At this stage of the pipeline, a paper texture and its respective normal map are loaded into a single image buffer. The red and green channels contain the surface inclinations from the normal map, while the blue channel stores a heightmap of the surface texture. The other two intermediate images are created at the MRT blur fragment shader stage. At this stage, two different blurring algorithms are run over the rendered color image, generating intermediates for edge darkening and color bleeding effects. To amplify the filtering on minimal computational cost, the resolution of these intermediates is half the size of the stylized output resolution, and linearly interpolated through a texture sampler at later stages. Further optimization includes the use of separable kernels.

The first blurring algorithm involves simple Gaussian blurring operations, convolving the equivalent of a 5×5 Gaussian kernel ($\sigma = 1$). The resulting blurred intermediate image is used to simulate edge darkening. The second blurring operation involves a 4-dimensional joint-bilateral filtering algorithm to simulate color bleeding, which is explained next.

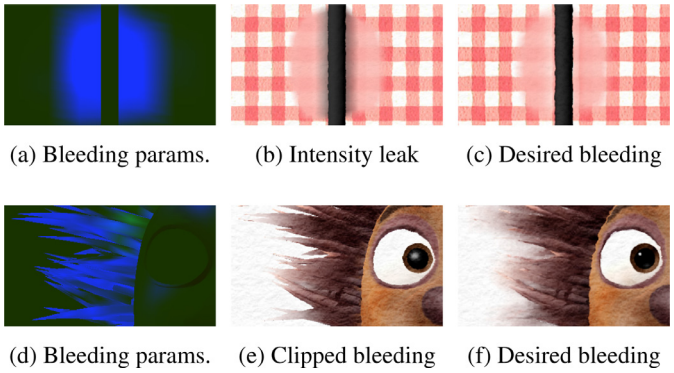


Fig. 9. Images (a) and (d) are the painted bleeding parameters; (b) and (e) are the results from the previous naïve bleeding approach; (c) and (f) are results using the new bleeding algorithm.

Color bleeding

At the beginning of Section 3.2, vertices were pulled outside their original geometry following the user painted bleeding parameters, altering the shape of the model. The approach implemented in the previous version of this paper [6] then convolved a 21x21 Gaussian kernel over the entire color image and blended/masked the resulting low-pass image with the color image – at the assigned bled sections (e.g. blue channel in Figs. 9a and d). While simple and straightforward to implement, this naïve approach has two fundamental problem cases: (1) when objects in the background are bleeding, but are obstructed by a non-bled object in the foreground and (2) when objects in the foreground are bleeding at their silhouettes over a non-bled background. These are illustrated in Fig. 9.

The first problem case presents itself as an intensity leak, which was caused as bled pixels were influenced by all neighboring colors, as shown in Fig. 9b. In this example, the black line at the *foreground* has not been given any bleeding parameters (see Fig. 9a) and should, therefore, not be considered when something at the *background* is bleeding (the bleeding is occluded). However, the previous naïve blurring approach considered the entire image equally and did not respect different bleeding parameters or depths. The second problem presents itself as clipped bleeding at the silhouettes, which was caused as the rasterized bleeding parameters abruptly mask the bled image, as shown in Fig. 9e. In this example, the spikes are bled according to the bleeding parameters (see Fig. 9d), but are constrained by the parameter edge (i.e. the bleeding does not flow from the spikes into the background color). This clipping would only be desired, if the bleeding mask was obstructed by a *non-bled* object at the *foreground*, as in the previously elaborated case. However, when the object at the *foreground* is the one bleeding, it should spread the color into objects found at the *background*. The previous naïve bleeding approach does not expand the parameter mask to account for this bleeding behavior. Figs. 9c and 9f show the desired outcome of the elaborated problems. These results have been rendered by the new color bleeding approach, which is featured next.

A 4D joint-bilateral filtering algorithm was derived to include and solve these problem cases, considering the depth and control values, in addition to the two spatial dimensions x and y. The bilateral filtering algorithm, coined by Tomasi and Manduchi [32], is therefore adapted to blur the image, while respecting depth and bleeding control differences/edges to solve the previously stated problem cases. To this purpose, two rules are followed:

- Bled pixels should not leak from un-bled foreground pixels.
- Bled pixels should bleed into/from un-bled background pixels.

This reasoning approaches a layered application of traditional watercolors, in which a wet-in-wet application of a foreground object will bleed into the background, whereas a wet-on-dry application of a foreground object should not bleed into a previously bled layer underneath. A pseudocode of the proposed algorithm is given in Algorithm 1. The algorithm can be further optimized for GPU computation to avoid the numerous conditionals, however, the conditionals were left for clarity.

Algorithm 1 Joint-bilateral color bleeding (two-pass).

Input: color image **I**, depth image **Z**, control image mask **C**, one dimensional Gaussian normalized kernel weights **W** ($size = 21$, $\sigma = 20$) and depth threshold **T**
Output: Bled image **B**, control image mask **C**

```
for all pixel I(x, y) do
    for i ∈ [-10 : 10] do
        // Check if source or destination pixels are bled
        if ((C(x, y) > 0) or (C(x + i, y) > 0) then
            // Initialize bleed and depth queries
            bleed = False
            sourceBehind = False
            // Check if source pixel is behind
            if (Z(x, y) − T) > Z(x + i, y) then
                sourceBehind = True
            // Check bleeding cases
            if C(x + i, y) > 0 and sourceBehind then
                bleed = True
            else if C(x, y) > 0 and not sourceBehind then
                bleed = True
            // Bleed colors
            if bleed then
                // Add weighted destination color
                B(x, y) = B(x, y) + I(x + i, y) × W[i + 10]
            else
                // Add weighted source color
                B(x, y) = B(x, y) + I(x, y) × W[i + 10]
        else
            // Add weighted source color
            B(x, y) = B(x, y) + I(x, y) × W[i + 10]
    //Expand control mask
    Modify C(x, y) as bled, if bleeding took place
```

Compute another pass with a vertical kernel using the modified control image mask, which includes the influences of previous horizontally bled pixels.

Once the bled intermediate image B has been calculated, all images/intermediates required for the proposed watercolor simulation are available and a series of stylization operations follow, which build up the watercolor simulated result. These operations take place at the stylization shader stage, as shown in Fig. 4. Color bleeding is the first effect to be included at the stylization stage by combining the previously bled intermediate image B into the color image I. For this purpose, the bled image B is up-sampled and blended according to the control mask $Ctrl$ – which was modified by the algorithm to include the surrounding bled pixels, when required.

$$I_{cb} = Ctrl(B - I) + I \tag{7}$$

The resulting color bled image I_{cb} simulates the pigment diffusion, which occurs when the colors bleed into each other. The effect and the bleeding parameters (blue control channel) can be clearly observed in Fig. 6. Further examples of color bleeding can be found integrated in Figs. 5, 11a, 11c, 11d–f, 12, 13a–d, 15d and 16.

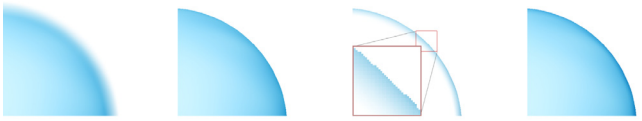


Fig. 10. Edge darkening simulation process ($C = (31, 171, 224)$). Note: The kernel in this example was increased for illustration purposes.

Edge darkening

Edge darkening is a characteristic effect that manifests itself when the pigment is carried out to the boundaries of a colored area, due to surface tension. This causes a higher concentration of pigments at the edges of the image and, therefore, a darker appearance. The edge darkening effect has been one of the most studied and implemented characteristics in watercolor. Early solutions in object-space involved analyzing nearby pixels for a difference in depth and object identifiers [24] and subtracting a blurred intensity from the alpha values of the object identifiers [26] – which in turn dilutes the main color. However, these models present a problem when texture information needs to be taken into account. Current solutions to simulate edge darkening in image-space involve edge detection algorithms such as the sobel filter [15,25]. However, edge darkening is not simply a darkened outline, which results from most edge detection algorithms. Edge darkening presents itself as a gradient darkening along the boundaries of the previously wet surface.

To model the gradient pigment accumulation found at the edges, we build upon the difference of Gaussians (DoG) feature enhancement algorithm. However, as the images do not contain any noise, only one low-pass filtered image is required. For this purpose, we take the previously calculated intermediate blurred image, found in the frame buffer. Additionally, in our algorithm, the image I is subtracted from the blurred image I_b. This deviation from the standard difference of Gaussians algorithm is done in order to avoid edge darkening at brighter areas of the image such as the paper color. Once the edges have been calculated, the maximum intensity value from the RGB channels is taken. To eliminate artifacts created by the DoG at color gradients, the edges are modulated through a *smoothstep* function. The resulting edges alter the bled image I_{cb} through the color transmittance modification model, darkening the color at the edges. A simple equation, without the *smoothstep* edge modulation, can be seen in Eq. (8). The simulation process from our model can be seen in Fig. 10.

$$I_{ed} = I_{cb}^{1+Ctrl \times max(I_b - I)} \tag{8}$$

Darkened edges do not occur evenly throughout the painting, therefore, the user is able to art-direct this effect by modifying the respective vertex color value $Ctrl$, represented in the parameterized control mask.

Paper distortion and granulation

Paper distortion occurs when the rough surface of the paper texture creates small distortions to the areas being painted. This happens through the interaction between the roughness of the surface and the hairs of the brushes or through surface inclinations which move the pigments towards the valleys of the paper. The latter cause is also directly correlated with the paper granulation effect. Paper granulation is the result of the higher concentration of pigments found at the valleys of the paper, which generate a darkened appearance. In order to simulate both of these effects, actual paper textures were used, as implemented by previous researchers [15,16,25,26].

In our system, paper distortion is effectively computed by directly sampling the color values at UVs that have been shifted

by the surface inclinations, available through the normal map of the paper texture. The amount of paper distortion can also be art-directed towards any desired result.

The final step in our pipeline involves the paper granulation effect. Our model is done in a way that permits individual control over the roughness of the paper and the accumulation of the pigments at the valleys of the papers. One of the main drawbacks of the color transmittance modification model is that once a channel is fully saturated, the color will not be darkened as $1^n = 1$ for $n \in \mathbb{R}$. Therefore, a split model is proposed which linearly interpolates between the color transmittance modification model and color subtraction, depending on the image saturation I.

$$I_g = I(I - P_{iv}) + (1 - I)I^{1+(Ctrl \times dP_{iv})} \quad (9)$$

The density amount variable d modulates the intensity of the inverted paper texture P_{iv}. This in turn is controlled by a *Ctrl* parameter to also art-direct the pigment density at the valleys of the paper. Once the geometry has been through all the changes and calculations from Section 3 onward, the final watercolor simulated image is presented to the user.

4. Results and discussion

To portray a successful watercolorized result through our approach, the individual stylization effects need to be directed together in a congruent fashion. For this purpose, the direct stylization pipeline (Fig. 3) presents itself as a creative catalyst. The artist has immediate feedback on stylistic changes and retains low-level control over each of the aforementioned watercolor effects. The real-time visualization also allows the user to effectively grasp and understand how to best combine these effects and achieve the desired result.

Various rendered results can be seen throughout Fig. 11. Additional rendered imagery, from professional CG artists which participated in a user study, can be found in Fig. 13. To observe the stylized results in animation, please refer to the accompanying video. An additional comparison to other object-space watercolor simulation systems can be observed in Fig. 15. To personally test the direct watercolor stylization system, an implementation, together with learning resources, are also provided in the Appendix A.

4.1. Versatility

The watercolor characteristics we have contributed and modeled extend the palette of watercolor effects found in object-space systems by introducing hand tremors, color bleeding and a painterly watercolor reflectance model – which takes advantage of the unique translucence and turbulence of the medium. Together with this contribution, we enhanced existing effects by modeling edge darkening with gradual pigment accumulation, proposed an efficient way to calculate paper distortion and enabled split control over paper granulation. All these effects are complemented by art-directed, localized object-space control using a direct stylization pipeline. Integrating all simulated watercolor effects with low-level control and immediate feedback into a stylization system enabled an unprecedented versatility and combination of effects, which can extensively control and transform the watercolorized rendered outcome.

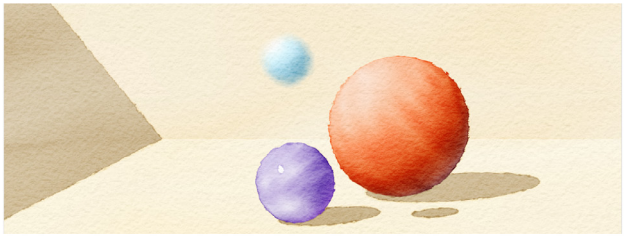
4.2. Photoreal watercolor

It is important to underline that, in the same way as traditional watercolors are used by acclaimed artists such as Anders Zorn and Steven Kozar to depict realism, the watercolor system can also be used to simulate watercolorized photorealistic depictions. This happens when using highly detailed, photorealistic textures,

(a) 3D watercolorized text

(b) Stylized character (gotita)

(c) A Patch of Heaven, model CC BY Sander Vander Meiren



(d) Spheres with different stylization parameters

(e) Stylized test scene

(f) Scarabs model © Abimael Gonzalez Lopez

Fig. 11. Rendered frames, captured from the Maya viewport.

(a) Specular shader (b) Watercolor shader

Fig. 12. Watercolorization example of a detailed surface texture.

as the system will not pre-filter them. Therefore, the level of abstraction and the degree of photo-realism in the watercolorized result depends mostly on the textures assigned by the user. This implementation supports a wider spectrum of watercolor imagery, but relies on the artist to source the amount of realism that is rendered. A simple, but clear example that illustrates how the system treats photorealistic textures is found in Fig. 12. In this comparison, both spheres are rendered with the same lighting conditions and normal/specular/color texture maps. However, the results are aesthetically quite different. The painterly watercolor reflectance model and its attributes, together with the art-directed effects, give the stylized example in Fig. 12b a watercolorized look. When observed in detail, the shade tint provides additional color variations, the edge darkening accumulates pigment and also adds granularity. Furthermore, the paper granulation and distortion gives a tangible property, whereas the color bleeding gives the result a manual touch. The rendered result, while realistic looking, possesses the qualities of the watercolor medium.

4.3. Implementation

Our system uses a direct stylization pipeline, implemented mostly in the GPU as an override to Viewport 2.0 within Autodesk Maya 2016 Ext. 2. By implementing the system in a widely used digital content creation software, users rapidly familiarized themselves with the new approach and were able to see immediate, art-directed results, at every step of the pipeline. The smooth learning curve was also confirmed by the unanimous response of the artists that participated in the user study. The stylization technology can further be used directly in their animated productions with their assets, improving their stylization workflow and creative decisions. This permitted the varied results created by the different CG artists, which can be seen in Fig. 13. The proposed watercolor simulation performs in real-time and scales well with scene complexity, as computation time is mostly resolution dependent. The maximum resolution is capped by the framebuffer of a graphics card. Modern GPUs support framebuffers of up to 16K × 16K (16.384 × 16.384) pixels, which amounts to more than 268 megapixel per image. Compared to the previous implementation, found in the conference version of this paper [6], there is a 1ms computation improvement, mostly achieved by using MRT shaders (shaders that write pixels to multiple render targets/buffers). Technical details and performance statistics for each test scene using our system can be seen in Table 1. These tests were conducted in Full HD Resolution (1920 x 1080) with the following hardware configuration: NVIDIA GeForce GTX Titan X, 2x Intel Xeon E5-2609 @2.40 and 16 GB RAM.

Table 1
Performance on different scenes (Full HD).

Figs.	Triangles	FPS (avg)
6	83,440	180
11d	26,088	210
11b	57,000	210
5	106,106	150
11e	347,680	90
16	38,324	125

5. Evaluation

Evaluating an expressive rendering approach is no straightforward task [33] and depends on the purpose it is being used for [34]. The evaluation of the system presented in this paper is especially challenging, as it involves low-level stylization control in object-space, with the purpose of obtaining aesthetically faithful and pleasing watercolor imagery. Therefore, a specialized skill-set is required of users to fully take advantage of the system's potential. These skills include object-space (3D) proficiency, proficiency within the software of implementation and experience/knowledge in the expressive visual style that is being sought after.

The nature of the stylization system and the work presented in this paper is better suited for a qualitative evaluation. Through a qualitative evaluation, we aim to gain a better understanding of the reception with its intended target users and the effectiveness of the combined effects to reproduce simulated watercolor imagery. To this end, two methods were used, which involve: (1) a user study of CG artists using the system by stylizing their own assets and (2) a comparison to previous object-space watercolor stylization approaches using similar assets. To conclude the evaluation, the limitations of our approach and system are formulated, which in turn present themselves as an interesting ground for future research.

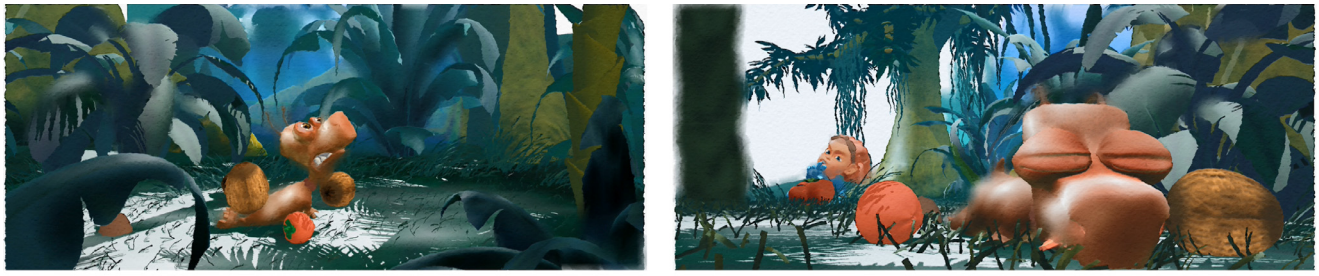
5.1. User study

The art-directed watercolor stylization system proposed in this paper is conceived to grant low-level stylization control to the end-user. As such, following the interaction spectrum of NPAR proposed by Tobias Isenberg [35], highly skilled artists are required to fully take advantage of its features and potential. This means that the quality of the aesthetic results does not only dependent on the algorithms and implementation, but is also highly influenced by the skills of the artist using it. An ideal candidate should possess proficiency in:

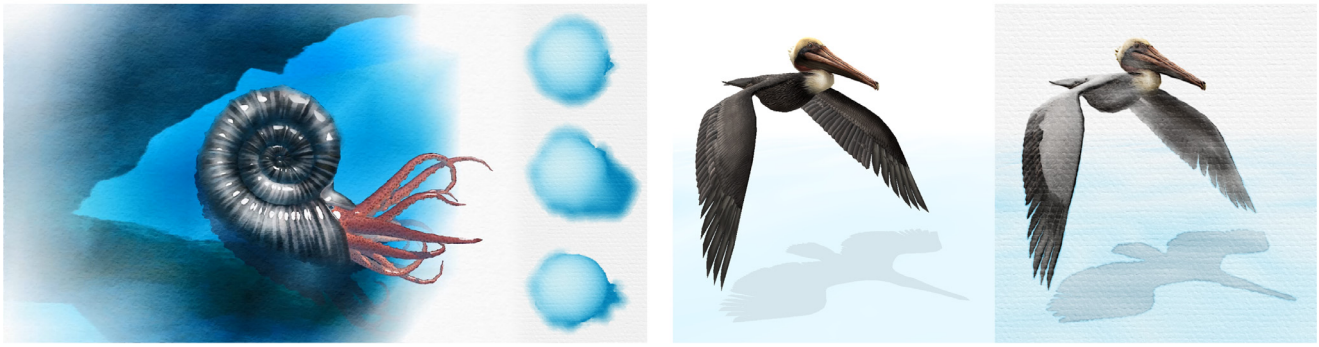
- Object-space (3D) creation and manipulation.
- The software of implementation.
- Rendering and composition.
- Watercolor aesthetics.

The skill-set is quite unique and involves both, technical and artistic knowledge. To this purpose, we approached professional CG artists with extensive experience in the animation industry to participate in our study. It can also be argued that traditional watercolor artists would be better candidates for the study. However, the complexity of working in object-space without previous 3D training and the possibility of testing the system with more varied assets, workflows and animation, supported our decision and resolve. A total of 7 CG artists from all around the world showed keen interest in using and providing feedback on the watercolor stylization system and 5 of them managed to participate in the user study, investing a total of over 51 h stylizing their own assets.

The conducted study first required the users to watch 15 min of instructional videos, explaining the different levels of control

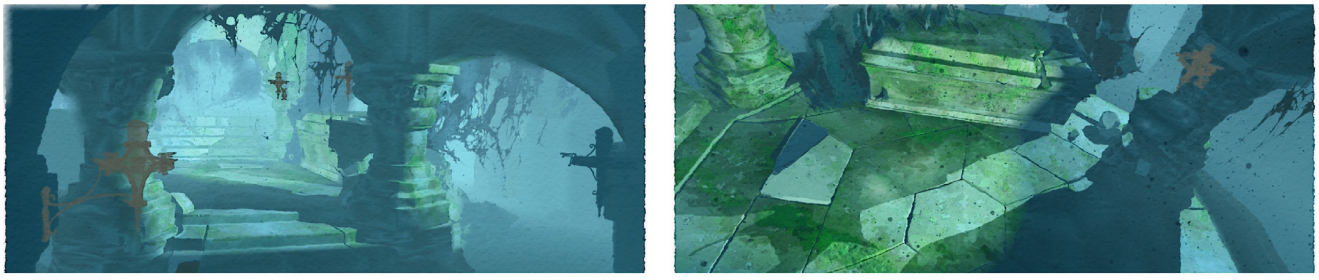


(a) Stylized by Joan and Miquel Cabot

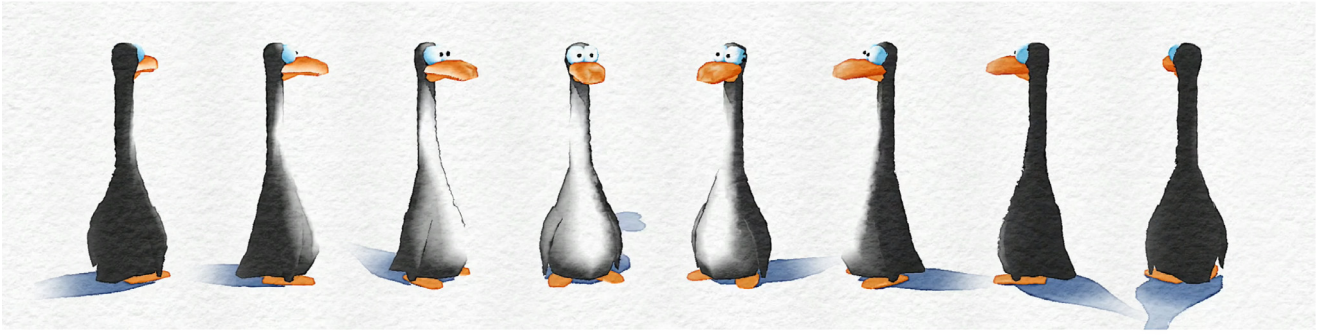


(b) Stylized by Eduardo Altamirano

(c) Stylized pelican comparison (photoreal - stylized)



(d) Stylized by Joan Cabot



(e) Stylized by Pierre Coffin

Fig. 13. Real-time viewport renders from different artists using the watercolor system. Animation clips of the stylization can be viewed in the accompanying video. Additionally, a stylized test scene is available with the provided implementation in Appendix A. The examples in (a), (b), (c) and (d) were created by participants of the conducted user study.

available in the direct stylization pipeline and the simulated watercolor effects. These tutorials were necessary to bring participants to a similar level of proficiency using the system and have also been included in the Appendix A. Following this, the participants were asked to spend over two hours stylizing their own assets. However, this time-frame did not stop their dedication, with some participants investing more than 15 h and providing us with their watercolorized imagery (featured in Fig. 13) and animated clips (featured in the accompanying video). To finalize the user study, the participants were asked to fill in a questionnaire with 24 questions, designed to understand and study the following:

- The reception of the watercolor stylization.
- The usefulness of a direct stylization pipeline.
- Their needs as artists for a successful watercolor stylization and a direct stylization pipeline.

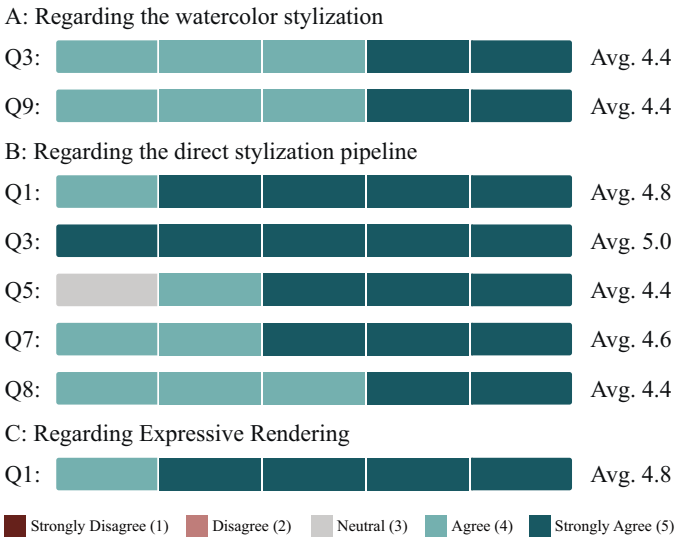


Fig. 14. Overview of Likert scale responses of the user study.

Regarding the watercolor stylization, only three of the participants had painted with watercolors before, but all participants agreed that the watercolor stylization in object-space satisfied their creative desires (see A-Q3 in Fig. 14). Some features/effects that participants would like to have improved are: the ability to locally control and embed the watercolor effects in the cast shadows, further improve on the edge darkening effect (incorporating custom widths and darker concentrations), improve the color-bleeding effect (incorporate a bleed direction and better temporal coherence), improve the object-space shader (supporting alpha transparency), an easier access to animate the painted parameters, custom paper colors and a way to export the different passes for final manual compositing. Additional new features/effects that they would like to have included are: object-space splatters, a more subtle solution to temporal coherence for effects influenced by the paper texture and subtractive color blending. None of the participants had experienced or seen object-space watercolor stylization before, but they all agreed that the control over the watercolor effects was enough for their stylization purposes (see A-Q9 in Fig. 14). Overall, the feedback over the watercolor stylization was quite positive. The users especially appreciated the way the color bleeding and edge distortion work, together with the ease of use, the amount of personalization, the simple unified tools, the painting of parameters and the immediate visual feedback.

Regarding the direct stylization pipeline, all of the participants strongly agreed on a Likert scale (see B-Q3 in Fig. 14) that the real-time stylization feedback benefited their creative decisions and aspirations. Additionally, to the most part, they also agreed that a direct stylization system is an improvement over previous methods of object-space stylization (see B-Q5 in Fig. 14). They all believe and agreed that a direct object-space stylization pipeline is much faster to work with and more artist friendly, compared to an offline/decoupled pipeline (see B-Q7 in Fig. 14). Especially because the results are given in real-time in object-space, giving the possibility to stylize and navigate the scene within a painted world. This offered them new stylization opportunities to capture details and shots that would probably have been missed, using a solely decoupled stylization pipeline. However, an option to render the different passes to a decoupled pipeline would offer further flexibility and custom final tweaks. The combination of both would be an ideal solution for three of the participants. Overall, the direct stylization pipeline had an outstanding reception, to the extent of having the participants agreeing that direct-stylization systems are the future of artist-friendly stylization tools (see B-Q8 in Fig. 14). This highlights the relevance of direct stylization pipelines in object-space and supports our previous claims over its benefits, compared to off-line/decoupled stylization pipelines. Some additional features/aspects that would facilitate the artist's workflow within the direct stylization pipeline are: the possibility of exporting/importing the stylization settings, having an option of texture map control (instead of vertex color control) for finer bleeding control, a tool to actually paint textures with traditional looking brushstrokes directly in the viewport, image-space stroke projection (is currently in object-space), together with some suggested improvements on the UI.

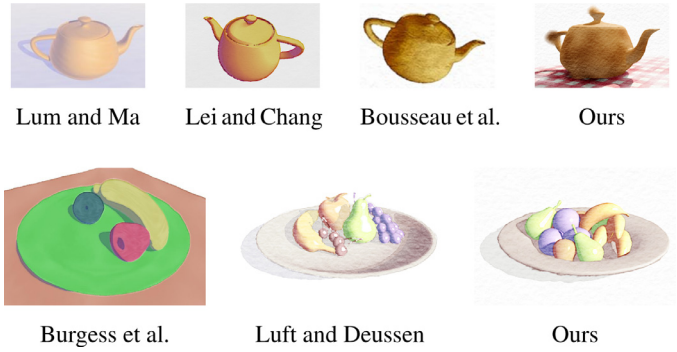


Fig. 15. Comparisons between watercolor simulation systems (magnify to see details).

Regarding Expressive Rendering (NPR) and the implementation of our system, all but one participant strongly agreed that the implementation in Autodesk Maya eased the learning curve (see B-Q1 in Fig. 14). Finally, four participants strongly agreed that they would start using expressive rendering stylization tools in their own projects, if these were to be available for them (see C-Q1 in Fig. 14). The entire questionnaire and all responses gathered from this user study can be found in the Appendix A.

5.2. Comparisons

Most of the characteristic watercolor effects available in object-space, studied by previous researchers, have been included in the system and improved upon. This allows our results to easily assimilate other outcomes and go beyond them, thanks to several unique contributions such as: object-space hand tremors, color-bleeding, a watercolor reflectance model and object-space pigment turbulence/fading – all bound and controlled together by the direct stylization pipeline. As it can be appreciated throughout the rendered imagery in this paper and the comparisons shown in Fig. 15, the results offer a significant improvement in versatility, thanks to its art-direction. This versatility results in an overall wider range of characteristic looks found in watercolors. Some previous attempts have focused only on recreating a specific style, but watercolor is a natural medium, which can be used to create artwork in a variety of different ways. The low-level, art-directed focus of our system allows to recreate this stylistic freedom, which is complemented through a direct stylization pipeline. In Fig. 15, we show previous approaches by other researchers and our results, using similar assets. The watercolorized image at the bottom, to the far right of the comparison is of special interest, as it was approximated to the results of Luft and Deussen [26], using our system. This stylization was conducted as an experiment to test how close an asset could be art-directed, to resemble the look of another approach. While the simulated effects vary and are implemented differently, the results do in general resemble each other. From here, our system can take full advantage of the

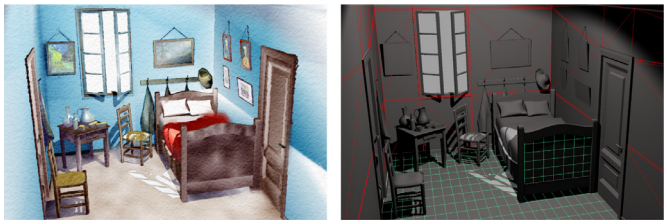


Fig. 16. Undesired shading lines on the wall, as varying turbulence control values are assigned in uneven topology (red wireframe). For a smooth interpolation of control values, even topology is recommended (green wireframe). Van Gogh Room model Ruslan Sokolovsky. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

art-directed stylization offered by our approach and produce more complex stylized results, as shown in Fig. 6.

5.3. Limitations

An inherent limitation of our implementation lies on the vertex density and topology of polygonal objects. As parameters are assigned per-vertex, a certain degree of polygonal resolution is required, to guarantee proper control. Topology artifacts may also appear when vertices with different control values are too close to each other and not evenly distributed, see Fig. 16. Mindful modeling and different levels of detail are suggested to overcome this limitation. Another alternative is to embed the control within object-space painted textures, as also suggested by one of the user study participants. This would allow for an arbitrary resolution of control, but would require properly unwrapped UVs for each geometric object. Finally, the use of Ptex to drive the stylization could overcome this limitation, at an additional performance cost.

The creation of a direct stylization pipeline for a specific style might require additional software development, compared to decoupled stylization pipelines. However, the immediacy of the system has proved to be of great advantage to artists who participated in the user study, potentially saving countless hours during production and augmenting their creative vision. However, art-directed systems are also propense to overwhelm with control, which might not necessarily be required. This in turn affects the performance and adds clutter to the control application. Proper training material and collaboration with artists is necessary to define their requirements and not overwhelm them with unnecessary options. Further user studies, including user behavior, could optimize any stylization system to address this limitation. There is still much to learn, as each stylization might require special considerations. The user study conducted for this paper involved only five professional CG artists. While the sample size was enough to get initial insight and feedback, more studies in the future might help us researchers to develop the technology that our intended target users want.

Finally, our system offers only a limited amount of watercolor effects compared to the traditional counterpart. In addition, some implemented effects might present small artifacts, as they have been optimized mainly for performance. The embedded control of effects at the surfaces of geometry are also prone to perspective distortion, producing potentially undesired effects such as scaling and foreshortening. Compared to previous approaches in object-space, a noticeable missing effect was gaps and overlaps, implemented by Luft and Deussen [27]. These were left out from our system, as the gaps and overlaps they simulated appear only at the silhouettes of rasterized layers, in which similarly grouped objects are found. In reality, gaps and overlaps may also happen within these layers, which are not taken into consideration in their approach. Their system is also substantially different to ours and does not scale well with scene complexity, which made a straightforward implementation impossible. Nonetheless, this effect does present itself often in watercolor imagery and, together with other additional characteristic watercolor effects, are open opportunities for future research, as are all previously stated limitations.

6. Conclusion and future work

We have presented an object-space watercolor simulation system, using a direct stylization pipeline, which facilitates art-directed watercolor rendered animation. The art-direction is based on intensity channels, which control an enhanced/extended palette of watercolor effects through rasterized masks. Our solution performs in real-time and can easily be adapted and implemented into most real-time rendering pipelines. We are able to produce an extensive variety of looks, outperforming previous systems, which rely heavily on global parametric values for watercolor stylization. Furthermore, we have evaluated our system by conducting a user study with professional CG artists. After stylizing their own assets, the artists were satisfied with the stylization versatility and highly praised the immediate feedback that it offered. Further valuable feedback and insight on their artistic needs for watercolor stylization was given and has been included. Our system manages to deal with the challenge of art-direction in object-space simulations and opens many new opportunities for future applications and research.

There are several ways in which the proposed watercolor simulation system can be improved and extended. The most important is to further extend the palette of simulated watercolor effects. Characteristic effects such as gaps and overlaps, dry-brush (dry-on-dry technique), the high frequency noise caused by layered patches of water and the outlining pencil strokes found sometimes in watercolor illustrations. Existing simulated effects are also open for improvements, which should be based on the artistic needs of NPR users. The conducted user study has given insight in this aspect and opens up many venues for future research and development – not only for watercolor, but on general stylization systems.

Traditional artists should also be considered in future NPR user studies. They might not be able to fully harvest the full potential of object-space stylization systems, but they might know better how their final rendered result should look like. These studies would bring us closer to fully satisfying the NPR Turing Test, proposed by David Salesin [36] and approaching a ground truth in expressive rendering.

As traditional watercolor is painted over a 2D surface and the control of the effects is placed on the surface of geometric objects, the 3D perspective will distort the rasterized control parameters, modifying the effects and potentially generating undesired results. Alternatives for avoiding unwanted perspective scaling of the parameterized control masks (i.e. pigment turbulence) and different ways to control and interact with simulated effects in object-space are valuable venues of future research in expressive rendering – together with efficient encoding of multiple stylization effects within object-space.

Finally, a watercolor stylization system could further benefit from dynamic paper simulation – when alternating paper textures are not desired. This feature could enhance frame-to-frame coherence and can be extended from previous work regarding dynamic canvas [37–39].

Acknowledgment

We would like to thank Bernhard Haux, Henriette Peter and Romain Vergne for their comments, advice and suggestions, together with the anonymous reviewers for their constructive feedback. Our immense gratitude is also with the participants in the user study:

Joan Cabot, Eduardo Altamirano, Miquel Cabot, Jason Labbe and an anonymous tester. Without their enthusiasm and dedication, we would not have such a variety of imagery and feedback. We would also like to thank all other artists that have shown their interest and tested the watercolor stylization system at different stages of development. Additional thanks to Oculus Story Studio for allowing us to stylize Henry and the modelers who have shared their scenes with us. This research is supported by the Interdisciplinary Graduate School (IGS) at the Nanyang Technological University, Singapore and the National Research Foundation, Prime Minister's Office, Singapore under its IDM Futures Funding Initiative.

Appendix A. Supplementary material

The supplementary material, provided in electronic form with this paper, includes the following:

- An accompanying video of the paper.
- An implementation of the direct watercolor stylization system for Autodesk Maya 2016 Ext. 2 with a test scene.
- The training material on how to install and use the direct watercolor stylization system, created for the user study.
- The questionnaire of the user study and the results from the participants.

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.cag.2017.03.002.

References

[1] Rim S, Dutcher L. Watercolor: paintings by contemporary artists. Chronicle Books; 2013.
[2] Ong KS. Mastering light & shade in watercolor. International Artist; 2003.
[3] Scheinberger F. Urban watercolor sketching: a guide to drawing, painting, and storytelling in color. Watson-Guptill Publications; 2014.
[4] Pollard JG. Watercolor unleashed: new directions for traditional painting techniques. North Light Books; 2013.
[5] Read P, Meyer M-P. Restoration of motion picture film. Butterworth-Heinemann Series in Conservation and Museology. Elsevier Science; 2000.
[6] Montesdeoca SE, Seah HS, Rall H-M. Art-directed watercolor rendered animation. In: Proceedings of the workshop on non-photorealistic animation and rendering NPAR'16. The Eurographics Association; 2016. p. 51–8. doi:10.2312/exp.20161063.
[7] Small D. Simulating watercolor by modeling diffusion, pigment, and paper fibers. Proc SPIE Image Handl Reprod Syst Integr 1991;1460:140–6. doi:10.1117/12.44417.
[8] Curtis CJ, Anderson SE, Seims JE, Fleischer KW, Salesin DH. Computer-generated watercolor. In: Proceedings of the 24th annual conference on computer graphics and interactive techniques (SIGGRAPH 1997). ACM Press/Addison-Wesley Publishing Co.; 1997. p. 421–30. doi:10.1145/258734.258896.
[9] You M, Jang T, Cha S, Kim J, Noh J. Realistic paint simulation based on fluidity, diffusion, and absorption. Comput Animat Virtual Worlds 2013;24(3–4):297–306. doi:10.1002/cav.1500.
[10] Van Laerhoven T, Van Reeth F. Real-time simulation of watery paint. Comput Animat Virtual Worlds 2005;16(3–4):429–39. doi:10.1002/cav.95.
[11] Chu NSH, Tai C-L. MoXi: Real-time ink dispersion in absorbent paper. ACM Trans. Graph. 2005;24(3):504–11. doi:10.1145/1073204.1073221.
[12] DiVerdi S, Krishnaswamy A, Měch R, Ito D. Painting with polygons: a procedural watercolor engine. IEEE Trans Vis Comput Graph 2013;19(5):723–35. doi:10.1109/TVCG.2012.295.
[13] Lu J, Barnes C, DiVerdi S, Finkelstein A. RealBrush: painting with examples of physical media. ACM Trans. Graph 2013;32(4). 117:1–117:12. doi: 10.1145/2461912.2461998.
[14] Johan H, Hashimoto R, Nishita T. Creating watercolor style images taking into account painting techniques. J Soc Art Sci 2004;3(4):207–15. doi:10.3756/artsci.3.207.
[15] Bousseau A, Kaplan M, Thollot J, Sillion FX. Interactive watercolor rendering with temporal coherence and abstraction. In: Proceedings of the 4th international symposium on non-photorealistic animation and rendering. ACM; 2006. p. 141–9. doi:10.1145/1124728.1124751.
[16] Wang M, Wang B, Fei Y, Qian K, Wang W, Chen J, et al. Towards photo watercolorization with artistic verisimilitude. IEEE Trans Vis Comput Graph 2014;20(10):1451–60. doi:10.1109/TVCG.2014.2303984.
[17] Lü C, Chen X. Image watercolorization based on visual weight-map. In: Proceedings of the 7th international congress on image and signal processing (CISP), 2014; 2014. p. 233–7. doi:10.1109/CISP.2014.7003783.
[18] Bousseau A, Neyret F, Thollot J, Salesin D. Video watercolorization using bidirectional texture advection. ACM Trans. Graph. 2007;26(3). doi:10.1145/1276377.1276507.
[19] Gatys LA, Ecker AS, Bethge M. A neural algorithm of artistic style. CoRR 2015. abs/1508.06576. URL http://arxiv.org/abs/1508.06576; [accessed: 2016-10-28].
[20] Fišer J, Lukáč M, Jamriška O, Čadík M, Gingold Y, Asente P, et al. Color me noisy: example-based rendering of hand-colored animations with temporal noise control. Comput Graph Forum 2014;33(4):1–10. doi:10.1111/cgf.12407.
[21] Bénard P, Cole F, Kass M, Mordatch I, Hegarty J, Senn MS, et al. Stylizing animation by example. ACM Trans. Graph. 2013;32(4). 119:1–119:12. doi:10.1145/2461912.2461929.
[22] Fišer J, Jamriška O, Lukáč M, Shechtman E, Asente P, Lu J, et al. StyLit: illumination-guided example-based stylization of 3D renderings. ACM Trans Graph 2016;35(4):92-1–92-11. doi:10.1145/2897824.2925948.
[23] Lum EB, Ma K-L. Non-photorealistic rendering using watercolor inspired textures and illumination. In: Proceedings of the 9th pacific conference on computer graphics and applications; 2001. p. 322–30. doi:10.1109/PCCGA.2001.962888.
[24] Burgess J, Wyvill G, King SA. A system for real-time watercolour rendering. In: Computer Graphics International. IEEE; 2005. p. 234–40. doi:10.1109/CGI.2005.1500426.
[25] Lei SIE, Chang C-F. Real-time rendering of watercolor effects for virtual environments. Berlin Heidelberg: Springer; 2005. p. 474–81. doi:10.1007/978-3-540-30543-9_60.
[26] Luft T, Deussen O. Real-time watercolor for animation. J Comput Sci Technol 2006;21(2):159–65. doi:10.1007/s11390-006-0159-9.
[27] Luft T, Kobs F, Zinser W, Deussen O. Watercolor illustrations of CAD data. In: Computational aesthetics in graphics, visualization, and imaging. The Eurographics Association; 2008.
[28] Semmo A, Limberger D, Kyprianidis JE, Döllner J. Image stylization by interactive oil paint filtering. Comput Graph 2016;55:157–71. http://dx.doi.org/10.1016/j.cag.2015.12.001.
[29] Semmo A, Dürschmid T, Trapp M, Klingbeil M, Döllner J, Pasewaldt S. Interactive image filtering with multiple levels-of-control on mobile devices. In: Proceedings of the ACM SIGGRAPH Asia symposium on mobile graphics and interactive applications, NA. ACM; 2016.
[30] Harvill A. Effective toon-style rendering control using scalar fields. Memo; 2007. URL: http://graphics.pixar.com/library/ToonRendering/index.html; [accessed: 2016-10-28].
[31] Birren F. Principles of color: a review of past tradition and modern theories. Schiffer Publishing; 1987.
[32] Tomasi C, Manduchi R. Bilateral filtering for gray and color images. In: Proceedings of the sixth international conference on computer vision. Washington, DC, USA: IEEE Computer Society; 1998. p. 839.
[33] Hall P, Lehmann A-S. Don't measure-appreciate! NPR seen through the prism of art history. In: Image and video-based artistic stylisation, 42. Springer-Verlag; 2013. p. 333–51. doi:10.1007/978-1-4471-4519-6. book 16.
[34] Isenberg T. Evaluating and validating non-photorealistic and illustrative rendering. London: Springer London; 2013. p. 311–31. doi:10.1007/978-1-4471-4519-6_15. chap. 15.
[35] Isenberg T. Interactive NPAR: what type of tools should we create?. In: Proceedings of the workshop on non-photorealistic animation and rendering NPAR'16. The Eurographics Association; 2016. p. 89–96. doi:10.2312/exp.20161067.
[36] Salesin D.H.. Non-photorealistic animation & rendering: 7 grand challenges. Proceedings of the keynote talk at second international symposium on non-photorealistic animation and rendering (NPAR 2002, Annecy, France, June 3–5, 2002); 2002.
[37] Cunzi M, Thollot J, Paris S, Debunne G, Gascuel J-D, Durand F. Dynamic canvas for immersive non-photorealistic walkthroughs. In: Proceedings graphics interface. A K Peters, LTD.; 2003. p. 121–30.
[38] Kaplan M, Cohen E. A generative model for dynamic canvas motion. In: Proceedings of the first eurographics conference on computational aesthetics in graphics, visualization and imaging. Eurographics Association; 2005. p. 49–56. doi:10.2312/compaesth/compaesth05/049-056.
[39] Bénard P, Bousseau A, Thollot J. Dynamic solid textures for real-time coherent stylization. In: Proceedings of the 2009 symposium on interactive 3D graphics and games. ACM; 2009. p. 121–7. doi:10.1145/1507149.1507169.