Vignette: Interactive Texture Design and Manipulation with Freeform Gestures for Pen-and-Ink Illustration

Rubaiat Habib Kazi[1,2], Takeo Igarashi[1,3], Shengdong Zhao[2], Richard C. Davis[4]

[1]JST ERATO, IGARASHI Design Interface Project [2]CS, National University of Singapore
[3]Information Systems, University of Tokyo [4]SIS, Singapore Management University
{rubaiat, zhaosd}@comp.nus.edu.sg, takeo@acm.org, rcdavis@smu.edu.sg

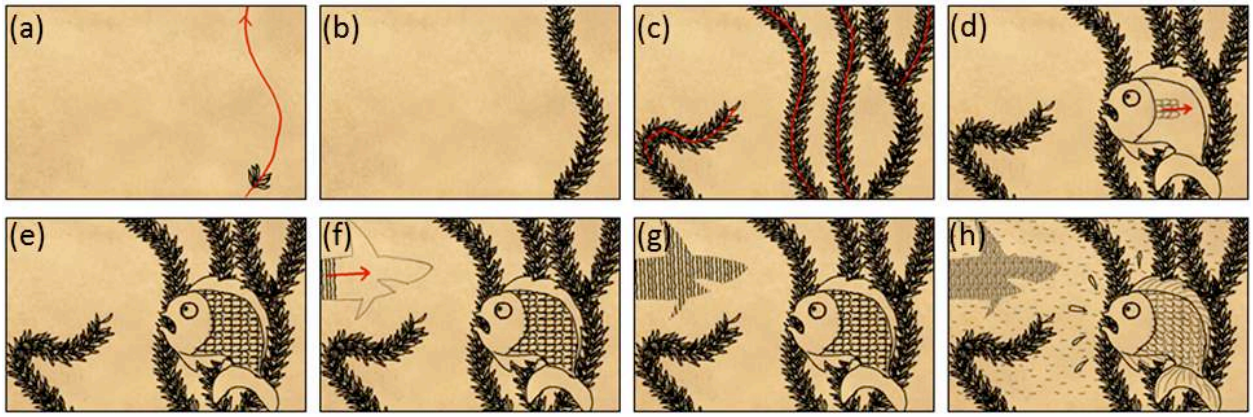

Figure 1: The steps of a pen-and-ink illustration with Vignette from scratch (a) Draw leaf strokes (black) and gesture (red) (b) Texture created from gesture and strokes(c) More textures (d) Draw scale strokes and gesture (e) Region filed with scales (f) Draw hatching strokes and gesture (g) Fill region with hatching (h) Final illustration created in minutes.

ABSTRACT

Vignette is an interactive system that facilitates texture creation in pen-and-ink illustrations. Unlike existing systems, Vignette preserves illustrators' workflow and style: users draw a fraction of a texture and use gestures to automatically fill regions with the texture. We currently support both 1D and 2D synthesis with stitching. Our system also has interactive refinement and editing capabilities to provide a higher level texture control, which helps artists achieve their desired vision. A user study with professional artists shows that Vignette makes the process of illustration more enjoyable and that first time users can create rich textures from scratch within minutes.

Author Keywords

Pen and ink illustration, stroke synthesis, creativity

ACM Classification Keywords: H.5.2 [Information Interfaces And Presentation]: User Interfaces - Interaction styles;

INTRODUCTION

Pen and ink illustration is a popular artistic medium that can be seen in textbooks, repair manuals, advertisements, comics, and many other printed and digital media. Illustrations typically incorporate a wealth of textures, tones and artistic styles. These effects take significant amounts of skill, artistry, and *patience* to create.

Many research systems [2, 3, 21, 22, 30, 31] can render scenes in the style of pen-and-ink illustrations. Also, professional tools like Illustrator, Photoshop, Comic Studio and InkScape can synthesize customized textures. These tools are powerful and widely used, but they fall short of preserving two key properties of traditional paper-based pen-and-ink illustrations.

The first key property is artists' rich personal style, as seen in Figure 2(a). Arthur L. Guptill explains, "*... the more conventional the art, the greater the opportunities for originality. We might go so far as to say that there is perhaps no medium offering one a better chance for development of a personal technique than the pen, for pen drawing is akin to handwriting, and just as no two people write alike, so no two people draw alike...*" [7].

Tools developed for pen-and-ink style renderings [2, 3, 21, 22, 30, 31] require some kind of 3D models or 2D images to serve as the template for guiding the generation of textures. As a result, these tools can create high quality pen-and-ink style drawings, but there is little room for variation

and artistic styles. Similarly, most tools for 2D texture generation and manipulation lack the natural feel of pen-and-ink drawing (Fig. 2(b)).

The second key property that existing tools fail to preserve is the workflow of pen and ink illustration. Generating a drawing from 3D scenes or images destroys this workflow completely. Texture generation tools do use artists' pen strokes, but much of the creation process with these tools is devoted to parameter tweaking. These tools can produce diverse effects, but they are often difficult to learn, and tedious to apply.

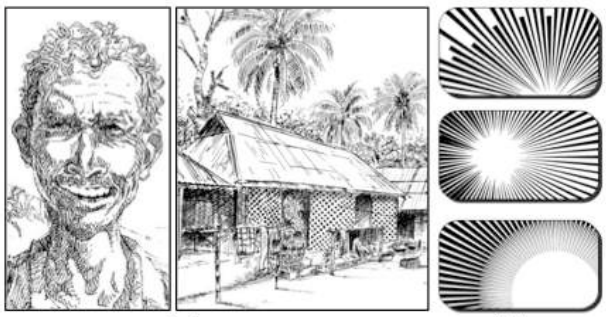

Figure 2: a) Illustrations made by hand have distinctive styles b) Illustrations made in comic studio have a mechanical look.

We present Vignette, an interactive system for pen-and-ink illustrations that uses free-form gestures for texture design and manipulation. Vignette provides tools to design, arrange, and manipulate pen-and-ink illustration textures through simple gestures. Vignette preserves traditional pen-and-ink illustration workflow while accelerating the creation of textures from user defined example strokes. We focus on drawing from scratch, using textures generated entirely from artists' hand-drawn strokes to preserve the original style and signature of individual artists.

In our system, the user draws a small fragment of the target texture, specifies the type of texture by choosing a tool, and gestures to define the growth of the texture (Fig. 1(a), 1(d), 1(f)). The system completes the texture, preserving the style of the example strokes (Fig. 1(b), 1(e), 1(g)). The user then interactively refines the textures, tones, perspective view, sweep and orientation of the texture to achieve desired results (Fig. 1(h)). Using Vignette, even first-time users can create complex and expressive illustrations within minutes.

This paper presents the following contributions:

- An analysis of traditional pen-and-ink illustration workflow and artifacts that guides interface design.
- The Vignette system, which facilitates texture creation while preserving this workflow.
- An evaluation with four artists that shows how Vignette reduces the tedium of texture creation.

After reviewing related work, we present our analysis of the traditional pen and ink illustration process and categorize the textures used by artists. We then describe the interface of Vignette, which is based on this analysis, and follow this with Vignette implementation details. Finally, we present an evaluation of our system with 4 professional artists.

RELATED WORK

The methodology we describe here builds on previous work for pen-and-ink illustration rendering, texture synthesis, and design of digital tools inspired by traditional approaches to creating artifacts. We discuss representative examples of previous work in these areas below.

Pen-and-Ink Rendering Systems

A number of systems render illustrations in a pen-and-ink style. Some are geometry-based [21, 30, 31], taking 3D scene descriptions as input, while others take 2D images as input [3, 22]. The tones and textures in these systems are therefore guided by the underlying 3D geometry or 2D image. Instead, we focus on workflows that allow illustrators to produce artworks from scratch, where no scene model or image exists. Our system analyzes reference patterns and gestures drawn by artists to synthesize new patterns with similar perceptual properties.

Commercial Drawing Applications

Applications like Adobe Photoshop, Adobe Illustrator, Comic Studio, Sketchbook Pro, InkScape, and CorelDRAW have become mainstays of digital artwork creation. In pixel-based applications like Photoshop, duplicating an example patch in multiple layers or using pattern brushes can speed up some repetitive tasks that illustrators encounter [16]. With pixel-based approaches, however, it is hard to control density, add variation, or deform textures.

Vector graphics editors like Adobe Illustrator and Comic Studio are very powerful but awkward for illustration. These tools allow artists to define custom textures that can be controlled by a set of parameters, but the resulting textures tend to lack the subtle variations found in traditional illustration. Furthermore, tweaking the many parameters to get a desired texture is tedious and shifts attention away from the artwork itself. These tools are oriented more toward graphic design than pen-and-ink illustration.

Bridging the Gap between Physical and Digital Art

In recent years, researchers have explored digital art systems inspired by traditional artistic media and workflows. SandCanvas[13], Project Gustav[18], Fluid Paint[26], Intu Paint[27] and IO Brush[20] all preserve traditional workflows while bridging the gap between physical and digital drawing artifacts. With Vignette, we have explored natural sketch based interaction for texture design and manipulation that preserves illustrators' traditional workflow.

Texture Synthesis

One way to preserve personal artistic style is to create larger textures from user drawn examples. Texture synthesis methods synthesize new textures from texture samples in such a way that, when perceived by a human

observer, they appear to be generated by the same underlying process. The idea of synthesizing textures, both for 2D images and 3D surfaces, has been extensively addressed in recent years (see a survey of this type of work in [29]). However, the basic representations in most existing texture synthesis methods such as pixels [11, 28], vertices [25], voxels [14] or parametric descriptors [32] cannot adequately represent individual or discrete elements with semantic meanings. Moreover, subtle variation in the reproduced pattern is desirable for changing density and avoiding regularity. It is difficult to achieve such variation with pixel-based texture synthesis.

The use of vector-based descriptions of an input pattern for synthesis is explored in [1, 2, 6, 8, 11]. These descriptions are more expressive and allow higher-level analysis than pixel-based approaches. However, [1, 2, 8] do not reproduce the interrelation of strokes within a pattern, and are thus limited to hatching and 1D synthesis only. Ijiri et al. [11] presented a method for synthesizing 2D elements by locally growing a 1D ring of elements in a neighborhood around an example. Their examples are points, not strokes, which limits the user to synthesizing dot patterns.

Barla at al. present a synthesis technique [6] that can automatically generate stroke patterns based on a user-specified reference stroke pattern. This is an extension of texture synthesis techniques to vector-based patterns. However, both Barla et al and Ijiri et al use triangulation to perform 2D synthesis. This approach cannot handle elements with complex shapes that are closely correlated with spatial distributions. Instead, we use a data-driven texture optimization method [15] for stroke synthesis.

Vignette provides a novel way to design and manipulate textures for pen-and-ink illustrations completely from scratch. We integrated texture synthesis methods with free-form gestures to provide powerful texture tools that help artists create beautiful artworks.

ELEMENTS OF PEN-AND-INK ILLUSTRATION

In this section, we review principles of pen-and-ink illustrations and introduce some terminologies.

Strokes

Strokes (Fig. 3(a)) are the building blocks of textures. For centuries of pen and ink illustrations, artists have infused drawings with their signature styles through careful use of individual strokes. Strokes become textures when drawn in groups. (Fig. 3).

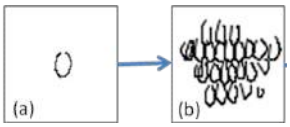

Figure 3: (a) Individual strokes (b) combine to form textures.

Textures

A texture is a collection of strokes that gives an object or scene the illusion of shape, surface properties, and lighting. In a texture, individual strokes are not of critical importance, but collectively they can clearly indicate the difference between textures like smooth glass and old knotted wood (see Figure 4(a)-4(f)).

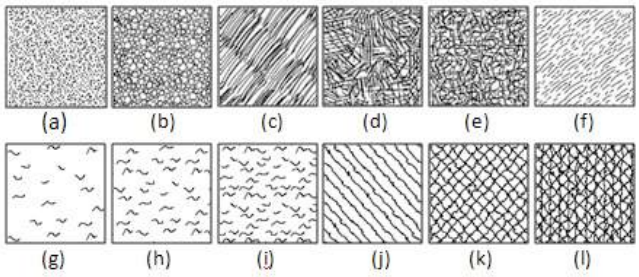

Figure 4: (a)-(f) Different kinds of textures. Variation of tones by (g)-(i) changing the density of strokes (j)-(l) subsequent cross-hatching

Tones

Tone (also known as "value" or "density) refers to the density of strokes in a texture. The tone is the ratio of black ink to white paper over a given region of the texture. Figure 4(g)-4(i) and 4(j)-4(l) shows the variation of tones with the same texture to indicate the brightness of a surface.

Together, stroke, texture, and tone provide artists with a rich language for producing expressive illustrations with a variety of personal styles [7].

PEN-AND-INK ILLUSTRATION ANALYSIS

In this section, we examine the process of creating pen-and-ink illustrations. We also and analyzed the textures in 56 illustrations to identify opportunities for automation. Our findings can be used to guide the design of pen-and-ink illustration systems.

Traditional Illustration Workflow

While the process of pen-and-ink illustration can vary from artist to artist or even between one artists' illustrations, the illustration process usually follows five steps [19]:

Step 1: create *outlines* of simple geometric shapes and regions of interest with a pencil. The drawing at this step is typically light and erasable.

Step 2: pencil in details and shadows. Iterate until the outline and any object highlights are well-defined.

Step 3: begin filling in the detailed textures, starting with small areas of example texture. We call these small example textures *patches*.

Step 4: repeatedly apply patches to fill in the outlines.

Step 5: add or modify details to complete the illustration.

Steps 1 and 2 determine the high-level structure of the illustration with shape outlines, their spatial layout, and indicators of properties such as shadows and highlights. Step 3 determines the detailed textures and tones of the shapes or regions determined in step 1 and 2. Essentially, these three steps contain most of the essential elements to uniquely define the style and content of an illustration.

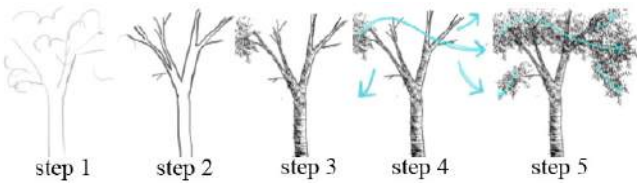

Figure 5: The steps in traditional pen-and-ink illustration.

In step 4, the artist repeatedly applies the various textures and tones to all shapes and regions in the illustration. This fourth step is the most tedious and contributes the least to the uniqueness of a pen-and-ink illustration. However, the illustration cannot be completed without it.

In the last step, the artist touches up the illustration with final details and adjustments. Note also that artists often iterate this process and jump between steps.

Analyzing this workflow, we found that a major component of an artist's personal drawing style lies in her procedure (or muscle) memory of using the pen, which is reflected in her strokes [19]. Using a third party image or model cannot preserve this unique style. If the artist were asked to produce the texture separately, saving it into an image before applying it to the drawing, it would break the creative flow of the drawing process. Therefore, we believe it is important to allow the artist to define both the outline and example textures from scratch using her hands.

Finally, we note that step 4 in the traditional workflow is the most repetitive and time consuming. Consequently, it is quite suitable for automation.

Texture Automation Techniques

To inform the design of systems that automate *step-4* of the traditional illustration workflow, we examined the kinds of textures that professional pen and ink illustrators use. We analyzed 56 rich pen and ink illustrations by 32 artists, mostly taken from *The Technical Pen* [24] and *Rendering with Pen and Ink* [7]. After analyzing the textures in these illustrations, we classified them according to techniques artists could use to automate the filling-in process. We identified three techniques: *brushing*, *flood filling,* and *continuous hatching*.

As we explained in Related Work, brushing and flood filling techniques exist in current graphical tools, but they are tedious, awkward, and do not preserve artists' style. Continuous hatching cannot be found in these tools at all. Vignette provides all three techniques, and it uses texture synthesis of vector geometry to produce pleasing results that preserve artists' style.

It should be noted that these techniques cannot reproduce all textures effectively. Automation requires textures to be repetitive so that a computer can synthesize them from example patches. Some textures have so much variation that they cannot be synthesized from patches.

In the following paragraphs, we describe these texture filling techniques along with applications and variations.

Flood Filling

A small set of discrete strokes can often be used to fill up a region. Flood Filling can also be done in a particular orientation to follow the contour of the volume or shape (Fig. 6(d)). We identified this effect in 25 out of 56 illustrations. Applications of flood fill include stippling (where tone and textures are applied with small dots and strokes (Fig. 4(a)), clothes textures, walls, illustrations, wood, landscape etc.

Brushing

In these textures strokes are augmented along a line, rather than filling up a 2D region (Fig. 6(a), 6(b))**.** In our analysis, this type of synthesis was more common than the other two (37 out of 56 illustrations). These textures are commonly applied to create a wide range of effects including hatched (Fig. 4(j)) and cross-hatched lines (Fig. 4(k & l)), landscape drawings for trees and grasses (Fig. 6(b), and many other complex textures.

Continuous Hatching

Continuous hatching is a set of closely spaced parallel lines from one edge of a shape to another, similar to symbolic indication of a cross section in an engineering blueprint (Fig. 6(f)) creating tonal and shading effect. Continuous hatching is different from brushed hatching, because the synthesis is two dimensional, i.e. it fills up a 2D region instead of extending along a 1D line. However, unlike the discrete elements in flood-fill, the lines are connected with each other to create longer lines and fill up a region (Fig. 6(e), 6(f)). Prevalent application of continuous lines includes architectural drawings, cross-hatching, and portraying the illusion of depth, by providing different darkness to different planes (layers) of drawing (Fig. 6(e)).

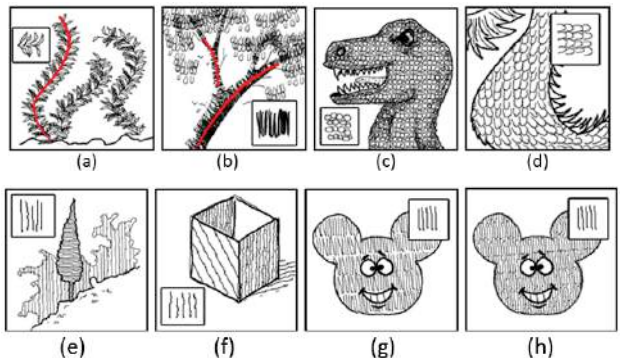

Figure 6: Applications of texture automation techniques. (a-b) Brushing. (c,d) Flood Filling. (e-f) Continuous Hatching. (g) Flood fill from the example patch (inset). (h) Continuous hatching from the same patch (inset), in which discrete strokes are uniformly stitched together.

VIGNETTE: INTERFACE AND INTERACTION

Our analysis of traditional pen-and-ink illustration processes and artifacts helped us to build Vignette, a texture synthesis system that is based on the traditional illustration workflow. Here we present Vignette's user interface. We begin with Vignette's toolbars and palettes, then describe our workflow, and close with interactive refinement tools.

Vignette's Toolbars and Palettes

As shown in Figure 7, Vignette has four toolbars located around a central drawing canvas. These are the *main toolbar* (Fig. 7(a)) for drawing and texturing tools; a *file/edit toolbar* (Fig. 7(b)) for common commands; *patch toolbar* (Fig. 7(c)) for adding, updating, importing and deleting patches; and a *background palette* (Fig. 7(d)).

The top left region of the drawing canvas is reserved for a palette of patches (Figure 7(e)); each patch is shown horizontally from left to right in small rectangles according to its creation order. Patches are example texture patterns created using the *Example Strokes* tool (details later). There is also a larger rectangle on right that displays the currently selected patch (Figure 7(f)). The remaining area of the drawing canvas is for freeform pen-and-ink illustrations.

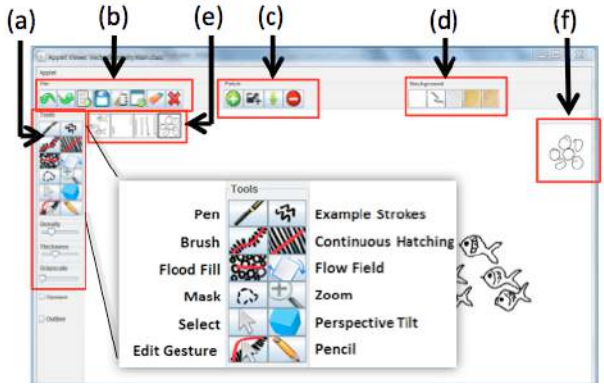

Figure 7: The User Interface of Vignette (a) Drawing and texturing tool (b) File/edit toolbar (c) Patch toolbar (d) Background palette (e) Patch palette (f) Selected patch

Vignette's main toolbar supports the five steps of the traditional illustration workflow. It has 12 buttons and 3 widgets (Fig. 7(g)), which can be grouped into 5 categories:

1) Tools for outlining. This category has the *Pencil* tool. In traditional pen-and-ink illustration, a pencil is used in steps 1 and 2 to outline the high level structure of an illustration. Similarly, strokes drawn with Vignette's *Pencil* tool are stored on a separate layer which can be easily removed after finishing the illustration.

2) Tools for detailed drawing. This category has the *Pen* tool, which is used to draw detailed non-repetitive strokes and fine details of an illustration, such as a person's eye.

3) Tools for specifying example textures. We created the *Example Strokes* tool to support the third step in the traditional workflow. Strokes drawn with this tool are collected into patches and later applied to different regions.

4) Tools for growing textures and tones. Tools in this category support the fourth step in the traditional workflow: *Mask*, *Brush*, *Continuous Hatching*, and *Flood Fill*. *Mask* defines a closed region to be filled up with the target texture. The others will be covered in more detail later.

5) Tools for texture layout and refinement. Textures can be refined interactively using the *Flow Field*, *Perspective Tilt*, and *Edit Gesture tool* (explained later). In addition, a number of slider widgets can be used to adjust the *tone*, *stroke width*, *grayscale value* of the textures.

Workflow in Vignette

The following steps illustrate the typical drawing workflow in Vignette:

Step 1: Users can draw a rough outline of the illustration using the *Pencil* tool.

Step 2: After the high level structure is defined, users can select the *Pen* tool to draw the detailed outlines. Users can use the *Mask* tool to define a region to be filled with texture.

Step 3: The user can then draw part of the texture using the *Example Strokes* tool. (Fig. 8(a), 9(a), 10(a)).

Step 4: The user then selects a texture filling tool (*Brush*, *Continuous hatching*, or *Flood fill*) and gestures to specify how the texture should be filled in (Fig 8(a), 9(a) and 10(a)). The example strokes are automatically collected into a patch, while the direction and curvature of the gesture specify the reference orientation of this patch. The system then generates the rest of the texture from the example patch to fill up the region (Fig. 8(b), 9(b), 10(b)). To understand how Vignette collects strokes into patches or fills in textures, refer to *Generating Patches From Example Strokes* and *Texture Synthesis* in our *Implementation* section.

Step 5: After generating the textures, users can interactively manipulate and fine-tune the textures to achieve the desired artistic effects, as explained in the following section.

Support for Interactive Refinement

Vignette's aids creative exploration by providing high-level controls for manipulating textures. Here, we briefly describe our interactive refinement capabilities.

Editing textures

To edit a texture, it must first be selected using the *Select* tool in the main toolbar. The corresponding patch appears as current patch in the top right of the canvas. As the user edits the example patch, the system interactively changes the selected output texture to reflect the change in the example patch (Fig 10(d)).

Editing tones

Users can edit the tone or density of a texture by manipulating a slider. Since each of the elements is represented by single point in the texture, we simply scale the density of the positions of the elements and re-render the elements. Variation of textures by tone editing is illustrated in Figures 8-11.

Editing the sweep of a texture

Users can interactively edit the sweep of a texture by editing the curvature of the reference gesture [10] (Fig 8(c)).

Matching Texture with Surface

By default, textures are filled in uniformly as if on a flat surface. Often, however, users may wish for textures to gradually change as they fill a region. Vignette provides two tools for this: perspective tilting and flow fields.

Perspective tilting is a technique for depicting 3D surfaces in illustrations. In perspective drawing, objects are drawn smaller (or "foreshortened") as their distance from the eye is increased. In our system, user can manipulate the perspective view of a texture without any underlying 3D information by manipulating the eye position with gesture with respect to the texture. Currently, our system supports one point perspective tilting (Fig 10). To tilt a texture, the user selects the *Perspective Tilt* tool in the main toolbar and drags the pen. The angle between the first point and the current point determines the direction of foreshortening, and the length determines the amount of foreshortening.

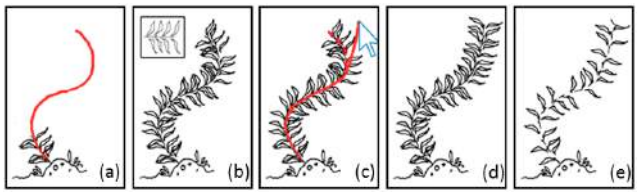

Figure 8: Brush operation and editing the curve of a texture. (a) Example strokes and user gesture (red). (b) Brush tool generates a texture from the example patch (inset). (c) User selects the curve editing tool and drags the mouse to sweep the curve. (d) New texture after the editing. (e) Tone variation (decreased density).

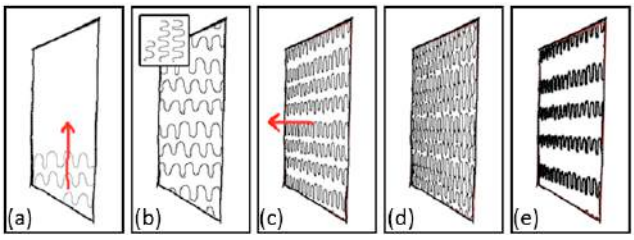

Figure 9: Continuous hatching and perspective tilting of a texture. (a) Example strokes and user gesture. (b) Continuous hatching creates a texture from the example patch. (c) User selects the tilt tool and creates a perspective tilt by dragging the mouse left. (d) Tone variation. (e) Tone and stroke width variation.

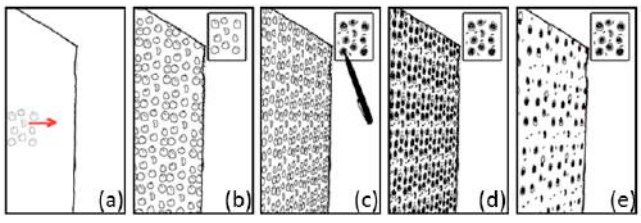

Figure 10: Flood fill and texture editing. (a) Example strokes and user gesture. (b) The flood fill creates the textures. (c) Perspective tilting and editing the source patch (inset). (d) The texture is updated interactively. (e) Tone variation.

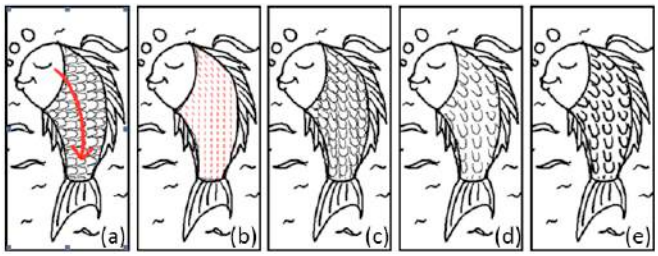

Figure 11: Orienting the elements with interactive flow field. (a) User gesture. (b) Underlying vector field from user gesture. (c) Rendering the elements along the vector field. (d) Tone variation. (e) Variation of stroke width of the strokes.

Flow fields allow users to specify the direction of the texture as it flows across a surface. In Vignette, users can select a texture, and then use the Flow Field tool in the main toolbar to adjust the direction of this field. Gesturing with the Flow Field tool tilts the field in the direction of the gesture, which orients the texture's strokes along the gesture. This is shown in Figure 11.

IMPLEMENTATION

In this section we discuss how Vignette supports the texture synthesis techniques described in previous sections.

Generating Patches From Example Strokes

The first step of our method is to generate a patch from the example strokes near a user gesture. After the user draws example strokes (black strokes in Fig 8(a), 9(a), 10(a)) and gestures over them (red curve in Fig 8(a), 9(a), 10(a)), example strokes near the gesture are gathered into a patch. The system clusters strokes together into elements by merging the strokes with overlapping bounds. Our intent was for an element to be a cluster of strokes that is perceived as a single feature by the user (as in [6]). In the leaf figure above, the five strokes are combined into a single element.

Texture Synthesis

The example patch provides a higher-level, perceptually meaningful description of example elements. The next step is to create a larger texture by synthesis from the example patch.

Each of the individual strokes is represented with a set of 2D points. An element is a group of strokes. In the textures, we represent each element by a point sample, which is the centroid of the element. During synthesis, we compute only the sample point without considering any other information of the original elements, like their geometry and appearance. After synthesis, we replace the sample points with the output elements.

Now we will briefly describe the synthesis techniques of the three tools: *brush*, *continuous hatching*, and *flood fill*.

Brush

The brush produces a 1D synthesis of elements [6] along the user gesture. Once the patch is computed, the elements are appended interactively along the gesture. The distance

between consecutive elements is computed from the example patch. The orientation of each element is computed from the tangent of the corresponding point on the gesture.

Continuous hatching (2D Synthesis and stitching)
Continuous hatching synthesis is performed in three steps. First, we generate the example patch and use the gesture direction (Fig. 12(a)) to perform a 1D synthesis along the gesture direction (Fig. 12(b)). Second, we duplicate and paste each element to fill up the region on either side of the gesture (Fig. 12(c)). Finally, we connect and merge the elements across the vertical direction using a simplified version of stitching [4] to create long seamless elements (Fig. 12(d)).

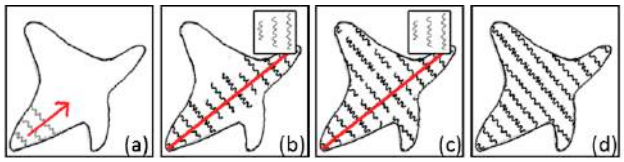

Figure 12: The steps for continuous hatching

Flood Fill (2D synthesis of discrete elements)
For flood fill, given an input exemplar patch ***I*** and an output masked area, the goal is to synthesize an output texture ***O*** that contains elements similar to exemplar patch ***I*** (Fig. 14).

In Vignette, we follow the EM methodology in [15] for texture synthesis because of its high quality and generality. This method iteratively places and then adjusts element positions in the texture to minimize the objective function ***E***. The objective function ***E*** is an evaluation criterion that quantitatively evaluates the arrangement of elements with respect to input example patch and performs heuristically chosen tests to try to reduce the energy. The basic solver gradually improves the neighborhood similarity term by iterating the two steps: *search* and *assignment* (explained below). The solver gradually decreases ***E*** while improving output quality iteratively (Fig. 14), and continues until the energy function ***E*** of output texture ***O*** is optimized.

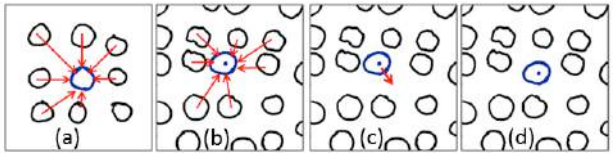

Figure 13: (a) The input example patch. (b) The output texture after initialization. In the *search step* for output element s_0 (marked blue in (b)), the algorithm finds the corresponding element s_i in the input patch (marked blue in (a)) with most similar neighborhood. (c) The *assignment step* computes the new position of the output element that minimizes the energy between corresponding input elements. (d) Finally, the element is moved into its new position.

Initialization: First, we copy the input patch into the output exemplar similarly to tiling methods, but with some random variations in positions.

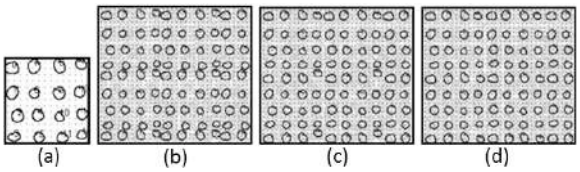

Figure 14: The iterative progression of texture optimization. (a) Input patch (b) Output texture after initialization (c) Output texture after iteration 2 (d) Output texture after iteration 4.

Search step: During the search step, for each output sample s_o, we find the input sample s_i with the most similar neighborhood, i.e. minimizing the energy value in accordance to a neighborhood similarity metric (Fig. 13(b)). This search is conducted by exhaustively examining every input sample for each output element.

Assignment step: After computing the best matching input patch, this step computes the position of output elements that minimizes the energy function (Fig. 13(c)) and moves the element to their new positions (Fig. 13(d)).

Interactive Refinement

Flow field: The task of arranging elements according to a gesture can be divided into two sub-tasks: 1) creating an orientation field over the surface from a user input gesture, and 2) rendering elements according to the orientation field.

We have used a vector field to represent the orientation. In our system, user gestures determine the direction of this field at points within a pre-defined distance of the gesture. Figure 11(b) shows a small number of red vectors that have been set by the gesture in Figure 11(a). With an orientation field in hand, we then orient the elements in accordance to the vector field using property layers similar to *modeling with rendering primitives* [23].

USER EVALUATION

Vignette has a unique approach to design and manipulation of textures in pen-and-ink illustrations. It keeps the essential steps of the traditional pen-and-ink workflow while providing gesture controls for texture synthesis. There are few existing research or professional tools designed for the same purpose, and none are directly comparable. Adobe Illustrator may be the closest match in terms of texture creation and manipulation, but it is a general purpose graphical editing tool with an entirely different workflow and interaction style. It also has many additional features/functions way beyond the need of pen-and-ink illustration.

Nevertheless, it is important to understand how professional artists feel about Vignette, and how it compares with the traditional pen-and-ink drawing experience and with existing digital tools such as Adobe Illustrator. To do this, we invited four professional artists to use Vignette, while we sought to answer the following three questions.

1. *How do artists generally feel about Vignette? Does Vignette fit their needs?*

2. *How does the pen-and-ink illustration workflow in Vignette compare with paper and with digital tools?*
3. *How are Vignette's features used and accepted by artists? Are there opportunities for improvement?*

Participants and Environment

Four professional artists (P1-P4, 3 males, 1 female, age range 23-55 years old) participated in our evaluation. P1 and P2 are accomplished expert artists. They both work as pen-and-ink illustrators, animators, and directors with 15 or more years of experience. P3 and P4 are intermediate level artists trained in design and illustration at universities. Both have 4 or more years of experience in digital painting. All participants are proficient with Flash, Photoshop, Illustrator and many other tools with 4 or more years of experience.

All evaluation sessions took place in a laboratory. Vignette is built with Java and runs on a standard laptop. All drawings were done on a Cintiq 12wx tablet.

Method

The evaluation was conducted in the following three steps.

Training (15-20 minutes): Participants were first given a brief introduction to Vignette. They then received a tutorial, which consisted of a printed sheet with seven practice drawings chosen to demonstrate the interface and features of the system. Participants were asked to create and interactively refine these drawings to achieve the target result. The facilitator did not intervene unless a participant had trouble creating a drawing.

Illustration (40-65 minutes): In this step, participants were asked to create pen-and-ink illustrations. Some of these can be seen in Figures 15 and 16 (far left).

Feedback. (10-15 minutes): Finally, participants answered a questionnaire about Vignette.

We sought to answer our three questions primarily by observing participants and recording their spontaneous comments. The following sections summarize our findings.

Overall impression of Vignette

Participants' overall reactions were very positive. During the course of the evaluation, the participants created many illustrations with a wide variety of textures, such as textures for architectural drawings, landscapes, animals, crowds, fireworks, and abstract scenes. Participants responded that Vignette was fairly easy to learn, all giving it a 4 on a scale of 1 (extremely difficult to learn) to 5 (extremely easy to learn). Participants also expressed satisfaction with their artworks, with an average rating of 4.25 on a scale of 1 (extremely unsatisfied) to 5 (extremely satisfied). All commented that Vignette provides a pleasant drawing experience.

Participants found Vignette particularly suitable for two purposes: 1) creating original pen-and-ink illustrations from scratch 2) quickly exploring and experimentation with different types of textures. As mentioned by P1, "*I can draw really quickly, and do a lot of explorations… inspire me to explore more...*"

Participants particularly liked the ability of Vignette to preserve their natural drawing styles. Three participants specifically like the natural and hand drawn scribbling effect of the final artworks, as mentioned by P1, "*it looks like I drew each and every stroke manually… and it is not obvious that the textures were created using a computer tool*".

Workflow and Experience Comparison with Alternatives

Participants were able to create artworks with rich textures in a short time (11 minutes per drawing in average) after only 15 minutes of training (see the artwork in Figure 16, all except the second artwork were created by the participants during the course of evaluation). All participants commented that it would be very tedious to produce drawings with similar quality either in traditional pen-and-ink style or using another digital tool.

Participants also commented that the advanced digital features make the illustration process enjoyable, which is traditionally very tedious to do. According to them, Vignette is both effective and convenient, and preferable to manual illustrations and other professional tools for pen and ink illustrations. The ability of users to create such a collection of artwork in a short time demonstrates the expressiveness and ease of use of Vignette.

Vignette vs. Traditional Pen-and-Ink Drawing

Participants commented that although Vignette has many advanced digital capabilities, but the fact that it is designed to follow the traditional pen-and-ink workflow makes the system feels natural to work with and easy to learn and use. On the other hand, using Vignette significantly improved the productivity of drawing.

During the course of evaluation: for example, in Figure 15, the user drew a patch of three persons and later used subsequent brush tools for creating a crowd from the example patch. Similar approach was used to draw fireworks. This process follows the traditional pen-and-ink workflow, but is much accelerated. This illustration was created (Fig. 15(right)) in less than 5 minutes, from three example strokes (marked blue) and few gestures (marked red)(Fig 15(left)).

However, the participants also mentioned that there are certain desirable properties of the traditional pen-and-ink illustrations currently lacking in Vignette, such as the variations of strokes produced by different types of pencils, pens and brushes.

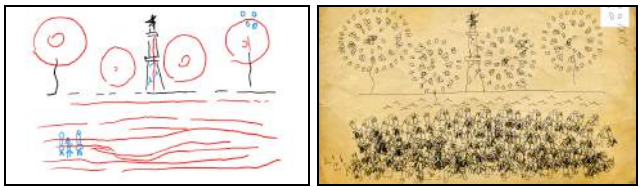

Figure 15: The example strokes (blue) and gestures (red) drawn by a participant to produce an illustration in 5 minutes (left). The final illustration (right)

Vignette vs. Adobe Illustrator

According to the participants, Adobe Illustrator is the closest tool they can think of to create and manipulate vector graphics textures. All the participants mentioned that one major difference between Vignette and Illustrator lies in the interface and interaction style. The design of Vignette allows creating illustrations quickly and easily. As mentioned by P2: "*I like the free-form gesture based interaction… it is easy to learn and use… With gestures, a few scratches in the canvas can create illustrations within a minute*".

Compared to Vignette, although Illustrator has many built-in support features for texture and patterns (such as Pattern Brush), it is not optimized for pen-and-ink illustrations. According to P1 – "*Traditional tools have too many functions and options. It is difficult for me to use, (these features) are very often distracting for performing a certain painting*".

Furthermore, participants noticed that Vignette provides additional useful capabilities not available in Illustrator. For example, Illustrator does not provide support for continuous hatching quickly and easily like Vignette. The flood fill effect of Illustrator is simple tiling, hence it can produce results like Figure 14(b). But, iterative texture optimization reduces the energy function and produce visually better results (Figure 14(d)), which is suitable for hand drawn textures, since hand drawn textures are not directly tillable most of the times.

Feature Usage and Feedback

To understand the relative usage of features, we recorded the number of times each feature was used in the free task. The four participants spent a total of 66 minutes on free tasks, during which we logged 215 feature usages (average 3.16 features per minute).

Each user made moderate use of most features, though the use of brush to create textures stood out, accounting for 123 (55.4%) occurrences of all feature usages logged. Two of the users made heavy use of brush (71% and 65% of total feature usage respectively). One of these two users didn't use continuous hatching at all, while the other two users made heavy use of continuous hatching (46% and 37% of feature usage respectively). Flood fill was used by two users.

Almost all participants used all the texture synthesis operations multiple times that indicate that users found them useful in creating art works on Vignette. The participants also like the interactive refinement capabilities, including - flow field, editing tones and textures. According to P1 and P2, editing the flow of elements is very useful and cannot be done easily with other traditional tools

Opportunities for Improvement

Vignette worked well overall, but we saw several ways to improve it. For certain kinds of textures, having long overlapping example strokes, the synthesis results sometimes look repetitive. Also, like any stroke-based rendering system, Vignette's performance degrades as the number of strokes increases. Our experience is that performance degrades with more than 1000 strokes.

Another limitation of Vignette is that leftward brush gestures appear to create different textures from rightward gestures, because leftward gestures vertically flip the texture. This happened to P1 and P3 a total of 9 times. One user suggested having a preview panel for testing the gesture effects before applying them in final drawings.

Our study users suggested two additional tools. One suggested a gradient fill tool that would allow elements to vary in shape, color, and size across a region. This is similar to our perspective tilting feature, but more general. Another user wanted to save a separate version of a patch with a different scale, tone, and stroke width.

CONCLUSION

Vignette is a practical tool with a natural workflow for pen-and-ink illustrations. Texture illustration is tedious, but current texture synthesis tools cannot easily capture illustrators' personal style. Furthermore, these tools disrupt the traditional illustration workflow, because they are tedious and draw attention to dialog boxes and away from the illustration itself. Vignette speeds up texture creation while preserving the traditional workflow capturing artists' personal style.

We analyzed the traditional illustration workflow and illustration artifacts to guide designers of illustration systems that preserve this traditional feel. We then described the user interface and implementation of our Vignette system. Finally, we presented an evaluation that shows how artists can use it to quickly create artworks in their own personal style.

Our exploration of natural workflow and gesture-based interaction was inspired by a traditional approach to creating illustrations. We hope to inspire others to create digital art media that preserve the beauty of traditional media.

ACKNOWLEDGEMENT

We would like to that Jun Mitani and Kenshi Takayama for their valuable feedback and suggestions during the development of the project. We also thank Yuki Tsujita, UrmaDelvi and other artists for taking part in user testing and providing valuable insights.

Figure 16: Different artworks with Vignette. The artworks took 16, 13 and 19 minutes respectively, completely from scratch

REFERENCES

1. Barla P, Breslav S, Thollot J, Sillion F and Markosian L. Learning Style Translation for the Style of a Drawing. ACM Transactions of Graphics 22,1 (2003), 33 - 46
2. Barla, P., Breslav, S., Thollot, J. and Markosian, L. Interactive hatching and stippling by example. INRIA Technical Report (2006)
3. Deussen, O., Hiller, S., Overveld, C. and Strothotte, T. Floating Points: A Method for Computing Stipple Drawings. *IEEE Computer Graphics Forum* 19,3 (2000), 41 - 50
4. Efros, A.A. AND Freeman W.T. Image quilting for Texture Synthesis and Transfer. In Proc. ACM SIGGRAPH (2001), 341 - 346
5. Efros, A.A. and Leung, T.K. Texture Synthesis by Non-parametric sampling. IEEE Computer Vision Vol.2 (1999), 1033 – 1038
6. Gröller, E. and Szirmay-Kalos, L. Stroke Pattern Analysis and Synthesis. EUROGRAPHICS, 25 (2006), Number 3
7. Guptill A.L. Rendering with pen and ink. Watsom-Guptill Publications, New York, 1976.
8. Hertzmann, A., Oliver, N., Curless, B., Seitz, S.M. Curve Analogies. *In Proc. Rendering Techniques* (2002), 233- 246
9. Igarashi, T. and Hughes, J.F. Clothing Manipulation In Proc. ACM UIST (2002), 91- 100
10. Igarashi, T., Moscovich, T. and Hughes, J.F. As-rigid-as-possible Shape Manipulation. In Proc. ACM SIGGRAPH 24 ,3 (2005), 1134 - 1141
11. Ijiri, T., Mech, R., Miller, G. and Igarashi, T. An example-based procedural system for element arrangement. Computer Graphics Forum 27,2 (EUROGRAPHICS 2008), 429 - 436
12. Kalnins, R.D., Markosian, L., Meier, B.J., Kowalski, M.A., Lee, J.C., Davidson, P.L., Webb, M., Hughes, J.F. and Finkelstein, A. WYSIWYG NPR: Drawing Strokes Directly on 3D Models. *In Proc. ACM SIGGRAPH in Transactions on Graphics* 21,3 (2002), 755-762
13. Kazi, R.H., Chua, K.C., Zhao, S., Davis, R., Lim, K. SandCanvas: A multi-touch art medium inspired by sand animation. ACM CHI 2011. pp. 1283-1292.
14. Kopf, J., Fu, C., Chohen-Or, D., Deussen, O., Lischinski, D. and Wong, T. Solid Texture Synthesis from 2D exemplars. In Proc. ACM SIGGRAPH 26, 3 (2007), 21- 29
15. Ma, C., Wei, L. and Tong, X. Discrete Element Textures. *In Proc. ACM SIGGRAPH* 30, 4 (2011), Article 62
16. McCloud, S. Making comics. Harper Paperbacks, 2006.
17. McCloud, S. Understanding Comics. Harper Paperbacks, 1994.
18. Nelson Chu WB, Li-Yi Wei, and Naga Govindaraju. Detail-preserving paint modeling for 3D brushes. *Non-Photorealistic Animation and Rendering* 2010.
19. Nice, C. Drawing in Pen and Ink. North Light Books, 1997
20. Ryokai K, Marti S, and Ishii H. I/O brush: drawing with everyday objects as ink. In. *Proc. of CHI* 2004, 303-310.
21. Salisbury, M., Anderson, S., Barzel, R. and Salesin, D. Interactive pen and ink rendering. *In Proc. ACM SIGGRAPH* (1994), 101 - 108
22. Salisbury, M.P., Wong, M.T., Hughes, J.F. and Salesin, D.H. Orientable Textures for Image-Based Pen-and-Ink Illustration. *In Proc. ACM SIGGRAPH* (1997), 401- 406
23. Schwarz, M., Isenberg, T., Mason, K. and Carpendale, S. Modeling with Rendering Primitives: An Interactive Non-Photo Realistic Canvas. Non-Photorealistic Animation and Rendering (2007), 15 - 22
24. Simmons, G. The Technical Pen. Watson-Guptill, 1992.
25. Turk, G. Texture synthesis on surfaces. In Proc. ACM SIGGRAPH (2001), 347 - 354
26. Vandoren P, et al. FluidPaint: an interactive digital painting system using real wet brushes. In. *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces* 2009, ACM (2009), 53-56.
27. Vandoren P, et al. IntuPaint: Bridging the Gap between Physical and Digital Painting. *IEEE TABLETOP* 2008(2008).
28. Wei, L. and Levoy, M. Fast texture synthesis using tree-structured vector quantization. In Proc. ACM SIGGRAPH (2000), 479 - 488
29. Wei, L., Lefebvre, S., Kwatra, V. and Turk, G. State of the Art in Example-Based Texture Synthesis. *Eurographics* (2009)
30. Winkenbach, G. and Salesin, D. Computer Generated Pen and Ink illustration. *In Proc. ACM SIGGRAPH* (1994), 91 - 100
31. Winkenbach, G. and Salesin, D.H. Rendering Parametric Surfaces in Pen and Ink. *In Proc. ACM SIGGRAPH in Computer Graphics* (1996), 469 – 476
32. Winnemoller, H., Orzan, A., Boissieux, L. and Thollot, J. Texture Design and Draping in 2D images. Computer Graphics Forum. In Proc. Eurographics 28, 4 (2009), 1091 - 1099