1 Introduction

In recent years, painterly rendering techniques have achieved remarkable success for simulating watercolor, oil, and oriental ink paintings. Many artistic filters, such as [DeCarlo and Santella 2002; Collomosse and Hall 2006] have been proposed for converting real photos into painting styles by applying the global operation with minimum user interaction. Although artistic filters are particularly suitable for impressionism and impressionist by clustering the same color pixels into simple geometrical pieces as strokes and simulating the overlapping of these strokes, they hardly achieve oriental ink painting effect. The reason is that oriental ink painting embeds all scene information, such as color, texture, shape, and light sources into few abstract strokes. The shapes of the oriental ink strokes therefore are not simple as bended rectangles but contain more complex changes. 

Alternatively, sketch-based approaches provide users with more flexibility and control by allowing them to interact with the system. In previous sketch-based approaches, such as [Okabe et al. 2007], the user sketches the brush trajectories using a mouse or a pen-like device. The system then automatically converts the trajectories into strokes. The limitation of inputting the trajectories is that the user loses the control to generate the shapes of the strokes. In our system, the user sketches the boundaries of the strokes as the input. It overcomes the drawback of losing the control of the shape as well as inheriting the advantage of the draft drawing that is efficient for designing the composition of the painting by contours of the strokes when the idea comes out. In our previous work [Xie et al. 2010], we proposed a framework for interactively rendering realistic images into oriental ink paintings. The distinctive feature is that it can handle a large variety of brush shapes and styles which cannot be generated by previous methods. In this paper, we focus on the implementation details of the software and design aspects of the user interface.

2 Overview of the iR2s system

The system first analyzes the complexity of the shape covered by the input boundary. Complex shapes should be decomposed into elementary drawable parts that can be rendered with a single brush stroke. The decomposition is done automatically based on the distribution of branches on the skeleton of the shape. Then, we automatically estimate the optimal brush trajectory on each drawable part by minimizing an energy function parameterized by the physical parameters of the painting process. This energy function depends on the attributes of the motion inertia of the brush and the friction between a brush and a paper, such as the variations in the rotation, size, and angular velocity of the brush along a trajectory. The oriental ink painterly rendering of these estimated trajectories is lastly done by direct brush footprint texture mapping. We take the sample collection of the footprint images during the real painting process and build a dictionary of the sampling footprints, which have different texture and ink quantities. Using different permutation of the footprints, we map them onto the estimated footprint positions and interpolate in-between to render smooth strokes with variant texture. Since the estimated series of footprints cover the shape of the stroke by touching the boundary, strokes with asymmetric texture can be easily rendered.

3 The user interface

iR2s allows users to sketch the shapes they want to convert into oriental ink. The user interface is composed of a control panel, a drawing canvas and a window for displaying the painting results in Figure 2. On the control panel, all of the options are integrated. Users can easily choose any option they want. We also provide the general edition options, such as undo, remove all and save. We defined the undo process is the action to remove the last input boundary and its rendering result. Before starting the painting process, the user will choose either the blank white canvas or the real photo canvas. The user paints on the blank white canvas as a normal painting way. When the user wants to edit a real photo into oriental ink style, they can use the real photo canvas model to sketch the boundaries on the source image. We later introduce the way to draw a basic stroke on the blank canvas in Section 3.1. Section 3.2 describes the process of the complex shapes. We also illustrate the way to edit the real image into oriental ink painting in Section 3.3.

3.1 Basic stroke generation

To paint a basic stroke, the user first clicks the contour button to order the system to generate a new stroke and then sketches on the canvas the boundary of the stroke. For drawing the boundary, we recommend users to draw a continue closed line without any cross. The line drawing should be stopped near the start point. The system will link the start point and the end point and smooth the line. The user can select the style of the brush by choosing the desired style from the control panel. We provide six styles which are the most common in oriental ink painting: dry-ink stroke, full-ink stroke,
first-half hollow stroke, latter half hollow stroke, middle hollow stroke and both ends hollow stroke. Users can tune the DryCoef bar to set up the initial ink quantity of the brush. We also provide two most useful drawing ways to estimate the brush trajectory: the vertical brush trajectory and the slop brush trajectory. When users want to generate a stroke in vertical style, they can check the isVerticalBrush checking box. The vertical brush trajectory is usually used to generate the slim and straight strokes. The slop brush trajectory is used to generate the strong strokes with rich variance of the texture. The user can also choose the color of the pigment from the pop-up color palette. Finally, the user click the render button to render the stroke in oriental ink painting style. Visualizing the intermediate processing steps is also important for the user to understand how our system works step by step. The user can click the appropriate check box on the control panel, such as the brush path check box, to show the result of the optimized brush trajectory algorithm.

3.2 Complex shape processing
After the user draws the boundary, the system will automatically analyzes the shape surrounding by the input boundary and then decide if it needs to be decomposed. Users do not need to care if the input shape is complex or how to decompose the input shape. When the system detects the shape is complex, it will decompose the shape into the elementary drawable parts. The system then estimates the brush trajectory for each elementary shape and renders these trajectories into oriental ink painting. For observing the visualization of the complex shape decomposition, the user clicks the decomposition button on the control panel. The result will be displayed on the sumi-e window.

3.3 Interactive image to oriental ink painting
We recommend users who do not expert at drawing with a brush to choose the model of painting on real images. Users can easily sketch the boundaries on the photo and directly compare the rendering result with it. In the system, we apply the Drag and Drop tool [Jia et al. 2006] for interactive object segmentation. Instead of careful specification of the boundary by user, the tool will optimize the boundary by iteratively finding the shortest closed-path around the region of interest. The system automatically samples the points of the optimized boundary and make it into a closed smooth curve which will be displayed on the control panel marked in different color from neighbors. The system then automatically process this smooth curve in the way mentioned in section 3.1 and 3.2. We have explained how to generate a simple stroke. For rendering a complex scene as shown in the sumi-e window in Figure 2, the user just needs to repeat the process of generating the simple stroke one by one as the normal painting process.

4 Conclusion
We developed the software for generating strokes in oriental ink painting style. Normal users can easily render the strokes without any painting knowledge by just sketching the boundary and choosing the brush style which they want to use. In the future, we plan to extend this interactive image painterly rendering software into automatic generation of the painterly style animation from the video. We believe it will be helpful for reducing the difficulty of making a animation and shorten the production cycle.

References