Data-Driven NPR Illustrations of Natural Flows in Chinese Painting

Yu-Chi Lai, *Member, IEEE*, Bo-An Chen, Kuo-Wei Chen, Wei-Lin Si, Chih-Yuan Yao, *Member, IEEE*, and Eugene Zhang, *Senior Member, IEEE*

Abstract—Introducing motion into existing static paintings is becoming a field that is gaining momentum. This effort facilitates keeping artworks current and translating them to different forms for diverse audiences. Chinese ink paintings and Japanese Sumies are well recognized in Western cultures, yet not easily practiced due to the years of training required. We are motivated to develop an interactive system for artists, non-artists, Asians, and non-Asians to enjoy the unique style of Chinese paintings. In this paper, our focus is on replacing static water flow scenes with animations. We include flow patterns, surface ripples, and water wakes which are challenging not only artistically but also algorithmically. We develop a data-driven system that procedurally computes a flow field based on stroke properties extracted from the painting, and animate water flows artistically and stylishly. Technically, our system first extracts water-flow-portraying strokes using their locations, oscillation frequencies, brush patterns, and ink densities. We construct an initial flow pattern by analyzing stroke structures, ink dispersion densities, and placement densities. We cluster extracted strokes as stroke pattern groups to further convey the spirit of the original painting. Then, the system automatically computes a flow field according to the initial flow patterns, water boundaries, and flow obstacles. Finally, our system dynamically generates and animates extracted stroke pattern groups with the constructed field for controllable smoothness and temporal coherence. The users can interactively place the extracted stroke patterns through our adapted Poisson-based composition onto other paintings for water flow animation. In conclusion, our system can visually transform a static Chinese painting to an interactive walk-through with seamless and vivid stroke-based flow animations in its original dynamic spirits without flickering artifacts.

Index Terms—NPR, data-driven, chinese ink painting, tensor field smoothing, naiver-stokes equations

1 Introduction

Chinese ink paintings are pervasive in Asia and some art departments in Europe and the Americas. Classes are offered to teach this unique style of paintings. Historically, Chinese scholars with accomplishments in politics, literature, martial arts, and science learned this art. Their products depicted beauty and harmony of nature through brush strokes that shape and move objects such as water. In still paintings, special techniques are employed to deliver the endless spirit and flowing sense of water. On the other hand, dynamic water movement can enhance the effect dramatically. There have been several recent developments to augment traditional static paintings with moving objects. For example, the movie titled "Moving of the Riverside Scene at Chingming Festival" receives positive reviews when shown at WorldExpo 2010 (Shanghai, China). In the movie, animated characters and dynamic objects (e.g., rivers) are seamlessly blended into famous classical Chinese paintings. Those animations bring the original static painting to life.

- *Y.-C. Lai, B.-A. Chen, W.-L. Si, K.-W. Chen, and C.-Y. Yao are with the Department of CSIE, National Taiwan University of Science and Technology, Taipei, Taiwan 106. E-mail: {cheeryuchi, ambersook, chen51202, ssss0177, cyuan.yao}@gmail.com.*
- *E. Zhang is with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331. E-mail: zhange@eecs.oregonstate.edu.*

Manuscript received 15 May 2016; revised 1 Oct. 2016; accepted 1 Oct. 2016. Date of publication 31 Oct. 2016; date of current version 27 Oct. 2017. Recommended for acceptance by H. Theisel. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TVCG.2016.2622269

Animating characters and adding dynamic elements such as river flows to a Chinese painting requires a large amount of manual work, which is often in the order of several man-months. Our paper addresses these difficulties with an intuitive and dynamic data-driven system to animate water flows with strokes in an existing Chinese painting. Fig. 1 demonstrates the extraction of the strokes and two snapshots of animations, and Fig. 2 gives an overview of our work flow. The user can first semiautomatically create and place $2.5D$ scene objects by a few scribbles to construct the virtual setting of an existing painting. He/she can also specify the flow regions on an existing Chinese painting through placing rectangles in order to animate the flow in the painting. Our system automatically detects and identifies all strokes recorded as skeletal strokes and estimates a flow pattern based on the extracted stroke structures, ink densities, and placement densities on the user selected flow regions. The flow pattern does not take interactions among flows and barriers into consideration. Thus, a fluid simulator solving the Navier-Stokes equations [1] takes the extracted flow pattern as initial conditions to create an intended field in a spatiotemporally coherent fashion. Furthermore, the solver allows us to consider interaction between flows and dynamic objects as addition of disturbance forces along the moving objects. Additionally, we also provide editing tools to let the user adjust the constructed flow fields for better matching the designer's intention in animating strokes. Since painters generally use patterned strokes to profoundly portray flow patterns and surface structures, our system clusters extracted strokes into

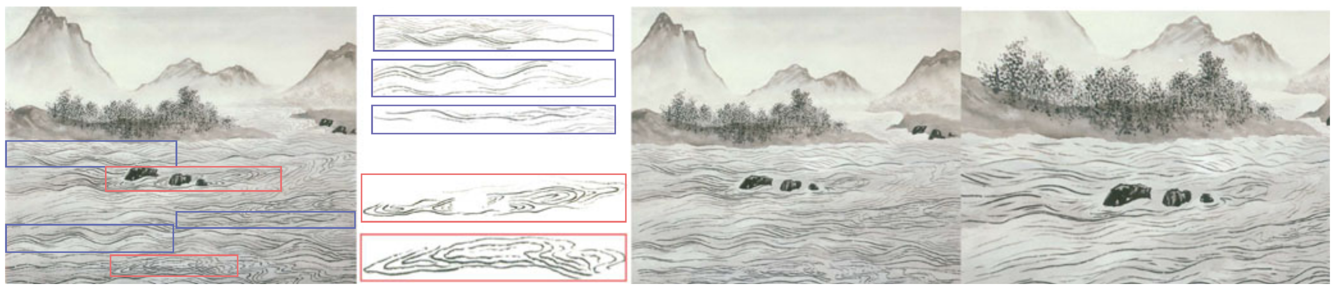

Fig. 1. These consist of 1) the electronic scanning of “Chi Bi Tu”/“Red Cliff” drawn by Yang, Shin-Hsian, 2) several stroke pattern groups extracted from the painting, 3) a snapshot of the animated “Red Cliff”, and 4) another snapshot of the animated “Red Cliff” when a user changes the view point.

stroke pattern groups based on the computation field, and their adjacency and curvature. Finally, the extracted strokes are dynamically added onto or removed from the water flow with fading effects, and the flow field is used to animate strokes for creation of an endless water flow animation with controllable smoothness and temporal coherence. These designs remove flickering and popping artifacts commonly observed in NPR animations. Furthermore, our system can allow users to place the extracted stroke pattern groups seamlessly onto the water region of another painting using an adapted Poisson style transition algorithm, and the placed strokes can be animated in a similar style as the original one.

Technically, we make the following contributions in the paper.

1) We have proposed an interactive, efficient, and intuitive data-driven system consisting of a 2.5D scene design subsystem, a flow pattern extraction subsystem, and a procedural field generator. Overall speaking, our system can augment traditional Chinese paintings by animating water objects such as rivers and waterfalls in a 2.5D manner.
2) Our system artistically drives and animates strokes with a flow field constructed procedurally to simulate artistic intention conveyed in the painting using the properties of extracted flow strokes and the physical settings of the virtual world. Our proposed animation scheme could control the speed and appearance of stroke pattern groups with temporal coherence. The stroke groups could keep the original sprits and animation coherence of Chinese painting.

As demonstrated in the results, users can easily create an interactive walk-through visually similar to an existing Chinese painting and seamlessly and vividly animate water objects without flickering artifacts in its original painting spirit.

2 Related Work

There are a number of algorithms that simulate the appearance of Chinese paintings by the NPR community. Many of these techniques focus on simulating the interactions among paper, water, ink, and light [2], [3], [4], [5]. These techniques can generate realistic appearances of Chinese paintings, but they are often too computationally expensive for real-time walk-through and interaction. Xie et al. [6], [7] perform stroke placement with contours extracted from photos while Lu et al. [8] save simulation time with a stroke database. However, these algorithms are not efficient for real-time applications and do not consider the temporal coherence to possibly induce serious flickering artifacts. Our system overcomes their limitations by automatically placing and animating stroke pattern groups with an artistically and procedurally constructed flow field.

There are image-based methods [9], [10] that can generate flow effects using pixel movement. Bhat et al. [11] analyze the motion of textured particles along user-specified flow lines to transfer flow effects onto other flow lines. Eden et al. [12] integrate fluid simulation with comic shading effects for animations. These methods work for advecting pixel- and patch-based drawing and hatching used mostly in oil and water color paintings. Chinese paintings illustrate

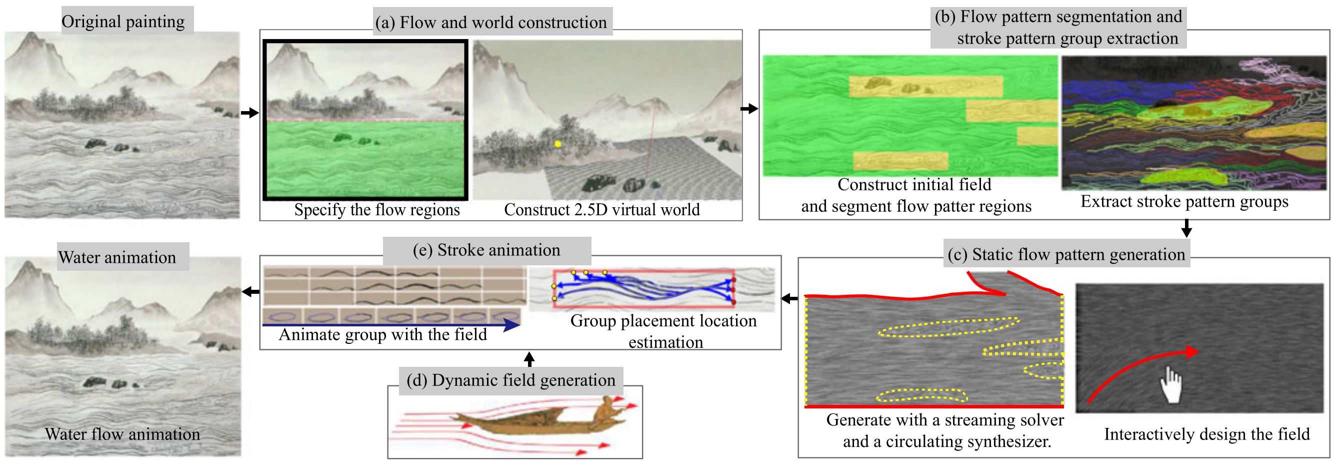
Original painting
(a) Flow and world construction
Specify the flow regions
Construct 2.5D virtual world
(b) Flow pattern segmentation and stroke pattern group extraction
Construct initial field and segment flow patter regions
Extract stroke pattern groups
(c) Static flow pattern generation
Generate with a streaming solver and a circulating synthesizer.
Interactively design the field
(d) Dynamic field generation
(e) Stroke animation
Animate group with the field
Group placement location estimation
Water animation
Water flow animation

Fig. 2. The proposed algorithm consists of the following steps: interactive water region identification and 2.5D world construction (a), flow profile extraction (b), data-driven procedural field generation (c), static and dynamic object placement (d), and stroke generation and animation (e).

flow motion with long and curly strokes instead of pixels and patches, and our system achieves this by advecting strokes with a data-driven artistic flow field.

Semmo et al. [13] compute a feature-aligned distance field to place lines and textures, but they do not consider flow dynamics. Although Svakhine et al. [14] combine different focal and contextual illustrative styles to informatively depict 3D flows, they do not aim at expressing and abstracting flows artistically. Furthermore, they place curves with an uncontrolled amount of overlapping which contrasts the spirit of tranquility often accentuated in Chinese paintings. Yu et al. [15] combines procedural and structural line textures with water flow templates to depict cartoon water and animate flows with fluid dynamics. However, their system depends on the predetermined template, stroke styles, and flowing patterns. Our system extracts the drawing styles from an existing painting and animates them with a data-driven artistic flow field computed based on the extracted strokes.

Zhang et al. [16] develop an interactive Chinese painting water flow system which takes a video containing a river or a fall as input to track the water flow for stroke placement. They also provide a user interface to adjust the content and strokes of the resulted painting animation to convey the desired spirit of the user. However, their optical flow tracking is erroneous, and the system does not take the banks and barriers into account. Furthermore, heavy manual adjustment is needed to generate reasonable results. Xu et al. [17] extend the image-space skeleton animation technique to animate characters and figures in an existing Chinese painting. They first use image segmentation to segment a character or figure into several patches and then compute a skeleton based on the segmented patches for animating the character or figure in the painting. However, how to specify water skeletons is not trivial. In addition, all these research focus on existing 2D paintings and cannot easily be extended to change the view point for a 3D walk through of the virtual world. Our system can generate an animating 2.5D Chinese painting, and users can easily choose any viewing position and direction as they desire. Due to the lack of proper techniques which can faithfully and seamlessly animate existing flow strokes of a Chinese painting in 3D walk-through, artists have to spend a huge amount of labor and time on laying down, moving, and deforming flow strokes of simple structures for creating "Moving of the Riverside Scene at Chingming Festival". As a result, when taking an existing Chinese painting, our system can compute a flow field based on automatic extraction of an initial flow pattern to artistically animate the extracted flow strokes in the painting. Before describing our flow animation details, we first provide a taxonomy of flow-portrayed techniques to depict clearly the technical difficulties, and our focus in the next section.

3 Taxonomy of Water Flow Skills in Chinese Painting

Comparing to western paintings, Chinese paintings strive to capture closeness to nature in spirit rather than physical resemblance. Accordingly, there are generally four major differences: 1) Painters portray scenes with more lines than western painters due to pursuit on spiritual similarity. 2) Painters draw a world with multi-perspective views instead of a single perspective view to have an aggregated view of the world. 3) Painters mainly focus on expression and pose semantics instead of physically correct figure information. 4) Painters left blank for the background instead of complex details.

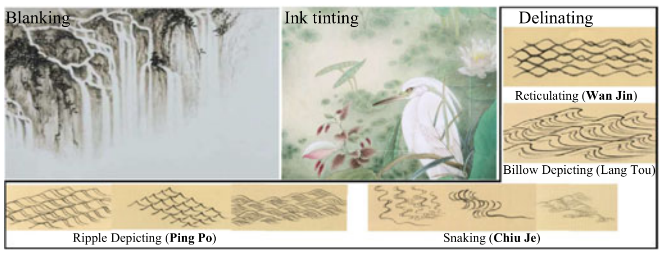

Fig. 3. These show an example of blanking used to imitate a fall, an example of ink tinting describing a peaceful pond, and several basic curvilinear structures for description of water flows.

The water flow is either top-down for falls or left-right because famous rivers in mainland China is geologically from west to east. Painters mainly draw flows with the following three skills as shown in Fig. 3: *Blanking* leaves blank for flows and uses other painted objects such as banks and river rocks to imitate the existence of water. *Ink tinting* uses ink and pigment dispersion instead of curvilinear structures to portray flows. *Delinating* uses curvilinear structures to express different flow states and surface structures. Additionally, painters sometimes use both *Ink tinting* and *Delinating* for better results. *Blanking* can be animated by moving imitated objects, and *Ink tinting* can be animated by pixel-based advection [9], [11]. *Delinating* has a variety of styles and exists in a large number of Chinese paintings based on our observation. There are several representative drawing styles as shown in Fig. 3. *Ripple depicting (Ping-po)* portrays ripples using upward sharp turns for peaks and downward smooth transitions for valleys and then stack multiple fundamental strokes together to illustrate the overall water flows. *Snaking (Chiu-je)* uses sets of strokes with fluent brush motion to portray complex flow patterns. *Reticulating (Wan-jin)* lays down net-like long strokes for the sense of widely and extensively spreading. *Billows depicting (Lang-tou)* portrays wave breaking, agitated sea, and tidal waves for flows with tremendous momentum. This work mainly focuses on animating curvilinear strokes in the form of *Ping-po*, *Chiu-je*, and *Wang-jin* classified as the Gou-ran style for an existing painting.

4 Chinese Painting Analysis For Static Scene and Water Flow

In order to provide an interactive walk-through of an existing Chinese painting, our system provides a user interface for semiautomatically intuitive and fast extraction of scenic objects to create the 2.5D virtual world by laying down a few scribbles with specific depth values. He/she can also specify the flow regions through laying down squares. Next, as shown in Fig. 4, our system automatically extracts flow strokes recorded as skeleton strokes using extracted appearance, texture, and oscillation factors. We record stylish stroke patterns by grouping strokes based on their vicinity, adjacency, and curvature. Our system can also postulate

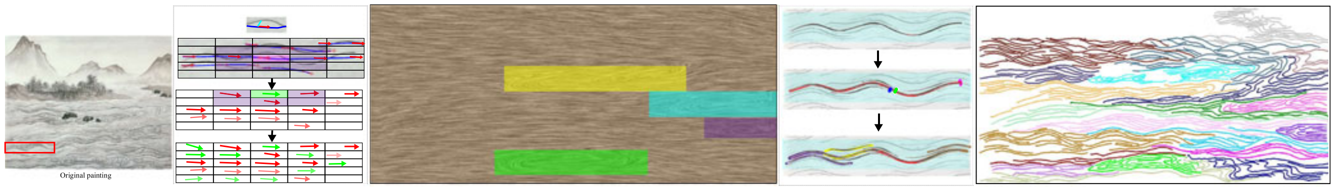

Fig. 4. This overviews static stroke analysis consisting of stroke extraction, data-driven initial flow velocity estimation, data-driven initial field construction, flow pattern segmentation and classification, and stroke clustering.

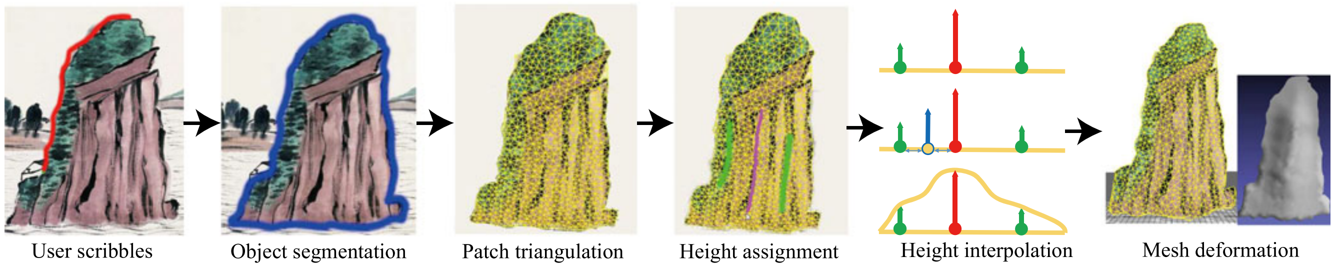

Fig. 5. This shows the construction procedure of a 2.5D scenic object. Our system lets a user select scenic objects with our implemented Intelligent Scissors [18], construct corresponding meshes using constrained Delaunay triangulation, and deform these meshes with user specified depth assignment to form the 2.5D scenic objects.

an initial flow pattern based on analysis of extracted stroke structures, placement density, and ink density.

4.1 Interactive 2.5D Scene Construction and Editing

Since painters of Chinese ink paintings generally use multi-perspective panoramic views to portray a scene and arrange scenic objects in different depth layers, this gives us the idea of creating 2.5D scenic objects and placing them at different depth layers. Fig. 5 shows the process of creating the rock mountain patch of "Heart Mirror". Since segmentation is hard to be fully automatic, a user first lays down a few scribbles. Our system use Intelligent Scissors [18] to create optimal pathes along edges based on these scribbles. Furthermore, painters depict the distinct textures and structures of scenic objects with strokes, these distinct borders indicate possible geometric boundaries. Then, we use the patch boundary and edges detected with the Canny edge detector to tessellate the segmented object patch with constrained Delaunay triangulation. After triangulation, we automatically assign the corresponding texture coordinate of all vertices. Finally, the user can intuitively and simply lay down a few height scribbles to assign height values on the patch, and our system automatically sets the height of boundaries to zero. The height of all vertices are interpolated based on the height scribbles using inverse distance interpolation [19]. After creating 2.5D scenic meshes, users can drag these patches aligned with the painting to create the virtual world. After constructing the virtual world, our system extracts strokes, postulates an initial flow field, classifies flow patterns, and groups strokes from the painting as described in the following sections.

4.2 Stroke Extraction and Representation

Strokes generally reflect flow states and artistic styles, and we must detect and identify them from a painting before animating them. We apply data-driven stroke extraction [17] to identify and locate strokes from the user-specified regions. When imagining flow motion depicted by strokes, we find that artists generally draw them with two components: global flow motion and local waving movement. Therefore, this work decides to record strokes as two-level sausage-style skeletal strokes [20] shown in Fig. 6. The first level expresses the rough flow motion, $\mathscr{B}^{flow}$, the second level describes the waving details, $\mathscr{B}^{skeleton}$, and the flesh represents the brush textural details, $\mathscr{M}$, where $\mathscr{B}^{flow}$ and $\mathscr{B}^{skeleton}$ are B-spline curves, and $\mathscr{M}$ is a triangular mesh. After locating strokes, we first apply the Ramer-Douglas-Peucker (RDP) algorithm [21], [22] to select a set of connection points from the medial axis of each extracted stroke and create a skeleton, $\mathscr{B}^{skeleton}$, by connecting segments fitting samples between two consecutive connection points. Furthermore, since the flow motion is global and changes slowly and smoothly, it should only contain low-frequency components. However, different drawing styles use different stroke lengths and shapes to describe the flow. *Ripple depicting (Ping-Po)* generally lays down short single-waving strokes tangent to the flow direction, and we approximate flow paths by fitting all samples to line segments. *Snaking (Chiu-Je)* and *Reticulating (Wan-Jin)* use long multiple-waving strokes to depict flow motion in a larger region, and we approximate flow paths by fitting all samples to a five-control-point B-Spline. In order to differentiate two separate fitting operations, we choose the length of strokes as the criteria. When the stroke length is shorter than a user selected threshold, T^{stroke}, we use line segment fitting; otherwise, we use a five-point B-spline fitting method. Finally, our system encodes the extracted boundary and texture as its flesh mesh, $\mathscr{M}$, for subsequent skeleton-based deformation proposed by Hsu

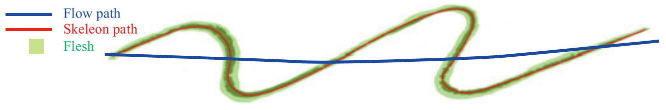

Fig. 6. We depict strokes as two-level sausage-style skeletal strokes [20] of a first-level flow path, a second-level skeleton path, and its appearance flesh.

et al. [20]. Next, we can use extracted strokes to postulate an initial driving field.

4.3 Data-Driven Initial Field Construction

Delinating (Gou-ran) depicts the flow direction, speed, and pattern as well as the surface structures such as ripples and waves with stroke pattern groups of different oscillations and ink dispersion densities in Chinese paintings. Consequently, our system first extracts drawn strokes and constructs an initial flow field accordingly which reflects artists' intention. Later, we plug the initial field into a procedural solver for maintenance of certain physical interaction with static and dynamic objects such as rocks, banks, and boats. We also provide editing tools to direct and adjust the field based on users' design. Painters generally use different numbers of strokes with different orientations, oscillations, and ink dispersion densities for flow characteristics and surface structures. For example, they portray stronger streaming regions with a larger number of vehemently oscillating dark strokes. Moreover, they also align them to be parallel or perpendicular to the flow direction for streaming motion or ripple and wave propagation. Since this work aims at animating these strokes in a manner postulated from these extracted strokes, we decide to generate a driving field with its direction parallel/perpendicular to local flow paths and magnitude computed based on extracted stroke properties. Our system builds artistically postulated flow field as follows. The user first indicates the relations between the flow path orientation and field direction because this relation is easily determined by humans instead of computers. Our system creates a grid of $N_x \times N_y$ over the user-specified flow region where N_x and N_y determine simulation resolution and are set to 256 respectively for all our examples. We estimate the global mean ink dispersion density over all extracted strokes, and each stroke selects N^{sample} samples in the flow path parameter space where N^{sample} is a user selected parameter. For each active cell containing at least one stroke sample, our system computes its field direction by weighing the average of the direction of path samples falling in the cell. For each active cell, we select an estimation circle centering the cell with a radius of N^{cell} where N^{cell} is a user specified parameter to determine the stroke-affected extension. Our system computes the local sample dispersion density, flow direction variation, and mean oscillation magnitude of the cell by weighing the average of the corresponding values in the circle. Our system chooses a field strength proportional to all three terms with a user specific parameter, S^{field}. However, it is hard to construct a proper driving field based on these sparse active cells, and thus, we iteratively advance the known velocities to their neighboring unknown cells using the mean flow field of its eight-neighbor known cells. Next, we can use the postulated field to group extracted strokes into stroke pattern groups.

4.4 Flow Pattern and Stroke Pattern Group Extraction

Delinating (Gou-ran) uses sets of strokes to portray flow patterns and surface structures, and driving and animating individual strokes independently may not fully convey the dynamic spirits. Important characteristics for a stroke pattern group are its flow pattern, its occupied area, and its component strokes with their respective lengths, oscillation magnitudes, and ink dispersion densities. When observing water flows in landscape paintings, painters generally use different types of strokes to express two different flows, streaming and circulating, which are denoted as $\mathscr{F}^{stream}$ and $\mathscr{F}^{circulate}$ for corresponding regions respectively. Thus, our stroke clustering scheme consists of two stages, flow pattern classification and stroke characteristics clustering, as shown in Fig. 4. Before clustering, we first give a definition of our stroke pattern group as

$$SPG(f^{type}, s^{number}, w, h) = \left\{ \begin{array}{l} f^{type} : \text{the flow pattern type}, \\ s^{number} : \text{the number of strokes}, \\ w : \text{the width of bounding box}, \\ h : \text{the height of bounding box} \end{array} \right\}. \tag{1}$$

And this work also denotes singularities as places whose flow field or field derivative is undefined. We first start clustering by decomposing the initial flow field with the Poincaré index theory [23] for recognition of hyperbolic, parabolic, and elliptic sectors. Then, our system uses each sector composition to compute the index of singularities and locates circulating regions, $\mathscr{F}^{circulate}$. After identifying and removing all $\mathscr{F}^{circulate}$, the rest of the flowing region is the streaming region, $\mathscr{F}^{stream}$. For stroke clustering, our system locates the circulating strokes inside the regions which have their flow paths similar to elliptical arcs in the neighborhood inside the bounding box. At the same time, we also connect those detected singularities as the structural line singularities. Furthermore, we can simply cluster all strokes in $\mathscr{F}^{stream}$ as a SPG group, but painters locally express the streaming flow state by first laying down main strokes which are generally long and curved to express the global flow state and then drawing auxiliary strokes along with main strokes to express local flow variation. In order to better portray flow characteristics, we cluster strokes into streaming stroke pattern groups as follows. We first sort all strokes inside the streaming region according to their length and select a set of main stroke candidates, $\mathscr{S}^{main} = ms_i$, based on a user specific length threshold, T^{main}. We take the remaining longest candidate stroke in the list, and sweep a mask of $N^{stroke} \times N^{stroke}$ along its flow path to create the stroke pattern grouping region. Our system puts strokes intersected with the main stroke along the field into the group with the following steps: 1) We compute the bounding box of all strokes and select those colliding with the grouping region as member candidates. 2) For each candidate, our system advects their end control point of the flow path forwardly along the flow field using Eq. (3) in Section 5.5. 3) We let the rest of control points take the location of its consecutive descendant control point and update the corresponding flow and stroke paths using the skeletal coordinate [20]. 4) Our system applies the algorithm proposed by Lou et al. [24] to determine whether the candidate intersects with the main stroke. 5) If there is an intersection, our system puts the stroke into the stroke pattern group of the candidate; otherwise we keep advecting the stroke until it leaves the grouping region. Then, our system picks up the remaining longest stroke and repeats the process until there is no candidate stroke left in the list. Our system aligns those ungrouped strokes to select main strokes using non-rigid registration algorithm [25] for estimation of their matching errors to the main stroke and put them into a group whose combination of matching error and distance to its main stroke is the smallest. Finally, we sort the strokes inside a group according to their length, choose relatively long strokes as transition strokes, and label the rest as decoration strokes. After these steps, we have a 2.5D virtual world built from the Chinese painting which is ready for 3D walk-through, an initial flow field for constructing the plausible stroke driving field, and a set of stroke pattern groups which are animated and rendered along the field to better convey dynamic spirit of a water flow. We give the details of field construction and modification and flow stroke animation in the next section.

5 Interactive Field Editing and Animation Generation

Our system uses a flow field for spatiotemporally coherent stroke animations, but the initially postulated field may fail to have proper interaction with static and dynamic objects. Therefore, we provide a procedural solver which takes the postulated flow pattern as initial conditions and uses objects as boundary conditions to create a spatiotemporally coherent field. However, the solver has difficulties in controlling the generation of circulating structures, and therefore, we use a field synthesis technique to create whirling and radiant structures on the designed location. Additionally, we also provide two editing tools to adjust the constructed flow fields for better matching the designer's intention in animating strokes. Since this work uses stroke groups to express flow states locally, our system dynamically adds/removes extracted stroke pattern groups into/from the water flow with fading effects and animates strokes with the flow fields for spatio-temporal coherent stroke animation. This section gives details about these field editing and animation techniques.

5.1 Procedural Streaming Field Generation

The data-driven initial flow field can artistically drive and animate strokes, but it cannot prevent strokes from penetrating static and dynamic objects. We decide to generate an artistic plausible field with a procedural 2D Navier-Stokes solver due to the following reasons. The solver can use the initial data-driven artistic field as driven forces to postulate the painter's intention on depicting flow motion. The solver can take static and dynamic objects as boundary conditions in field generation. Although we can also apply a field synthesizer, it is hard to take static and dynamic objects into consideration. The solver can let a user direct and adjust the field with our sketch-based field adjustment system described in the *Interactive field adjustment* paragraph. The procedural solver estimates the field on a 2D Euler grid with fluid simulation, but we currently do not support self-intersecting regions. Our system solves the Navier-Stokes equations: $\nabla \cdot \vec{\mathbf{u}} = 0$ and $\frac{\partial \vec{\mathbf{u}}}{\partial t} + (\vec{\mathbf{u}} \cdot \nabla)\vec{\mathbf{u}} = -\frac{1}{\rho}\nabla \mathbf{p} + \nu \nabla^2 \vec{\mathbf{u}} + \vec{\mathbf{f}}$ where $\vec{\mathbf{u}}$ is fluid velocity, t is time, ρ is fluid density, $\mathbf{p}$ is pressure set to zero, ν is viscosity, and $\vec{\mathbf{f}}$ is acceleration induced by external forces. We adapt the semi-Lagrangian fluid solver [1] for its simplicity and efficiency. After constructing the virtual world, our system projects the 2.5D scenic objects onto the flowing plane to create static object borders. Our system chooses free boundary conditions for the flow entrance/exit boundaries and circulating boundaries and uses fixed boundary conditions for those projected borders. Furthermore, we design three extra driving forces for the solver to achieve different effects, and the details are as follows:

1) *Stroke-based initial flow pattern*: The initial flow pattern encodes painter-intended artistic flow motion, and our system includes it through adding an external force term at corresponding grid points.
2) *Dynamic disturbance for moving objects*: When dynamic objects move, our system pushes water away by a force whose location is the center of the object and magnitude is proportional to the relative velocity between the object and flow. We provide a user specified parameter, $S^{dynamic}$, to adjust the force magnitude.
3) *Interactive field adjustment*: Although the initial field postulates painters' flow intention, the user may have other ideas about the flow motion. Our system lets the user direct and adjust the flow patterns by laying down a few scribbles which are transformed as external forces applied at the intersected grid cells. Direction is the tangent of the scribble, and magnitude is proportional to the moving speed of the scribble.

We have implemented our procedural solver with the CUDA framework, and the entire system can dynamically estimate a flow field of 256×256 in real time. This allows the user to examine the instructed results for immediate feedback.

5.2 Circulating Field Generation

Painters sometimes use whirling and radiant structures to portray flow relations to barriers and express dynamic spirits where a whirling structure is a field flowing around its center and a radiant structure is a field flowing out of its center. Although we can also use external force term to construct these structures, the interaction with other flowing cells complicates the process to make it hard to have the desired form and place at the desired location using the Euler grid solution [26]. Therefore, our system uses field synthesis adapted from the singularity-field addition technique [27], [28], [29] to replace the flow pattern in the circulating regions. Our system uses the detected circulating regions as places of combination, identifies circulating strokes for the field boundaries, and connects singularities as its central singularities. We can construct a field, $\vec{\mathbf{U}}^{Laplace}$, with the constraints of boundaries and singularities using $\vec{\mathbf{U}}^{Laplace}(\overline{p}) \cdot \nabla\omega(\overline{p}) = 0$ where $\omega(\overline{p})$ is a chosen 2D *scalar harmonic function* and satisfies the Laplace equation, $\triangle\omega = \frac{\partial^2\omega}{\partial x^2} + \frac{\partial^2\omega}{\partial y^2} = 0$. After identifying boundaries and central singularities, our system sets line singularities' value to 1 and the boundaries' value to 0 and solves the whirling and radiant fields by compositing harmonic functions [30] and interpolating their values. Finally, our system constructs whirling and radiant fields using the derivative of ω in the major and minor directions respectively. Fig. 7 shows the process of constructing a whirling field, and Fig. 8 shows

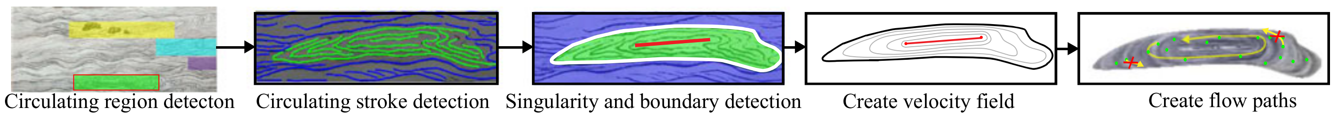
Circulating region detecton
Circulating stroke detection
Singularity and boundary detection
Create velocity field
Create flow paths

Fig. 7. This illustrates generation of the whirling flow field using harmonic functions. Our system determines possible circulating regions with the Poincaré index theory [23], identifies circulating strokes, locates their boundaries whose Harmonic value is set to 0 and central singularity whose Harmonic value is set to 1, solves the field with a composite harmonic solution, and finally, traces random initial starting points for path tracks.

the intermediate results of our driving field constructor for "Red Cliff".

5.3 Optional Singularity Removal

Although our procedural solver can generate an artistically plausible field, it may also create undesired whirling and crease structures to induce chaotic and unpredictable movement of strokes. Our system analyzes their causes and provides optional tools to remove these structures.

Tensor Field Smoother: Although artists may intentionally add whirling structures as described previously, undetected ones cause unexpected circulation and large deviation to animated strokes. When analyzing causes, the field of a whirling structure circulating around a center in either clockwise or counter-clockwise results in undefined derivatives at the center. We denote these undefined derivatives as concentric singularities. It is formed by imbalance of exerted forces from different parts of the water flow to create a location where mathematically the derivative is undefined. Although counter forces can be applied to reduce the possible generation, but it cannot completely remove them. Since the visual representation of a tensor field is more natural and smoother than a one-directional vector field [27], our system transforms a vector field, $\vec{\mathbf{U}}$, to a symmetric bidirectional tensor field as $\mathbf{T}(\overline{p})\begin{pmatrix} \vec{\mathbf{T}}_x(\overline{p}) \\ \vec{\mathbf{T}}_y(\overline{p}) \end{pmatrix} = \begin{pmatrix} \vec{\mathbf{U}}_x(\overline{p}) & \vec{\mathbf{U}}_y(\overline{p}) \\ \vec{\mathbf{U}}_y(\overline{p}) & -\vec{\mathbf{U}}_x(\overline{p}) \end{pmatrix}$ where $\vec{\mathbf{T}}_x$ and $\vec{\mathbf{T}}_y$ are a tensor vector pair. Our system selects the one which is better aligned with the user-specified global direction, $\vec{V}^{global}$, of water flows as our field. After transformation, most concentric singularities could be eliminated.

Laplacian Smoothing Operator: The flow directions of a crease structure is so disorderly that the flow field has a visible boundary, where flow field is undefined, and directs strokes to collide into each other. We denote the boundary as crease singularities. While analyzing the causes, crease singularities are caused by the interaction of flows with opposing directions and be detected by judging whether the flow direction of two neighboring cells are opposing among themselves or not. Our system applies a 2×2 mask over all field cells and label them as a crease singularity, $\overline{g}_i^{singulairty} = (x_i, y_i)$, if the angle difference between any two mask cells is over 90 degrees. After labelling, we create a Laplacian smoothing region, $\mathscr{G}$, consisting of a singularity operation region, $\mathscr{G}^{singulairty}$, and an extended smooth field region, $\mathscr{G}^{extend}$.

We combine all connected singularities to form $\mathscr{G}^{singulairty}$ and group cells whose distance to $\mathscr{G}^{singulairty}$ is within a user-specified radius, $N^{Laplacian}$, as $\mathscr{G}^{extend}$. For clarity, some notations are given: $\overline{g}$ is a grid point in $\mathscr{G}$ which can be either $\overline{g}^{singulairty} \in \mathscr{G}^{singulairty}$ or $\overline{g}^{extend} \in \mathscr{G}^{extend}$. For each singularity in $\mathscr{G}$, the Laplacian smoothing operator recomputes its flow field, $\vec{\mathbf{U}}(\overline{g}^{singulairty})$, via the existing smooth field in $\mathscr{G}^{extend}$ in the following manner. Our system transforms the Laplacian smoothing region into a triangular mesh to construct a one-ring structure, $\mathscr{N}(\overline{g})$, for a point, $\overline{g}$. We apply the Laplacian operator in $\mathscr{G}$ as $\mathscr{L}(\vec{\mathbf{U}}(\overline{g}_i)) = \vec{\mathbf{U}}(\overline{g}_i) - \sum_{j \in \mathscr{N}(\overline{g}_i)} w_{ij} \vec{\mathbf{U}}(\overline{g}_j) = 0$ where w_{ij} is the mean weight of an edge, $\{\overline{g}_i, \overline{g}_j\}$, and is estimated with the mean value weighting scheme [31] as $w_{ij} = \frac{\tan(\alpha_{ij}/2) + \tan(\beta_{ij}/2)}{\|\overline{g}_i - \overline{g}_j\|}$ where α_{ij} and β_{ij} are the angles made by edge$\{\overline{g}_i, \overline{g}_j\}$ and its neighboring edges at $\overline{g}_i$. Our system computes the velocity field at $\overline{g}_i^{singulairty}$ by solving the Laplacian linear system of the previous Laplacian operator for singularity elimination. Users can first use the procedural solver and circulating field synthesizer to automatically generate plausible flow fields in a painting. Then, they can further design the fields with our provided force-field director, tensor field smoother, and Laplacian smooth operator. Next sections provide the details in dynamically activating and deactivating stroke pattern groups with the estimated placement and removal time and location of all pattern groups using fading effects and animating them along the user designed flow fields.

5.4 Streaming Stroke Pattern Group Placement and Timing Registration

Painters traditionally arrange flow strokes in a left-right manner and draw stroke groups to depict flow states and dynamics using continuous long strokes to convey senses of continuity and endlessness, and locally varied short strokes to portray local flow states. Therefore, our system is designed to let main strokes flow across the entire animated region for continuation of water flow and animate other shorter strokes locally to imitate local flow states and

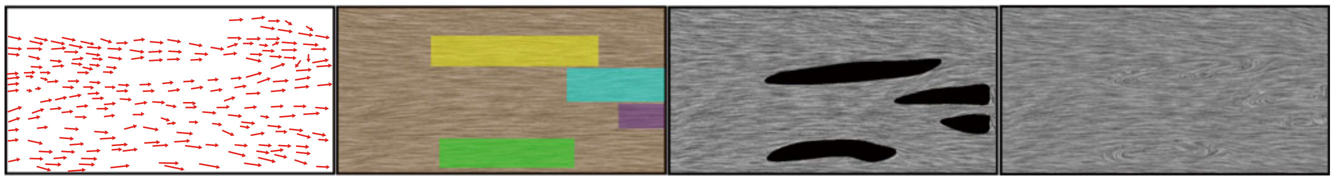

Fig. 8. These are the intermediate flow-field generation results from "Red Cliff" with 1) the initial velocities using extracted strokes, 2) the initial flow field interpolated from the initial velocities, 3) the flow field generated by solving the Navier–Stokes equations, and 4) the complete field after adding whirling fields computed by harmonic functions.

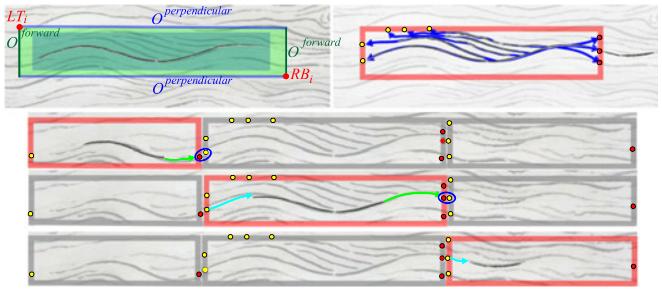

Fig. 9. This shows the process of identifying the destined region of pattern groups, registering strokes with its corresponding destined region to determine flow-in and flow-out valves (yellow and red dots), and connecting consecutive destined regions by pairing neighboring valves based on the flow field.

dynamics. Our system must create localized spatiotemporal frameworks based on groups for animating transition and decoration strokes. In the first, we identify a destined region for a pattern group by constructing the axis-aligned bounding box of all strokes in the pattern group and extending the box with $O^{forward}$ to the left and right boundaries and $O^{perpendicular}$ to the top and bottom boundaries as shown in the top-left of Fig. 9. A destined region is where a group is activated and animated. As shown in the top-right of Fig. 9, the stroke of destined region can be described with its top-left corner and bottom-right corner as $\{\overline{LT}_i, \overline{RB}_i\}$ where i is the index of the group, and uses it as the spatiotemporal framework to register the released location and time of all group strokes. We advect backwardly the end control point of all group strokes using Eq. (3) until they move out of the destined region. Our system records the moving-out location as the placement location, $\overline{P}_{i,j}^{placement}$, and the number of advected time steps as the placement time, $ST_{i,j}^{placement}$, where j is the index of a pattern stroke in the ith group. Similarly, we also advect all strokes forwardly to record its removal location and time, $\overline{P}_{i,j}^{removal}$, and $ST_{i,j}^{removal}$. Our system considers these placement and removal location as flow-in and flow-out valves represented as yellow and red dots in Fig. 9, and we globally connect consecutive destined regions by pairing up these valves based on the estimated field and their spatial location in a manner from left to right. While pair up flow-in and flow-out valves, we update the corresponding group states including the pattern placement location and time, $\overline{P}_i^{pattern}$ and $ST_i^{pattern}$, along with the flow speed, flow orientation, and ink density and register all other strokes based on updated states. These connections and registrations avoid sudden change across destined region boundaries to avoid animation artifacts and keep flow smooth.

5.5 Streaming Flow Stroke Animation

Our system associates all strokes with an initiated location, $\overline{P}_{i,j}^{initiate}$, an initiation counter, $C_{i,j}^{initiate}$, to indicate when to initiate the stroke, and a life counter, $C_{i,j}^{life}$, to indicate when we deactivate it from animation for proper temporal relationship. There are multiple mechanisms designed to deliver the vivid water flow to audiences.

Advance Active Strokes with the Field. At the beginning of every frame, our system decreases $C_{i,j}^{initiate}$ of all strokes by one. If $C_{i,j}^{initiate} \leq 0$, we decrease $C_{i,j}^{life}$ of all strokes by one and advect them forwardly as shown in Fig. 10

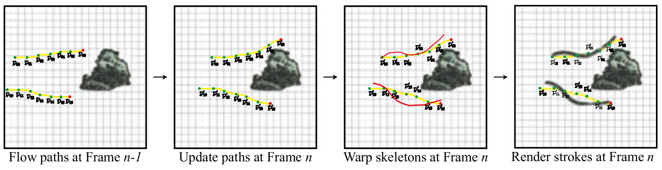

Fig. 10. This illustrates the process of advecting and animating the stroke with the flow field. The end control point of the flow path is advected with the flow field and the rest of control points take the location of its consecutive descendant control point. The skeleton path and flesh is deformed based on the newly deformed flow path coordinate described by Hsu et al. [20].

$$\overline{e}_n = \overline{e}_{n-1} + \vec{\mathbf{U}}(\overline{e}_{n-1})\Delta t \tag{3}$$

where Δt is a user specific constant for flow animation, $\overline{e}_n$ is the stroke end position after n-step advection, and $\vec{\mathbf{U}}$ is the static flow field. After updating the end location, the rest of points take the previous point location. We can compute the flow path, skeleton path, and flesh of all strokes based on the newly updated control points accordingly using the skeletal stroke deformation method [20].

Deactivate Strokes. Our system must dynamically stop drawing strokes to prevent staggering strokes at boundaries, remove chaotically crossing strokes among strokes, and remove extra processing cost. We deactivate a stroke from the scene when it encounters obstacles such as rocks and boats, reaches boundaries such as whirling or radiant structures, or uses up its life time.

Activate New Stroke Pattern Groups. Since strokes may move out of the flow region and become deactivated from the scene, our system must activate them for maintenance of endless flow motion and stroke densities. For a starting destined region, once its main stroke moves close to its right boundary, our system chooses a random $C_{i,M}^{initiate}$ for the main stroke, sets up $C_{i,M}^{life} = ST_{i,M}^{placement} + ST_{i,M}^{removal}$, and mutates its initial location, $\overline{P}_{i,M}^{initiate}$, by perturbing its y of $\overline{P}_{i,M}^{placement}$ with a random amount. The index, M, indicates the index of the main stroke in the group. After activating the main stroke, our system uses the registration time and placement location to animate other strokes. If our system animates all strokes across their destined region, there are a large number of intersections among strokes. Therefore, we have separated localized strokes into transition and decoration strokes. We have also let transition strokes flow across the destined region and decoration strokes appear and disappear shortly inside the region. For a transition stroke, we set up $C_{i,t}^{initiate} = C_{i,M}^{initiate} + (ST_{i,t}^{placement} - ST_{i,M}^{placement})$, $C_{i,t}^{life} = ST_{i,t}^{placement} + st^{Random}$, and its location as $\overline{P}_{i,M}^{initiate} + (\overline{P}_{i,t}^{placement} - \overline{P}_{i,M}^{placement}) + \overline{p}^{Random}$ where t is the index of a transition stroke, st^{Random} is a random scalar value, and $\overline{p}^{Random}$ is a random vector. Since our system animates decoration curves only in their drawn neighborhood to avoid chaotic situations, our system randomly initiates $C_{i,d}^{initiate}$, $C_{i,d}^{life}$, and $\overline{P}_{i,d}^{initiate} = \overline{P}_{i,M}^{initiate} + (\overline{P}_{i,d}^{placement} - \overline{P}_{i,M}^{placement}) + \overline{p}^{Random}$ where d is the index of decoration

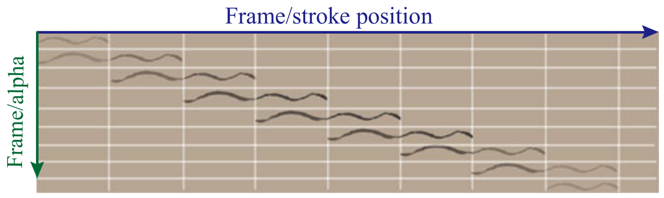

Fig. 11. The ink density of a stroke changes with the accumulated appearance time. The left axis shows the ink density from light to dark and from dark to light due to the accumulation of stroke appearance time. The top axis shows the movement of the stroke according to a flow from left to right along with appearance and disappearance of sections of a stroke.

strokes. For a linked destined region, once the main stroke from the ascendant region touches its left boundary, our system initiates the main stroke with $C^{initiate}_{i,M} = 0$, $C^{life}_{i,M} = ST^{placement}_{i,M} + ST^{removal}_{i,M}$, and use its intersection point, $\overline{P}^{initiate}_{i,M}$, as its placement location.

Animate the Density of Ink Dispersion. Our system has precomputed a 1D ink alpha map with $\eta = 10, 12, 14, \ldots 180, 178, \ldots, 0$. Then, our system adapts the fading in and out animation rule to render a stroke in light-to-dark and dark-to-light cycles for the addition and removal process by using the mapping as shown in Fig. 11. This can prevent popping up artifacts of strokes and also draw the audience's attention to different parts of the water surface with high-ink-density strokes to emphasize the water flowing effect. Furthermore, when flow strokes are long, fading in/out strokes makes the entire painting look sparse and have popping artifacts. Therefore, two mechanisms are used to overcome these problems: segments of a complete stroke are faded in/out consecutively where segments are located by detecting the peaks and valleys of the stroke oscillation pattern; the fading in and out period depends on the length of the fading stroke, i.e., main and transition strokes have a longer fading period, and decoration strokes have a shorter fading period.

Initiate the Animation. In order to mimic the painting, our system directly uses all extracted strokes to initiate the animation. We set up $C^{initiate}_{i,j} = 0$, $C^{life}_{i,M}$ as their respective removal time, $ST^{removal}_{i,M}$, and $C^{life}_{i,t}$ and $C^{life}_{i,d}$ with a random number. Finally, we set up all stroke locations at the extracted places.

5.6 Whirling and Ripple Stroke Animation

When strokes flow into a whirling structure, advection with a finite time may induce vibration to stroke animation and unexpected flowing out of the structure to cause chaos around the structure. Additionally, the entering strokes move around the outside boundary and do not move into the interior of the structure. In another situation, when ripples radiate outward from the center, advection causes stroke expansion to induce aliasing artifacts on the rendering results. Therefore, our system provides different animation mechanics for both structures as follows.

Animate Whirling Strokes. When constructing the whirling field, those strokes inside the field are labelled as whirling strokes, and extracted and stored as skeleton strokes. Since our animation runs at tens FPS, these large time steps induce numerical errors while advecting these strokes and cause

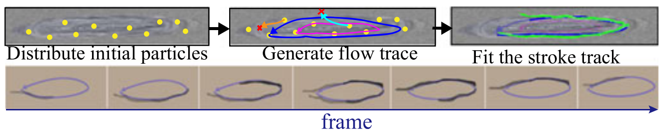

Fig. 12. The top shows the process of track construction and stroke registration by randomly distributes starting particles, advecting particles to construct flow traces, and registering whirling strokes to the closest trace. The bottom shows the time line of a whirling stroke animated along the track with time increment from left to right.

vibration and flowing-out artifacts. Therefore, we precompute a set of whirling tracks for animating them instead of advecting them freely with the whirling field for artifact removal, such as the Fig 12. shown. Our system first places a set of particles inside the structure randomly and advects them with the whirling field with a very small time step to avoid integration errors until connecting them back to the starting location. If particles flow out of the field, we remove their corresponding tracks out of the list. After tracking construction, we then select a set of representative tracks based on even distribution inside the structure without colliding into each other. During animation, when a stream flowing stroke hits the whirling structure, the stroke fades out, and our system fades in a whirling stroke selected from those extracted inside the structure on a starting location of a randomly chosen track which is close to the collision point. According to the whirling field strength, our system moves the whirling stroke along the destined track. Additionally, our system can swap the strokes to a track next to the current one randomly. Our system use the newly deformed flow path to reconstruct the final appearance of the stroke. When the life counter of a stroke reaches zero, the stroke may fade out if it is not connected with outside flow strokes. Otherwise, our system initiates a stream flow stroke to flow out of the whirling structure.

Animate Ripple Strokes. When constructing the radiant field, the ripple strokes are also extracted and stored as skeleton strokes in the Fig. 13. Our system identifies its centers and sort them based on its spatial distance from the center to the ripple center. Since radiating outward may result in overstretching of stroke textures, our system limits the amount of radiation by using consecutive ripple strokes. Our system randomly initiates a ripple stroke centered at a random location near the ripple center using the most inner stroke. We advect all control points of the stroke with the built radiant field outward and then reconstruct the appearance with the newly deformed flow path. When the stroke

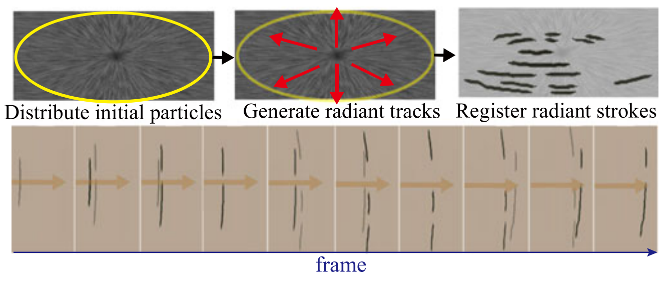

Fig. 13. The top shows the process of stroke registration by uniformly distributing starting particles along the central singularity boundary, advecting particles to create radiant traces, and registering radiant strokes to a trace passing through their center. The bottom shows the time line of a radiant stroke animated along the track with time increment from left to right.

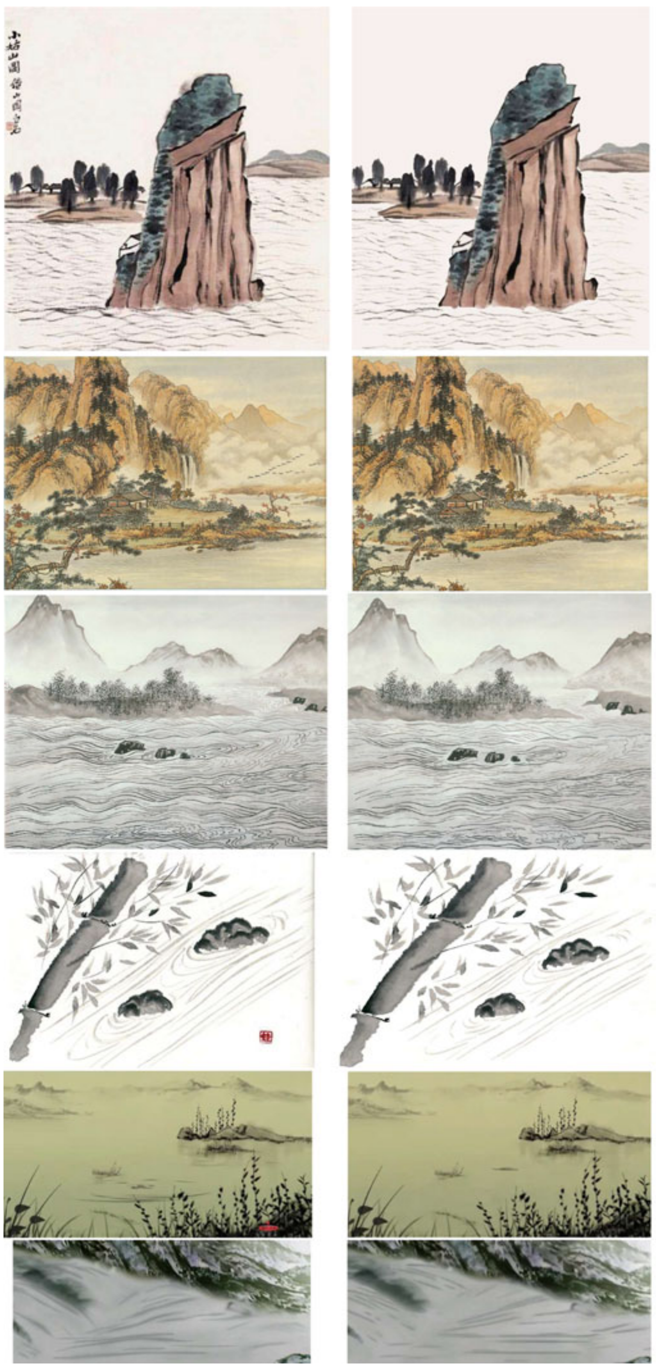

Fig. 14. From top to bottom shows the snapshots of "Heart Mirror", "Autumn Wind", "Red Cliff", "River Song", "The Legend of Shangri-la", and a regenerated video-based water simulation animation [16]. The left columns show the original pictures, and the right columns show the first frame.

reaches the spatial distance to the next ripple stroke, our system blends and swaps the stroke to the next ripple stroke. If the stroke is the outermost one and reaches the radiant boundary, the stroke fades out.

6 Results

Since our work focuses on animating water flows in an existing Chinese painting, the following sections demonstrates the animation results when applying to several Chinese paintings, Japanese Sumi-e, and artist-made Chinese painting videos. Additionally, we can also extend the concept to transfer extracted stroke groups to other paintings for augmenting the flow, and we give the details and results in this section.

TABLE 1
This Shows the Statistics for "Heart Mirror", "Autumn Wind", "Red Cliff", and "Moving of the Riverside Scene at Chingming Festival"

	# Str	# Grp	# Obj	2.5D	Flow	Rendering
Mirror	97	3	5	12 m	0 m	206.1 fps
Autumn	65	4	8	37 m	0 m	255.6 fps
Red	277	15	6	16 m	3 m	64.6 fps
Moving	61	5	2	45 m	2 m	263.2 fps

# Str denotes the number of extracted strokes for each painting, # Grp describes the number of clustered stroke pattern groups, # Obj illustrates the number of 2.5D objects in the world, 2.5D denotes the time to construct the 2.5D world including constructing 2.5 patches and aligning them to form the world, Flow describes the editing time to create the demonstrated field, and Rendering shows the frame rate of interactive walk-through operations.

6.1 Data-Driven Results

All rendered results and statistics presented in this section are collected based on a computer with Nvidia GTX 760, Intel i7 3,820 and 8 GB main memory. As described previously, this work develops a data-driven interactive system to construct the 2.5D virtual world of an existing Chinese painting with animated flow stroke pattern groups for interactive walk-through. The accompanying video provides a sense of the user interface for creating the 2.5D animated virtual worlds for existing paintings, as well as a demonstration of the animated results. However, due to the length and space limitation, Fig. 14 shows selected results, and complete results can be found in the supplemental web site.[1] Table 1 summarizes the number of extracted strokes, clustered pattern groups, 2.5D objects for the four Chinese paintings, "Heart Mirror", "Autumn Wind", "Red Cliff", and "Moving of the Riverside Scene at Chingming Festival" along with the time that it takes a user to create the 2.5D virtual worlds, the demonstrated flow fields, and the playback speeds. Painters depict these worlds in multiperspective views and thus, we used our interactive 2.5D scene constructor to segment scenic objects and create 2.5D patches and arrange them to construct the virtual world of all demonstrated paintings. Generally speaking, the construction of 2.5D world takes the large majority of the design time. In all cases, the animated paintings take from about 10 minutes to a little under an hour to create. Note that two of the animated pictures whose timings are presented above, "Mirror Heart" and "Autumn Wind", are created by a complete novice user who only has a few minutes of instruction before working on the pictures. Furthermore, a user can use our data-driven field constructor to edit the field and adjust the structures in the field. Data-driven generated fields are proper for driving strokes, and the time is needed for addition of whirling and radiant structures. We implement the 2.5D scene constructor with C++ and field editing techniques including Laplacian smoothing, tensor smoothing, whirling addition, and radiant addition with C++. These editing tools would take a few

1. web site: http://graphics.csie.ntust.edu.tw/pub/ChinesePainting/main.html

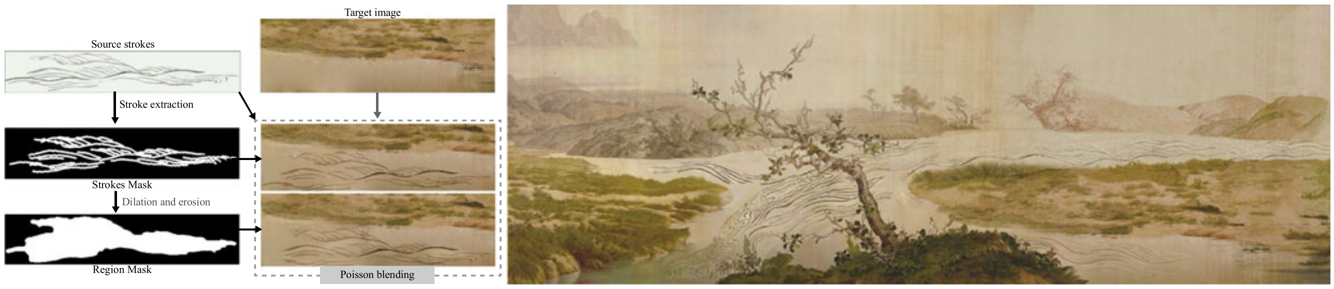

Fig. 15. The left illustrates the process of compositing a stroke pattern group onto another painting by constructing individual stroke masks, combining individual masks into a group mask, and compositing with the Poisson blending method [32]. The right shows a snapshot of an animation generated by semiautomatically placing and transferring extracted strokes onto “Bai Jun Tu”/‘one hundred stallions” drawn by Giuseppe Castiglione (Lang Shihning).

seconds to finish each operation. These parts of our code presently take no special advantage of graphics hardware, but all of the operations could be readily mapped to GPU for enhancing frame rates. Furthermore, in order to have dynamic objects interacting with our flow and real-time stroke-based field editing, our procedural flow field solver is implemented with Nvidia CUDA. The frame rate of procedural field computation is about 60 frame per second (fps). Finally, we implement our interactive walk-through renderer with GLSL to take full advantage of the parallel rendering abilities of GPUs for interactivity.

“Heart Mirror” uses long and curly strokes in the *Wan-jin* style to depict a part of the Yangtze river when flowing through a small island. Our system animates strokes in groups to give better description of the surface structures. Furthermore, our system estimates the driving field to provide a strong and energetic sense of the flow. When passing through the island, the procedural solver allows strokes to deviate and avoid colliding into the island. “Autumn Wind” uses long and curly strokes in the *Wan-jin* style to describe a smooth river flowing inside a valley surrounding by mountains. Additionally, the painter also uses a few *Chiu-je* strokes to describe the ripples around the rock. Our system can group these wave net-like strokes and let them flow out together to further emphasize their long lasting and peaceful spirits. “Red Cliff” uses large-oscillation composite *Ping-po* strokes and *Chiu-je* whirling strokes to depict the vivid streams and express exciting spirits in the famous relic along the Yangtze river. Our group placement and timing registration and animation mechanism provides mechanics to maintain the magnitude of oscillations, describes the complex surface structures, and avoid chaos during animation. Additionally, the whirling strokes circle around their centers without colliding into each other. “Moving of the Riverside Scene at Chingming Festival” uses composite *Ping-po* strokes to depict the complex wave surfaces to express the liveliness of water. The complex strokes are hard to animate manually, and thus, the video built for Expo replaced them with simple and independent strokes. These simple strokes cannot express the complex flow patterns and structures existing in the original painting. Our system can still better depict the spirits and dynamics by animating the original strokes with spatiotemporal coherence.

Furthermore, in order to examine the effectiveness of our system, we also applied our algorithm to other types of drawings including two Japanese Sumies, “Heron and Lotus Pods” and “River Song”. We also collected several flow animations created by artists and the video-based flow animation system. These animations include “The Cowboy’s Flute”, “The Legend of Shangri-la”, and a video-based water simulation example. Since the data-driven stroke identification algorithm [17] can correctly extract the desired strokes even when strokes are thick and with light ink dispersion densities as the ones presented in “The Cowboy’s Flute”, our system can cluster them based on the postulated flow field to express the water flow.

6.2 Interactive Stroke Pattern Group Transfer

Blanking (Liou-bai) uses other painted objects to imitate the existence of water, but it requires viewers’ imagination to depict the flow. Moreover, it is hard to find proper object movements to animate water flows. Although stroke pattern groups encode the flow painting style, portraying flow patterns and surface structures requires a high-level designing and thinking process and is hard to be automatic. Therefore, instead of automatically placing them, we provide a simple selection and transformation tool to arrange extracted pattern groups to set up the desired water flows on another painting. However, directly compositing stroke groups induces seams because ink dispersion density and color differences between the original and composited paintings. Since the ink dispersion density and the color differences between the background and the strokes are the main elements that catch viewer’s attention, our system adapts the Poisson blending algorithm [32] to blend these groups seamlessly onto the painting as shown in the left of Fig. 15. After arranging a pattern group, we construct a blending mask, Ω, as the union of all individual stroke masks and its boundary, $\partial\Omega$. We seek a composite painting, $\mathbf{I}^{composite} = \mathbf{I} + \mathbf{I}^*$, where $\mathbf{I}$ is the unknown group blending region over the interior of Ω, and $\mathbf{I}^*$ is the rest of the blending painting, $\mathbf{I}^{blend}$. Our system conducts a guided interpolation along with the gradient of the blending painting, $\nabla\mathbf{I}^{blend}$, by $\min_{\mathbf{I}} \int\int_{\Omega} |\nabla\mathbf{I} - \nabla\mathbf{I}^{blend}|^2$ with $\mathbf{I}|_{\partial\Omega} = \mathbf{I}^*|_{\partial\Omega}$. We can determine $\mathbf{I}$ by solving the associated Euler-Lagrange equation, $\triangle\mathbf{I} = \triangle\mathbf{I}^{blend}\ over\ \Omega\ with\ \mathbf{I}|_{\partial\Omega} = \mathbf{I}^*|_{\partial\Omega}$, which must be solved along with Dirichlet boundary conditions [32]. However, individual stroke masks are too thin and small to induce blending seams and artifacts as shown in the left of Fig. 15. Therefore, our system combines individual stroke masks into a composite group mask with morphological operators, dilation and erosion. After composition, our system can run the same data-driven method to animate the transferred water flows.

Results. The right of Fig. 15 shows the snapshot of the animated transferred results for "One Hundred Stallions". In "One Hundred Stallions", vivid colors are used, but there is no stroke on the water flow region. Artists from Digimax inc. build up the 2.5D scene for exhibition, and we spend about 10 minutes to select, transform, and place groups onto the flow region. After placing, we can directly use the data-driven pipeline to generate the walk-through animation. The result shows that our tool can easily and successfully arrange stroke pattern groups extracted from "Red Cliff" onto the wiggling river bed and maintain the ink dispersion and color characteristics of strokes. The more results could be referred to our supplemental web site.

7 Experimental User Studies

In order to understand the performance of our data-driven system, we conduct three user studies to evaluate our plausibility of group-based animation, our delivery in flow dynamics with strokes, and our imitative ability to existing flow animations. Since our algorithm aims at generating flow animations for exhibitions, we first invite 40 amateur participants for our studies. They are students and workers from National Taiwan University of Science and Technology. They have normal or corrected normal vision. Their ages range from 21 to 36 years old with a mean of 23.25, and the participants comprise nine females and 31 males. All are volunteers. Later, we also want to collect opinions from Chinese painting specialists who have at least one year of Chinese painting experience, and we invite 24 specialist participants. They are artistic students and workers around Taipei City. They have normal or corrected normal vision. Their ages range from 40 to 60 years old with a mean of 52.04, and the participants comprise 18 females and six males. Their average experience in Chinese painting is 4.29 years. They are recruited with 16 US dollars for compensation. The amateur study is conducted under a computer with Nvidia GTX 760, Intel i7 3,820 and 8 GB main memory and the specialist study is under a laptop with Nvidia GTX 740, Intel i5 4,200U and 8 GB main memory. All studies are with an AOC i2757fm monitor of a resolution of $1,920 \times 1,080$, brightness of 250 $\mathrm{cd/m^2}$ and refresh rate of 60 Hz. Due to the limited length, the complete data sets and material are provided in supplemental web site.

7.1 Group-Based Flow Animation Plausibility

Since painters generally use sets of strokes to depict flow patterns and surface structures, it is important to generate more plausible animations by animating and driving stroke pattern groups instead of individual strokes independently. Therefore, we choose the three scenes, "Mirror Heart", "Autumn Wind", and "Red Cliff". For each scene, we construct two walk-through animations respectively that one uses stroke pattern groups, and the other uses individual strokes. The user study is conducted as follows: 1) Participants are asked to carefully examine the original painting for a minute especially on flow strokes. 2) We would show the original painting and proposed animation in the meantime, and the first flow animation is shown to participants and invite participants to grade it using five-point Likert scales where 1 to 5 ranks the degree of agreement to the criteria as very disagree, disagree, neutral, agree, and very

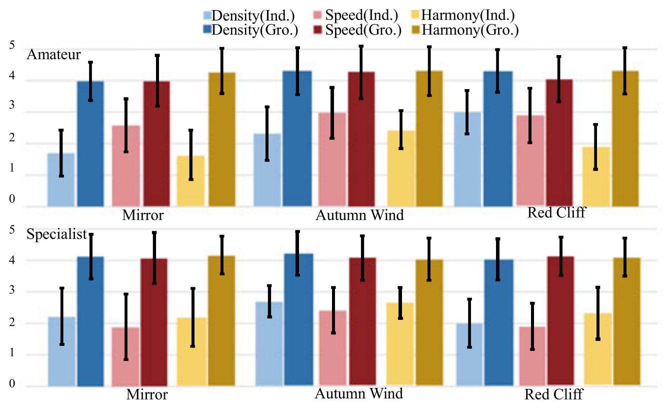

Fig. 16. The mean and standard deviation of the scores of the amateur and specialist groups for the three questions in our group-based flow animation plausibility study for three scenes, "Mirror Heart", "Autumn Wind", and "Red Cliff", when animating without/with grouping.

agree based on the following criteria: a) Whether strokes are animated and driven with a stroke density and spacing in a similar way as expected when watching the painting. b) Whether strokes are animated and driven with a speed in a similar way as expected when watching the painting. c) Whether are strokes animated and driven in a harmonious way. 3) The second flow animation is shown to participants for grading. 4) Repeat 1, 2, and 3 until finishing grading all three sets of animations. The order of each scene is predetermined, and the order of each sequence in a scene is chosen in a counter-balance manner to avoid the order bias. During the study, we record the scores for each animation and computed the mean and standard deviation of scores for the amateur and specialist groups in Fig. 16. This study aims at evaluating whether animating stroke pattern groups is plausible. Different gender and age groups judge differently. In order to limit the influence of these nuisance variables, i.e., age and gender, we distribute known and unsuspected variation sources among the units over the entire experiment using a blocking procedure. For instance, we use a randomized block factorial design (RBFD) [33] to assign the levels of nuisance variations randomly to the experimental units and compute their $F_{1,234} = 165$, 490, and 652 ($p < .05$) for the density, speed, and harmony scores respectively of the amateur group and $F_{1,138} = 228$, 223, and 207 ($p < .05$) of the specialist group according to Eq. 9.4-1 of the book written by Kirk [33]. The result shows that the animation mechanic is a significant factor for the density, speed, and harmony scores. Based on the scores, animating flow pattern groups is more acceptable than animating strokes independently, and we conclude that group-based animation is more acceptable and plausible. Furthermore, the scores are generally over 3.5 which means that the satisfaction of our animations are high when the participants watching the flow animations.

7.2 Comparison with Manual Animation

There exist several famous Chinese painting animations containing water flows. Generally, artists carefully place and animate strokes for consistent and pleasant flow stroke animations. This study aims at understanding whether our flow animation can be comparatively more acceptable when comparing to existing flow animations drawn by artists.

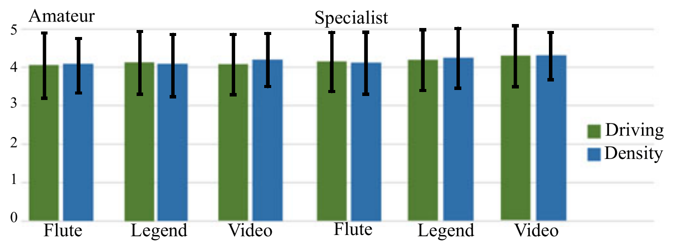

Fig. 17. The mean and standard deviation of the scores of the amateur and specialist groups for two questions when comparing against three manual animations, “The Cowboy’s Flute”, “The Legend of Shangri-la”, and “Video-based Brush Animation”.

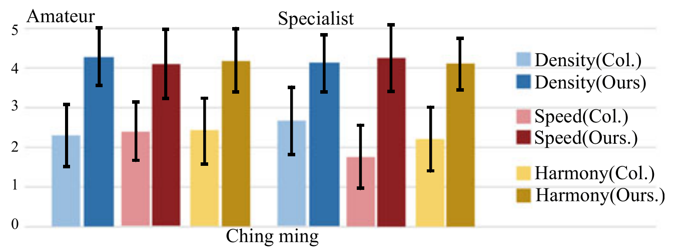

Fig. 18. The mean and standard deviation of the scores of the amateur and specialist groups for three questions in comparison between the data-driven and manually-created stroke animations, one collected from a web and the other built by our system.

Therefore, we collect three animations, “The Cowboy’s Flute”, “The Legend of Shangri-la”, and “Video-based Brush Animation”, and create corresponding animated video sequences with our system. The study is conducted as follows: 1) The instructor shows two animations to participants twice and asked them to focus on the stroke animations. 2) Two animations are put in the left and right monitors respectively, and participants are asked to grade it using Likert scales where 1 to 5 ranks the degree of agreement to the criteria as very disagree, disagree, neutral, agree, and very agree based on the following two questions: a) Whether strokes are animated and driven in a similar way for both animations. b) Whether strokes are animated and driven with a similar stroke density for both animations. 4) Repeat 1, 2, and 3 until finishing grading all three sets of animations. During the study, we record the scores for each animation and computed the mean and standard deviation of scores for the amateur and specialist groups in Fig. 17. Since we use our system to imitate the flow animation created by artists and other algorithm, we cannot find a comparison ground truth, and thus, we use one-sample t-test to examine whether our generated animations are similar to the original ones by having the null hypothesis (H_0) as the average score is equal to 4 ($\mu = 4$) and the alternative hypothesis (H_1) as the average score is greater than 4 ($\mu > 4$). We compute their $t_{40} = 2.12$, 2.04, 2.56, 1.74, 3.55, and 1.75 ($p < .05$) for the driving and density scores for three scenes of the amateur group respectively and their $t_{24} = 2.01$, 2.84, 2.32, 2.28, 4.41, and 2.58 ($p < .05$) of the specialist group. Therefore, we can conclude that our algorithm can truly imitate existing flow animations drawn by artists or generated by another algorithm.

7.3 Comparison Between Data-Driven and Manually-Created Stroke Animation

Animators create the “Moving of the Riverside Scene at Chingming Festival” animation by placing different strokes into their created 2.5D scene for consistent and pleasant flow stroke animations, and we would like to know whether our data-driven flow animation can better preserve the flowing spirits and dynamics when comparing to the manually-created animation. Therefore, we extract the stroke pattern groups from the original Chinese painting and use our Poisson-based transfer mechanic to transfer them onto the videos for flow animation. We follow the same procedure as the one discussed in Section 7.1 and use the same three criteria as described in Section 7.1. During the study, we record the scores for each animation and computed the mean and standard deviation of scores for amateur and specialist groups in Fig. 18. Similarly, we employ a randomized block design (RBD) [33] to avoid the influences of these limited nuisance factors and compute their $F_{1,78} = 70.9$, 155, and 95.0 ($p < .05$) for the density, speed, and harmony scores respectively of the amateur group and their $F_{1,46} = 104$, 39.1, and 70.9 ($p < .05$) of the specialist group according to Eq. 6.2-2 of the book written by Kirk [33]. Just like the previous section, our animation mechanic is again a significant factor for the score. Based on the scores, our animated water flows can better preserve the flowing spirits and dynamics than the manual flows.

8 Conclusion and Limitation

This work presents a data-driven water flow animation system to create interactive walk-through of an existing Chinese painting by animating water stroke pattern groups in a similar style of the original painting in a spatiotemporally coherent fashion. A user can build the 2.5D virtual world of a painting. Our system can semiautomatically extract the stroke pattern groups, compute an artistically plausible flow field, and coherently and consistently animates extracted groups with the field without flickering artifacts. Furthermore, we also provide several field editing tools to direct and adjust the postulated field for desired effects. At the end, a set of user studies are conducted to verify that water flow animations are smooth, coherent, and natural, and the style and spirit is consistent with the original painting. However, our system is not without limitations and there are a few future research directions. First, this work currently focuses on animating steady flows but cannot animate more dynamic water effects such as wave breaking and water spreading. We would like to extend our solver to incorporate these dynamic effects along with *Billows depicting (Lang-tou)* to animate more flow styles. Second, our data-driven mechanic depends on available strokes existing in the painting for conveying the rich and abundant sense, but while there are a limited number of strokes, our system cannot automatically create new strokes for richness and variability. We would like to allow the users to input their drawn and designed strokes for placing into the flows for relieving the limitation of our data-driven algorithm. Third, painters sometimes combine *Ink tinting (Hung-ran)* with *Delinating (Gou-ran)* but both effects have different mechanics to convey the flowing patterns. We would like to develop algorithms for extraction of the represented strokes and tinting regions and computation of driving fields to create reasonable animations. Fourth, there are painting styles using pattern patches to portray the flow, and it would require other mechanics to extract the representation, compute the corresponding driving field, and animate the representation. Finally, our stroke pattern group transferring is done interactively based on user's composition, and it is tedious and requires a large amount of manual work. We would like to analyze the statistics of stroke placement and composition for better style transfer.

Acknowledgments

This work was supported by the National Science Council of Taiwan under Grants MOST104-2221-E-011-029-MY3 and MOST105-2221-E-011-120-MY2 and by US National Science Foundation (NSF) of America under Award IIS-1619383. This work was supported by the National Science Council of Taiwan under Grants MOST 104-2221-E-011-029-MY3, MOST 104-2218-E-011-006, and MOST 105-2221-E-011-120-MY2. Chih-Yuan Yao is the corresponding author.

References

[1] J. Stam, "Stable fluids," in *Proc. 26th Annu. Conf. Comput. Graph. Interactive Techn.*, 1999, pp. 121–128.

[2] S. Strassmann, "Hairy brushes," *ACM Comput. Graph.*, vol. 20, no. 4, pp. 225–232, Aug. 1986.

[3] C. Curtis, S. Anderson, J. Seims, K. Fleischer, and D. Salesin, "Computer-generated watercolor," in *Proc. 24th Annu. Conf. Comput. Graph. Interactive Techn.*, 1997, pp. 421–430.

[4] S.-W. Huang, D.-L. Way, and Z.-C. Shih, "Physical-based model of ink diffusion in chinese ink paintings," *J. WSCG*, vol. 11, no. 1–3, pp. 1–8, Feb. 2003.

[5] N. S.-H. Chu and C.-L. Tai, "MoXi: Real-time ink dispersion in absorbent paper," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 504–511, Jul. 2005.

[6] N. Xie, H. Laga, S. Saito, and M. Nakajima, "IR2s: Interactive real photo to sumi-e," in *Proc. 8th Int. Symp. Non-Photorealistic Animation Rendering*, 2010, pp. 63–71.

[7] N. Xie, H. Laga, S. Saito, and M. Nakajima, "Contour-driven Sumi-e rendering of real photos," *Comput. Graph.*, vol. 35, no. 1, pp. 122–134, Feb. 2011.

[8] J. Lu, C. Barnes, S. DiVerdi, and A. Finkelstein, "RealBrush: Painting with examples of physical media," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 117:1–117:12, Jul. 2013.

[9] Y.-Y. Chuang, D. B. Goldman, K. C. Zheng, B. Curless, D. Salesin, and R. Szeliski, "Animating pictures with stochastic motion textures," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 853–860, Jul. 2005.

[10] M. Okabe, K. Anjyo, and R. Onai, "Creating fluid animation from a single image using video database," *Comput. Graph. Forum*, vol. 30, no. 7, pp. 1973–1982, Sep. 2011.

[11] K. Bhat, S. Seitz, J. Hodgins, and P. Khosla, "Flow-based video synthesis and editing," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 360–363, Aug. 2004.

[12] A. Eden, A. Bargteil, T. Goktekin, S. Eisinger, and J. O'Brien, "A method for cartoon-style rendering of liquid animations," in *Proc. Graph. Interface*, 2007, pp. 51–55.

[13] A. Semmo, J. E. Kyprianidis, M. Trapp, and J. Döllner, "Real-time rendering of water surfaces with cartography-oriented design," in *Proc. Symp. Comput. Aesthetics*, 2013, pp. 5–14.

[14] N. Svakhine, Y. Jang, D. Ebert, and K. Gaither, "Illustration and photography inspired visualization of flows and volumes," in *Proc. IEEE Visualization*, 2005, pp. 687–694.

[15] J. Yu, X. Jiang, H. Chen, and C. Yao, "Real-time cartoon water animation," *Comput. Animat. Virtual Worlds*, vol. 18, no. 4/5, pp. 405–414, Sep. 2007.

[16] S.-H. Zhang, T. Chen, Y.-F. Zhang, S.-M. Hu, and R. Martin, "Video-based running water animation in chinese painting style," *Sci. China Series F: Inf. Sci.*, vol. 52, no. 2, pp. 162–171, Feb. 2009.

[17] S. Xu, Y. Xu, S. Kang, D. Salesin, Y. Pan, and H.-Y. Shum, "Animating chinese paintings through stroke-based decomposition," *ACM Trans. Graph.*, vol. 25, no. 2, pp. 239–267, Apr. 2006.

[18] E. N. Mortensen and W. A. Barrett, "Intelligent scissors for image composition," in *Proc. 22nd Annu. Conf. Comput. Graph. Interactive Techn.*, 1995, pp. 191–198.

[19] D. Shepard, "A two-dimensional interpolation function for irregularly-spaced data," in *Proc. 23rd ACM Nat. Conf.*, 1968, pp. 517–524.

[20] S. Hsu and I. Lee, "Drawing and animation using skeletal strokes," in *Proc. 21st Annu. Conf. Comput. Graph. Interactive Techn.*, 1994, pp. 109–118.

[21] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Comput. Graph. Image Process.*, vol. 1, no. 3, pp. 244–256, 1972.

[22] D. H. Douglas and T. K. Peucker, *Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature*. Hoboken, NJ, USA: Wiley, 2011, pp. 15–28.

[23] X. Tricoche, G. Scheuermann, and H. Hagen, "Tensor topology tracking: A visualization method for time-dependent 2d symmetric tensor fields," *Comput. Graph. Forum*, vol. 20, pp. 461–470, 2001.

[24] Q. Lou and L. Liu, "Curve intersection using hybrid clipping," *Comput. Graph.*, vol. 36, no. 5, pp. 309–320, Aug. 2012.

[25] Y. Zheng and D. Doermann, "Robust point matching for nonrigid shapes by preserving local neighborhood structures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 643–649, Apr. 2006.

[26] R. Fedkiw, J. Stam, and H. W. Jensen, "Visual simulation of smoke," in *Proc. 28th Annu. Conf. Comput. Graph. Interactive Techn.*, 2001, pp. 15–22.

[27] C.-Y. Yao, M.-T. Chi, T.-Y. Lee, and T. Ju, "Region-based line field design using harmonic functions," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 6, pp. 902–913, Jun. 2012.

[28] E. Zhang, J. Hays, and G. Turk, "Interactive tensor field design and visualization on surfaces," *IEEE Trans. Vis. Comput. Graph.*, vol. 13, no. 1, pp. 94–107, Jan. 2007. [Online]. Available: http://dx.doi.org/10.1109/TVCG.2007.16

[29] G. Chen, V. Kwatra, L.-Y. Wei, C. D. Hansen, and E. Zhang, "Design of 2d time-varying vector fields," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 10, pp. 1717–1730, Oct. 2012.

[30] J. L. Walsh, *The Location of Critical Points of Analytic and Harmonic Functions*. New York, NY, USA: American Math. Soc., 1950.

[31] M. S. Floater, "Mean value coordinates," *Comput. Aided Geometric. Des.*, vol. 20, no. 1, pp. 19–27, Mar. 2003.

[32] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 313–318, Aug. 2003.

[33] R. Kirk, *Experimental Design: Procedures for the Behaviors Sciences*, 2nd ed. Emeryville, CA, USA: Brooks/Cole, 1982.

Yu-Chi Lai received the BS degree from Electrical Engineering Department, National Taiwan University, Taipei, Republic of China, in 1996, the MS and PhD degrees in electrical and computer engineering from the University of Wisconsin-Madison, Madison, Wisconsin, in 2003 and 2009, respectively, and the second MS and PhD degrees in computer science, in 2004 and 2010, respectively. He is currently an assistant professor with NTUST and his research interests include the area of graphics, vision, and multimedia. He is a member of the IEEE.

Bo-An Chen received the BS degree from Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, Republic of China, in 2014. He is currently working toward the graduate degree in the Department of Computer Science and Information Engineering, NTUST and his research interests include the area of graphics, vision, and multimedia.

Kuo-Wei Chen received the BS degree from the Department of Electronic and Computer Engineering, National Taiwan University of Science and Technology, Taipei, Republic of China, in 2013. He is currently working toward the graduate degree in Department of Computer Science and Information Engineering, NTUST and his research interests include the area of graphics, vision, and multimedia.

Wei-Lin Si received the BS degree from the Department of Computer Science and Information Engineering, Wen-Hua University, Taipei, Republic of China, in 2013. He is currently working toward the graduate degree in the Department of Computer Science and Information Engineering, NTUST and his research interests include the area of graphics, vision, and multimedia.

Chih-Yuan Yao received the MS and PhD degrees in computer science and information engineering from National Cheng-Kung University, Taiwan, in 2003 and 2010, respectively. He is an assistant professor in the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology (NTUST), Taipei, Taiwan. His research interest is computer graphics, including mesh processing and modeling, and non-photorealistic rendering (NPR). He is a member of the IEEE.

Eugene Zhang received the PhD degree in computer science from the Georgia Institute of Technology, in 2004. He is currently an associate professor with Oregon State University, where he is a member in the School of Electrical Engineering and Computer Science. During part of 2011 and 2012, he was a guest professor with the Free University of Berlin and the Max-Planck-Institute in informatics. His research interests include computer graphics, scientific visualization, geometric modeling, and computational topology. He received the National Science Foundation CAREER award in 2006. He has also served as a program co-chair for CAD/CG 2013. He is a senior member of the IEEE and the ACM.