Stroke-Based Stylization Learning and Rendering with Inverse Reinforcement Learning

Ning Xie[*], **Tingting Zhao**[†], **Feng Tian**[‡], **Xiaohua Zhang**[§], **and Masashi Sugiyama**[¶]
[*]Tongji University, China. [†]Tianjin University of Science and Technology, China.
[‡]Bournemouth University, UK. [§] Hiroshima Institute of Technology, Japan.
[¶]The University of Tokyo, Japan.

Abstract

Among various traditional art forms, *brush stroke drawing* is one of the widely used styles in modern computer graphic tools such as GIMP, Photoshop and Painter. In this paper, we develop an *AI-aided art authoring* ($A4$) system of *non-photorealistic rendering* that allows users to automatically generate brush stroke paintings in a specific artist's style. Within the reinforcement learning framework of brush stroke generation proposed by Xie et al. [Xie *et al.*, 2012], our contribution in this paper is to learn artists' drawing styles from video-captured stroke data by *inverse reinforcement learning*. Through experiments, we demonstrate that our system can successfully learn artists' styles and render pictures with consistent and smooth brush strokes.

1 Introduction

Artistic stylization in non-photorealistic rendering enables users to stylize pictures with the appearance of traditional art forms, such as pointillism painting, line sketching, or brush stroke drawing. Among them, brush stroke drawing is one of the widely used art styles across different cultures in history. In computer-generated painterly rendering, *stroke placement* is a big challenge and significant efforts have been made to investigate how to draw a stroke with realistic brush texture in a desired shape and how to organize multiple strokes [Fu *et al.*, 2011].

The goal of this paper is to develop an AI-aided art authoring system for artistic brush stroke generation. In this section, we first review backgrounds in computer graphics and artificial intelligence, and then give an overview of our proposed system.

1.1 Background in Computer Graphics

The most straightforward approach for painterly rendering would be *physics-based painting*, i.e., giving users an intuitive feeling just like drawing with a real brush. Some works modeled physical virtual brushes including its 3D structure, dynamics, interaction with paper surface [Chu and Tai, 2004] and simulating the physical effect of the ink dispersion [Chu and Tai, 2005]. These virtual brushes can be used to draw various styles of strokes with a digital pen or mouse. However, it is very complex to control a virtual brush. Furthermore, since the computational cost is often very high to achieve satisfactory visual effects to human eyes, some physics-based painting approaches rely on graphics processing units (GPUs) for obtaining reasonable performance [Chu *et al.*, 2010].

To address these issues associated with physics-based painting, the *stroke-based rendering* approach was proposed to directly simulate rendering marks (such as lines, brush strokes, or even larger primitives such as tiles) on a 2D canvas. This stroke-based rendering [Hertzmann, 2003] underpins many artistic rendering algorithms, especially on those emulating traditional brush-based artistic styles such as oil painting and watercolor.

Although physics-based painting and stroke-based rendering are useful for (semi-)professional usage, often users who have no painting expertise are only interested in final results rather than the painting process itself [Kalogerakis, 2012]. To make the painterly rendering system more accessible to novice users, several researchers investigated *beautification*. The early work [Theodosios and Van, 1985] explored automatic techniques to beautifying geometric drawings by enforcing various relations such as the collinearity of lines and the similarity of their lengths. The approach by Igarashi et al. [Igarashi *et al.*, 1997] offered users several choices in the beautifying process.

Filter-based methods are also widely used for building artistic rendering algorithms applied in the image manipulation software such as Photoshop and Gimp. The main task is to find out the nice or beautiful outlines/edges based on novel filters, such as bilateral filter [Pham and Vliet, 2005], DoG filter [Sýkora *et al.*, 2005], morphological filter [Bousseau *et al.*, 2007], shock filter [Kang and Lee, 2008] and Kuwahara filters [Kyprianidis *et al.*, 2009]. These techniques are usually based on heuristics developed through hands-on experience, showing that certain combinations of filters produce an artistic look, more precisely, called stylized cartoon rendering, pen-and-ink illustration, or watercolor painting. However, the connection between the edge-preserving image simplification

[*]Email: ningxie@tongji.edu.cn
[†]Email: tingting@tust.edu.cn
[‡]Email: ftian@bournemouth.ac.uk
[§]Email: zhxh@cc.it-hiroshima.ac.jp
[¶]Email: sugi@k.u-tokyo.ac.jp

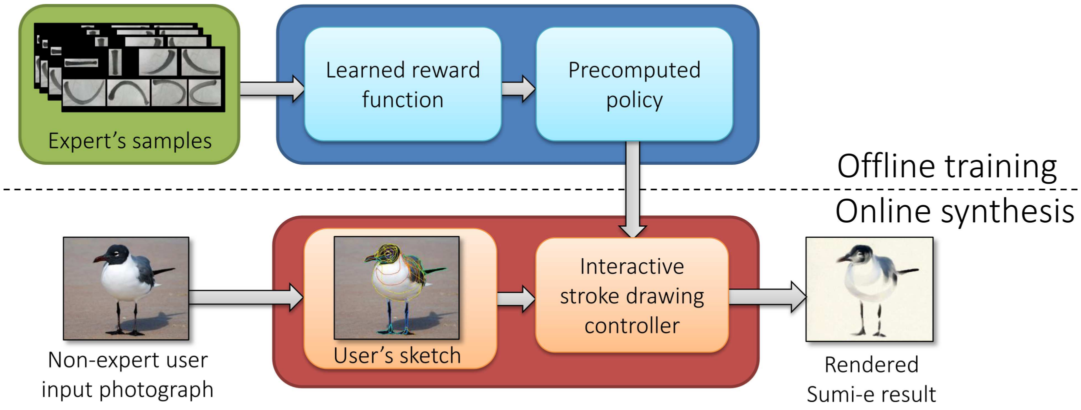


Figure 1: Overview of our AI-aided art authoring ($A4$) system.

and the artistic rendering is less obvious, because the significant artistic look is often achieved or further reinforced by taking the local image structure and brush stroke details into account, rather than the global image abstraction. In practice, the designers usually firstly apply the painting style filter on the real photo in order to imagine the whole art authoring in terms of the entire layout. Then, another layer is created on the top to emphasize the local parts of the image that are important to the users by hand or stroke-based methods.

More recently, methods that attempt not only to beautify generated artistic images, but also to maintain users' personal styles are pursued. Studies of style imitation in artistic rendering focused on ink sketching. Baran et al. [Baran *et al.*, 2010] proposed a method to draw smooth curves while maintaining the details. The sketch beautification approach by Orbay et al. [Orbay and Kara, 2011] used the model that automatically learns how to parse a drawing. Zitnick [Zitnick, 2013] proposed a general purpose approach to handwriting beautification using online input from a stylus. Since techniques of line sketching style imitation are not suitable to synthesizing brush strokes, quite a few previous works [Xu *et al.*, 2006; Zeng *et al.*, 2010; Lu *et al.*, 2012] tried to reproduce brush stroke texture as reference samples.

1.2 Background in Artificial Intelligence

Differently from the above approaches developed in computer graphics, the system we propose in this paper trains a virtual brush agent to learn the stroke drawing model according to particular artists' styles using their stroke drawing videos. The problem on truncated fault texture can be solved by using our learned stroke drawing behavior model. Since the brush agent is trained locally with the data set of basic stroke shapes, we can create strokes in new shapes even when they are quite different from an artist's examples. This is eminently suitable for the artistic stylization of images when non-expert users try to render their favorite photos into a particular artist's style with just a few button clicks.

Our proposed system is based on the *reinforcement learning* (RL) method to artistic brush stroke generation [Xie *et al.*, 2012], which allows users to automatically produce consistent and smooth brush strokes. In this RL approach, the task of synthesizing the texture of each individual stroke is formulated as a sequential decision making problem based on the Markov decision process, where a soft-tuft brush is regarded as an RL agent. Then, to sweep over the shape closed by the contour, the agent is trained by a *policy gradient method* [Williams, 1992] to learn which direction to move and how to keep the stable posture while sweeping over various stroke shapes provided as training data. Finally, in the test phase, the agent chooses actions to draw strokes by moving a virtual inked brush within a newly given shape represented by a closed contour.

In this paper, we extend this RL-based approach to be able to incorporate personal artistic stylization. More specifically, we propose to use a method of *inverse RL* [Abbeel and Ng, 2004] to learn the reward function from stroke data video-captured from artists: we first invite artists to draw strokes using our handcrafted device for recording the brush movement. The brush footprint in each key frame of the captured stroke-drawing video is then extracted, and time series data are obtained by assembling the extracted posture configuration of each footprint including the motion attitude, pose, and locomotion of the brush. The data are used to mimic the artist's stroke drawing style through the reward function learned by inverse RL (IRL).

1.3 Overview of Our Proposed System

An overview of our system, called *AI-aided art authoring* ($A4$) system for artistic brush stroke generation, is illustrated in Figure 1. Our system consists of two phases: an *online synthesis phase* and an *offline training phase*.

In the *online synthesis phase*, $A4$ provides a fast and easy-to-use graphical-user interface so that users can focus on developing art work concepts just by sketching the position and attitude of desired strokes. Given an input picture or photo, even non-expert users can sketch the shapes of desired strokes using either closed contours or simple curves.

In the *offline training phase*, the main task is to train the virtual agent so as to synthesize strokes in an artist's drawing style. Instead of the classical policy gradient method [Williams, 1992], we use the state-of-the-art policy gradient algorithm called *importance-weighted policy gradients with*

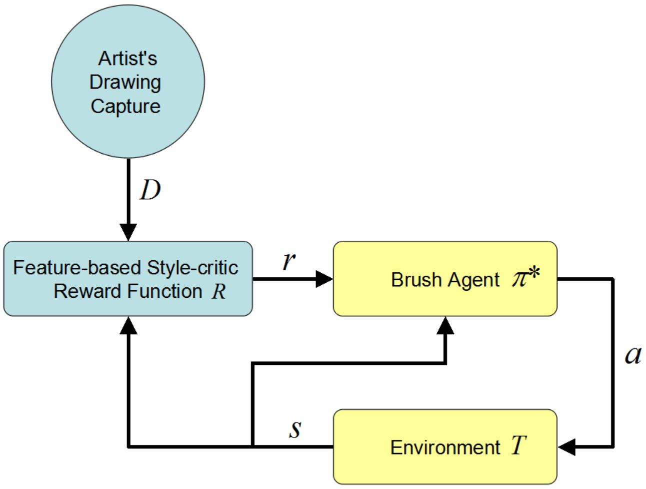


Figure 2: Brush agent with style learning ability. Our system is an extension of the existing approach, marked in yellow, to capture artists' drawing for learning feature-based style-critic reward function.

parameter-based exploration [Zhao *et al.*, 2013], which allows stable policy update and efficiently reuse of previously collected data.

Through experiments, we demonstrate that the proposed system is promising in producing stroke placement with a personalized style.

2 Reinforcement Learning Formulation of Brush Agent

In order to synthesize the painting imagery of an artist's personal style, we construct our brush agent equipped with the style learning ability by extending the existing RL-based approach [Xie *et al.*, 2012] as illustrated in Figure 2.

We assume that our stroke drawing problem is a discrete-time Markov decision process. At each time step t, the agent observes a state $\boldsymbol{s}_t \in \mathcal{S}$, selects an action $a_t \in \mathcal{A}$, and then receives an immediate reward r_t resulting from a state transition. The state space $\mathcal{S}$ and action space $\mathcal{A}$ are both defined as continuous spaces in this paper. The dynamics of the environment is characterized by unknown conditional density $p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t)$, which represents the transition probability density from current state $\boldsymbol{s}_t$ to next state $\boldsymbol{s}_{t+1}$ when action a_t is taken. The initial state of the agent is determined following unknown probability density $p(\boldsymbol{s}_1)$. The immediate reward r_t is given according to the reward function $R(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})$. The agent's decision making procedure at each time step t is characterized by a parameterized policy $p(a_t|\boldsymbol{s}_t, \boldsymbol{\theta})$ with parameter $\boldsymbol{\theta}$, which represents the conditional probability density of taking action a_t in state $\boldsymbol{s}_t$.

A sequence of states and actions forms a *trajectory* denoted by

$$h := [\boldsymbol{s}_1, a_1, \ldots, \boldsymbol{s}_T, a_T],$$

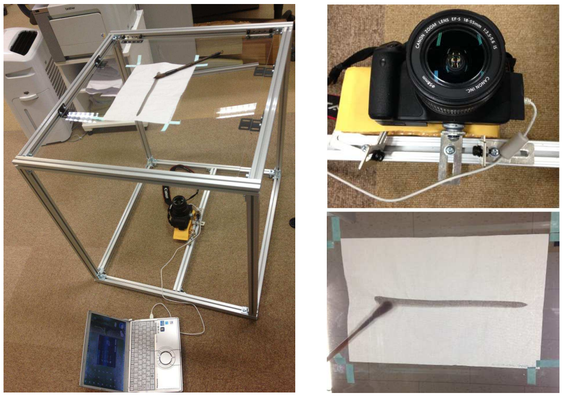


Figure 3: Illustration of our brush dynamic behavior capturing device. The picture on the left side shows the whole profile of the footprint capturing device. The picture on the top right shows the digital single-lens reflex camera. The picture at the bottom right is the glass panel for capturing stroke drawing.

where T denotes the number of steps called the horizon length. Given policy parameter $\boldsymbol{\theta}$, trajectory h follows

$$p(h|\boldsymbol{\theta}) = p(\boldsymbol{s}_1) \prod_{t=1}^{T} p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t) p(a_t|\boldsymbol{s}_t, \boldsymbol{\theta}).$$

The discounted cumulative reward along h, called the *return*, is given by

$$\mathcal{R}(h) := \sum_{t=1}^{T} \gamma^{t-1} R(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}),$$

where $\gamma \in [0, 1)$ is the *discount factor* for future rewards.

The goal is to optimize the policy parameter $\boldsymbol{\theta}$ so that the *expected return* is maximized:

$$\boldsymbol{\theta}^* := \arg\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta}), \tag{1}$$

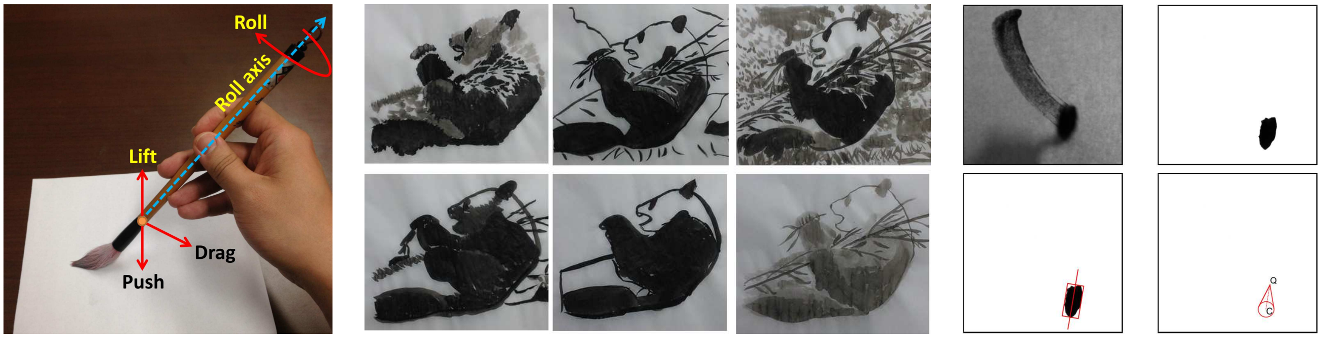
where $J(\boldsymbol{\theta})$ is the expected return for policy parameter $\boldsymbol{\theta}$:

$$J(\boldsymbol{\theta}) := \int p(h|\boldsymbol{\theta}) \mathcal{R}(h) \mathrm{d}h.$$

In the previous work [Xie *et al.*, 2012], the reward function was manually designed to produce "nice" drawings. On the other hand, we aim to *learn* the reward function from an artist's drawing data $\mathcal{D}$ in this paper. We assume that data $\mathcal{D} = \{\tau_1, \ldots, \tau_N\}$ is generated following optimal policy π^*, where the n-th trajectory τ_n is a T-step sequence of state-action pairs $\tau_i = \{(s_{i,1}, a_{i,1}), \ldots, (s_{i,T}, a_{i,T})\}$. In Section 3, we describe our device to capture an artist's drawing to obtain $\mathcal{D}$ and explain an inverse RL method [Abbeel and Ng, 2004]. Then in Section 4, a policy learning method [Zhao *et al.*, 2013] is introduced.

3 Reward Function Learning from Artist

To learn a particular artist's stroke drawing style, we collect stroke data from brush motion and drawings on the canvas and then learn the reward function from the collected data. In this section, we first describe the details of the data collection procedure and then introduce our reward function.



(a) (b) (c)

Figure 4: Data collection. (a) A stroke is generated by moving the brush with three actions: Action 1 is regulating the direction of the brush movement, Action 2 is pushing down/lifting up the brush, and Action 3 is rotating the brush handle. (b) Real data collected from six artists. Each picture corresponds to each artist, where difference in their drawing styles can be observed. (c) Footprint capturing process. Our 2D brush model with tip Q and a circle with center C and radius r are illustrated.

3.1 Data Collection

We designed a device shown in Figure 3 to video-record brush motion. A *digital single-lens reflex* camera is mounted at the bottom of the frame of the device. Data collection is carried out under normal in-door lighting and thus there is no need for automatic camera calibration in real-time. A traditional Asian calligraphy paper is placed on the transparent glass panel on the top of the device. In each data-collection session, an artist is asked to draw a panda with various strokes on the glass panel. The brush motion of an artist is captured when they dip the brush into the traditional calligraphy ink and start drawing strokes.

We split the recorded video of the stroke drawing into frames to analyze brush movement (Figure 4 (c)). To each frame, we apply the model-based tracking technique [Davies, 2005] and detect the posture configuration of brush footprints such as the brush movement information (the velocity, heading direction and pose) and the relative location information to the target desired shape over time. We then apply *principal component analysis* [Jolliffe, 2002] to compute the principal axis of the footprint which defines the direction of the footprint. Finally, the configuration of the footprint is determined by matching the template of the footprint which consists of a tip Q and a circle with center C and radius r.

3.2 Reward Function Design

We design the reward function to measure the quality of the brush agent's stroke drawing movement. First of all, a s-moother movement should yield a higher immediate reward value. We calculate the immediate reward value by considering (i) the distance between the center of the brush agent and the nearest point on the medial axis of the shape at the current time step, and (ii) the change of the local configuration of the brush agent after an action:

$$R(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) = \begin{cases} 0 & \text{if } f_t = f_{t+1} \text{ or } l = 0 \\ 1/C(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) & \text{otherwise,} \end{cases}$$

where f_t and f_{t+1} are footprints at time steps t and $t+1$, respectively. This reward design means that the immediate reward is zero when the brush is blocked by a boundary as $f_t = f_{t+1}$ or the brush is going backward to a region that has already been covered by previous footprints f_i for $i < t+1$. $C(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})$ calculates the cost of the transition of footprints from time t to $t+1$ as

$$\begin{aligned} C(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) =& \alpha_1|\omega_{t+1}| + \alpha_2|d_{t+1}| \\ &+ \alpha_3\Delta\omega_{t,t+1} + \alpha_4\Delta\phi_{t,t+1} + \alpha_5\Delta d_{t,t+1}, \end{aligned}$$

where the first two terms measure the cost regarding the location of the agent, while the last three terms measure the cost regarding the posture when the agent moves from time t to $t+1$. More specifically, $\Delta\omega_{t,t+1}, \Delta\phi_{t,t+1}$, and $\Delta d_{t,t+1}$ are normalized changes in angle ω of the velocity vector, heading directions ϕ, and ratios d of the offset distance between time t and time $t+1$:

$$\Delta\omega_{t+1} = \begin{cases} 1 & \text{if } \omega_t = \omega_{t+1} = 0, \\ \frac{(\omega_t - \omega_{t+1})^2}{(|\omega_t| + |\omega_{t+1}|)^2} & \text{otherwise.} \end{cases}$$

$\Delta\phi_{t,t+1}$ and $\Delta d_{t,t+1}$ are defined in the same way. To set the values of five parameters $\alpha_1, \alpha_2, \ldots, \alpha_5$, we use the *maximum-margin inverse reinforcement learning* method [Abbeel and Ng, 2004]. This allows us to learn the artist's personal style based on an his/her drawing data through inferring appropriate values for $\alpha_1, \alpha_2, \ldots, \alpha_5$.

4 Policy Learning

The previous work [Xie *et al.*, 2012] learned policies by the classical *policy gradient method* [Williams, 1992]. However, this algorithm is often unreliable due to the large variance of the policy gradient estimator [Zhao *et al.*, 2011].

To mitigate the large variance problem, an alternative method called *policy gradients with parameter based exploration* (PGPE) was proposed [Sehnke *et al.*, 2010]. The basic idea of PGPE is to use a deterministic policy and introduce stochasticity by drawing parameters from a prior distribution. More specifically, parameters are sampled from the prior distribution at the start of each trajectory, and thereafter the controller is deterministic. Thanks to this per-trajectory formulation, the variance of gradient estimates in PGPE does not

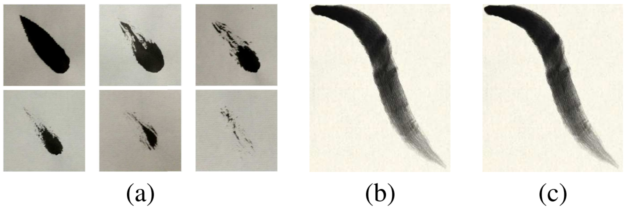


Figure 5: (a) Footprints extracted from the video in different water dispersion conditions. From the top left to the lower right corners, the ink content of hollow strokes is decreasing continuously. (b) A dry rendering result without water dispersion. (c) A rendering result with water dispersion using more ink.

increase with respect to trajectory length [Zhao *et al.*, 2011]. The gradient estimation of PGPE can be further stabilized by subtracting a *baseline* [Zhao *et al.*, 2011].

However, (baseline-subtracted) PGPE still requires a relatively large number of samples to obtain accurate gradient estimates, which can be a critical bottleneck for our application due to the large costs and time in data collection. To cope with this problem, we use a variant called *importance-weighted policy gradients with parameter-based exploration* (IW-PGPE) [Zhao *et al.*, 2013], which allows efficient reuse of previously collected data. In the online synthesis phase illustrated in Figure 1, the user is allowed to choose one or several learned policies to control the drawing behavior for each input shape.

5 Stroke Texture Rendering

We use both the raster brush texture mapping and the physical pigment dispersion simulation to generate both dry and wet textures. Rendering is carried out by capturing single footprints of a brush and then stamping them along the trajectory obtained by the brush agent's learned policy. The scanned footprint images are used as the reference texture of brush footprints and sampled with different contents of ink of hollow strokes for rendering the change of the stroke texture. Then, we save them into raster textures to create our brush footprint texture libraries as shown in Figure 5(a). For the *drying stroke rendering*, given the parameters of the brush ink style, footprint texture images with different ink contents are affinely transformed and then mapped onto the optimal sequential collection of footprints according to the shapes and orientations of the footprints.

Discrete series of footprint images need to be interpolated to render strokes with smooth textures. To do so, each intermediate pixel on the resulting stroke texture is linked by a pair of points on the two nearest footprints using the interval piecewise Bézier splines. Figure 5 (b) illustrates a dry stroke. *Wet stroke rendering* is carried out by adding ink and pigment dispersion into the brush texture mapping. We adopt the water-based paint flow physical simulation [Chu and Tai, 2005]. The quantity of ink and pigment on the paper canvas is initialized according to the current sampled brush texture images. Figure 5 (c) illustrates a wet stroke where its shape and trajectory is the same as those in Figure 5 (b).

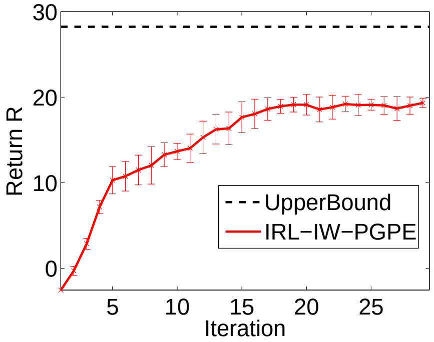


Figure 6: Policy iteration. The error bars denote the standard deviation over 16 runs.

6 Experiments and Results

Figure 6 plots the average return over 16 trials as the function of policy update iterations, obtained by the policies learned by our approach. Returns at each trial are computed over 300 training episode samples. This graph shows that the average return sharply increases in an early stage and then converges at about the 20th iteration.

Stroke drawing results by an artist, the agent trained with the learned reward function, and the agent trained with the manually designed reward function [Xie *et al.*, 2012] are compared in Figure 7. The results show that the proposed method imitates the real artist's stroke drawing better than the previous method. More specifically, the two results marked with red in the right-most column show that our rendered stroke texture is much smoother than the one obtained with the manually designed reward function.

Finally, we applied the policy obtained by our method to photo artistic conversion system [Xie *et al.*, 2011] (Figure 8), where we manually sketched contours from the original pictures that represent the boundaries of desired strokes. The results in Figure 8 (c) show that shapes are filled with smooth strokes by our IRL method and visually reasonable drawings are obtained.

To further investigate our IRL-based method, we performed the user study on the aesthetic assessment of the traditional oriental ink painting simulation between the proposed $A4$ system and the brush stroke (Sumie) filter of the state-of-the-art commercial software (Adobe Photoshop CC 2014). We invited 318 individuals to take the online questionnaire survey. We conducted a quantitative user study following the same approach as in [Xu *et al.*, 2008]. We asked the participants to tell which one is more like the oriental ink painting style for each pairs (shown as (b) and (c) in Figure 8) among four pairs of paintings. We include this question in the user study to directly compare subjective aesthetic assessment of the viewer by selecting which images they like. The aesthetic scores are given by participants shown in Figure 9. Obviously, our results obtained higher aesthetic scores than Photoshop.

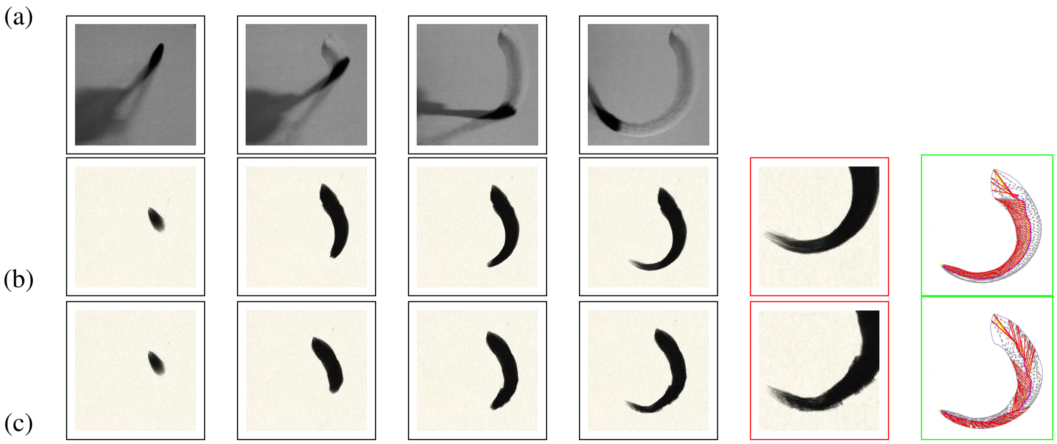
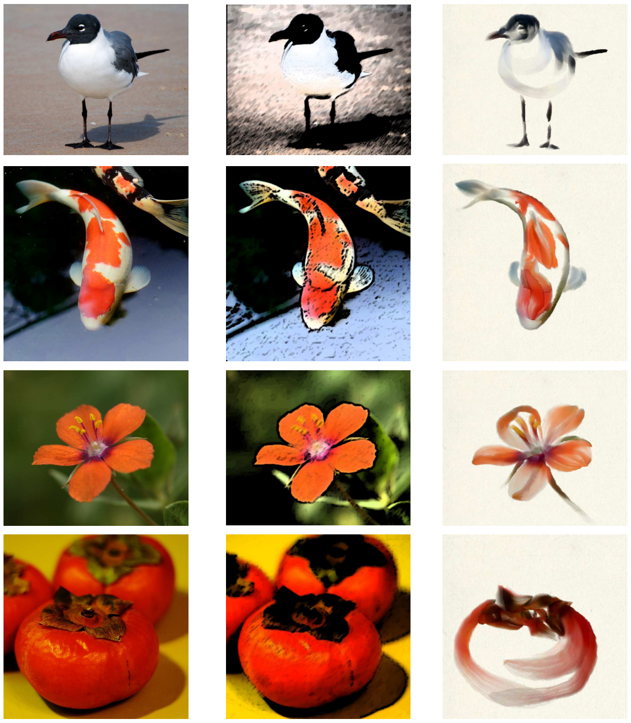


Figure 7: Comparison of stroke-drawing processes. (a) Artist's real data. (b) Trained with the learned reward function by our proposed method. (c) Trained with the manually designed reward function in the previous work. Green boxes show brush trajectories, while red boxes show rendered details.



(a) Original image (b) Photoshop (c) IRL

Figure 8: Results of photo conversion into the brush stroke drawings. (a) Original images. (b) Sumie filter in Photoshop. (c) Our proposed IRL.

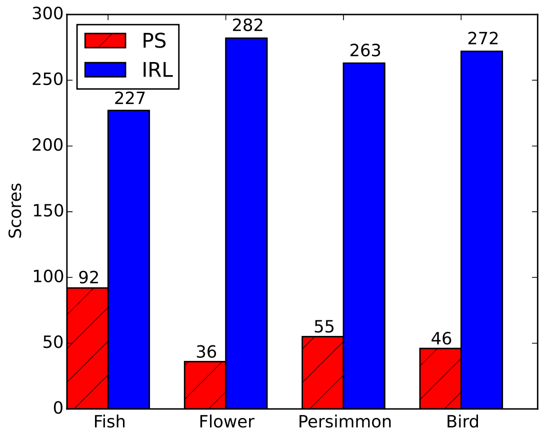


Figure 9: User study of the aesthetic assessment over 318 candidates. PS means the Sumie filter in Photoshop. IRL is our proposed method.

7 Conclusion and Future Work

We have proposed an AI-aided art authoring system ($A4$) so as to fast and easy creation of stylized stroke-based paintings. Our main contributions in this papers are (i) we developed a device to capture artists' brush strokes, (ii) we collected training data in various styles, (iii) we applied inverse reinforcement learning to learn the reward function from the data provided by artists, (iv) we applied the state-of-the-art reinforcement learning method, IW-PGPE (importance-weighted policy gradients with parameter-based exploration), to accurately learning the policy function by efficiently reusing previously collected data, and (v) we demonstrated through experiments the effectiveness of our proposed approach in converting photographs into stroke drawings.

In the future, an automatic contour extraction from pictures may be explored to simplify the process of photo stylization for non-expert users to both learn and detect local contour-based representations for mid-level feature information in the form of hand drawn contours in images.

Acknowledgments

We would like to thank Laga Hamid for his suggestions and advices at different stages of this research. We also thanks the reviewers for their helpful comments. Ning Xie was supported by National Science Foundation of China (No.61272276) and Tongji University Young Scholar Plan (No.2014KJ074), Tingting Zhao was supported by SRF for ROCS,SEM., and Masashi Sugiyama was supported by KAKENHI 23120004.

References

[Abbeel and Ng, 2004] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

[Baran *et al.*, 2010] I. Baran, J. Lehtinen, and J. Popovic. Sketching clothoid splines using shortest paths. *Comput. Graph. Forum*, 29(2):655–664, 2010.

[Bousseau *et al.*, 2007] A. Bousseau, F. Neyret, J. Thollot, and D. Salesin. Video watercolorization using bidirectional texture advection. *ACM Trans. Graph.*, 26(3):104, 2007.

[Chu and Tai, 2004] N. Chu and C. L. Tai. Real-time painting with an expressive virtual Chinese brush. *IEEE Computer Graphics and Applications*, 24(5):76–85, 2004.

[Chu and Tai, 2005] N. Chu and C. L. Tai. Moxi: real-time ink dispersion in absorbent paper. *ACM Trans. Graph.*, 24(3):504–511, 2005.

[Chu *et al.*, 2010] N. Chu, B. Baxter, L. Y. Wei, and N. K. Govindaraju. Detail-preserving paint modeling for 3D brushes. In *Proceedings of the 8th International Symposium on NPAR*, pages 27–34, New York, NY, USA, 2010. ACM.

[Davies, 2005] E.R. Davies. *Machine Vision: Theory, Algorithms, Practicalities, Third Edition*. Elsevier, 2005.

[Fu *et al.*, 2011] H. Fu, S. Zhou, L. Liu, and N. Mitra. Animated construction of line drawings. *ACM Trans. Graph. (Proceedings of ACM SIGGRAPH ASIA)*, 30(6), 2011.

[Hertzmann, 2003] A. Hertzmann. A survey of stroke-based rendering. *IEEE Computer Graphics and Applications*, 23:70–81, 2003.

[Igarashi *et al.*, 1997] T. Igarashi, S. Matsuoka, S. Kawachiya, and H. Tanaka. Interactive beautification: A technique for rapid geometric design. In *ACM Symposium on User Interface Software and Technology*, pages 105–114, 1997.

[Jolliffe, 2002] I. T. Jolliffe. *Principal Component Analysis*. Springer Verlag, New York, 2002.

[Kalogerakis, 2012] D.and Breslav S.and Hertzmann A. Kalogerakis, E.and Nowrouzezahrai. Learning hatching for pen-and-ink illustration of surfaces. *ACM Trans. Graph.*, 31(1):1:1–1:17, February 2012.

[Kang and Lee, 2008] H. Kang and S. Lee. Shape-simplifying image abstraction. *Computer Graphics Forum*, 27(7):1773–1780, 2008.

[Kyprianidis *et al.*, 2009] J. E. Kyprianidis, H. Kang, and Jürgen Döllner. Image and video abstraction by anisotropic kuwahara filtering. *Comput. Graph. Forum*, 28(7):1955–1963, 2009.

[Lu *et al.*, 2012] J. Lu, F. Yu, A. Finkelstein, and S. DiVerdi. Helpinghand: example-based stroke stylization. *ACM Trans. Graph.*, 31(4):46:1–46:10, July 2012.

[Orbay and Kara, 2011] G. Orbay and L. B. Kara. Beautification of design sketches using trainable stroke clustering and curve fitting. *IEEE Trans. Vis. Comput. Graph.*, 17(5):694–708, 2011.

[Pham and Vliet, 2005] T. Q. Pham and L. J. V. Vliet. Separable bilateral filtering for fast video preprocessing. In *International Conference on Multimedia Computing and Systems/International Conference on Multimedia and Expo*, pages 454–457, 2005.

[Sehnke *et al.*, 2010] F. Sehnke, A. Graves, C. Osendorfer, and J. Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010.

[Sýkora *et al.*, 2005] D. Sýkora, J. Buriánek, and J. ára. Colorization of black-and-white cartoons. *Image Vision Comput.*, 23(9):767–782, September 2005.

[Theodosios and Van, 1985] P. Theodosios and W. C. J. Van. An automatic beautifier for drawings and illustrations. In *SIGGRAPH*, pages 225–234. ACM, 1985.

[Williams, 1992] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

[Xie *et al.*, 2011] N. Xie, H. Laga, S. Saito, and M. Nakajima. Contour-driven Sumi-e rendering of real photos. *Computers & Graphics*, 35(1):122–134, 2011.

[Xie *et al.*, 2012] N. Xie, H. Hachiya, and M. Sugiyama. Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting. In *ICML*, 2012.

[Xu *et al.*, 2006] S. Xu, Y. Xu, S. B. Kang, D. H. Salesin, Y. Pan, and H. Shum. Animating Chinese paintings through stroke-based decomposition. *ACM Transactions on Graphics*, 25(2):239–267, 2006.

[Xu *et al.*, 2008] S. Xu, H. Jiang, T. Jin, F. C. M. Lau, and Y. Pan. Automatic facsimile of chinese calligraphic writings. *Comput. Graph. Forum*, 27(7):1879–1886, 2008.

[Zeng *et al.*, 2010] K. Zeng, M. T. Zhao, C.M. Xiong, and S.C Zhu. From image parsing to painterly rendering. *ACM Transactions on Graphics*, 29(1):1–11, 2010.

[Zhao *et al.*, 2011] T. Zhao, H. Hachiya, G. Niu, and M. Sugiyama. Analysis and improvement of policy gradient estimation. In *Advances in Neural Information Processing Systems 24*, pages 262–270, 2011.

[Zhao *et al.*, 2013] T. Zhao, H. Hachiya, V. Tangkaratt, J. Morimoto, and M. Sugiyama. Efficient sample reuse in policy gradients with parameter-based exploration. *Neural Computation*, 25(6):1512–1547, 2013.

[Zitnick, 2013] C. L. Zitnick. Handwriting beautification using token means. *ACM Trans. Graph.*, 32(4):53:1–53:8, July 2013.