Parallel Style-aware Image Cloning for Artworks

Yandan Zhao, Xiaogang Jin, *Member*, *IEEE*, Yingqing Xu, Hanli Zhao, Meng Ai, Kun Zhou

Abstract—We present *style-aware image cloning*, a novel image editing approach for artworks, which allows users to seamlessly insert any photorealistic or artificial objects into an artwork to create a new image that shares the same artistic style with the original artwork. To this end, a real-time image transfer algorithm is developed to stylize the cloned object according to a distance metric based on the artistic styles and semantic information. Several interactive functions, such as layering, shadowing, semantic labeling, and direction field editing, are provided to enhance the harmonization of the composite image. Extensive experimental results demonstrate the effectiveness of our method.

Index Terms—non-photorealistic rendering, image editing, seamless cloning, image stylization, style transfer, GPU

1 Introduction

Image cloning is very important and useful in image editing applications. It seeks to clone an object from a source image into a target background to create a seamless composite. Most of recent works (e.g., Poisson image editing [1] and seamless image cloning [2]) have focused on solving the color discrepancies between the source object and target background along the cloning region boundary. While these algorithms can generate good results for real world images, they may create undesirable results for artworks, especially if the source and target regions are of inconsistent visual styles (see Fig. 1(d)). Some methods can be used to correct the texture inconsistency between the source and target using multi-scale techniques [3] and patch-based synthesis [4]. However, when inserting an object into an artwork, these methods may still suffer from artifacts (Fig. 1(e)) because objects and textures in the source and target are too different.

In order to solve the above problem, texture transfer techniques [5], [6], [7] can be used to stylize the photo to match the artistic style of the artwork. However, discoloration artifacts may exist by using these methods because the semantic information of the cloned object is not explored. Moreover, when editing an artwork, system responsiveness is essential. We notice that the above-mentioned methods use a scan-line order to perform texture transfer, and are not suitable for a fast parallel implementation.

In this paper, we present a style-aware image cloning approach for artworks editing (see Figs. 1(b) and (c)). Since our cloning algorithm employs rich image semantics, it is able to simulate the styles of artists better.

(a) Objects/Artwork (b) Result

(c) Our method (d) Poisson editing (e) Image melding

Fig. 1. Given some objects (a girl and a wolf) and a pastel landscape drawing, our method creates a scene of Little Red Riding Hood by seamlessly integrating the objects into the given artwork. Top (left to right): (a) inputs; (b) our result. Bottom: (c) close-up of our result; (d) Poisson editing [Pérez et al. 2003]; (e) the image harmonization application of image melding [Darabi et al. 2012].

The semantic information is obtained by some easy-to-use interactive editing tools such as RepFinder [8], layering, shadowing, semantic labeling, and direction field editing. We formulate the artistic style transfer with semantic information as an energy minimization problem and solve it on the GPU for real-time editing. We also perform a user study to compare our presented

- *Yandan Zhao, Xiaogang Jin, Meng Ai and Kun Zhou are with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, China. E-mail: zhaoyandan@cad.zju.edu.cn, jin@cad.zju.edu.cn, aimeng90@gmail.com, kunzhou@cad.zju.edu.cn.*
- *Yingqing Xu is with Tsinghua University, Beijing, China. E-mail: yqxu@tsinghua.edu.cn.*
- *Hanli Zhao is with Wenzhou University, Wenzhou, China. E-mail: hanlizhao@gmail.com.*

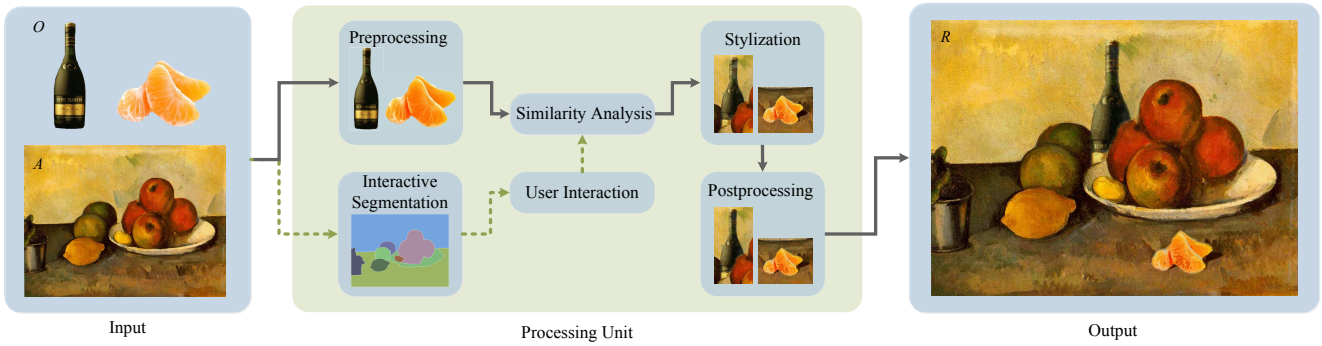


Fig. 2. Schematic view of our style-aware image cloning framework. The green lines denote optional operations.

approach with prior works, verifying the usefulness and effectiveness of our synthesis framework.

The main contributions of this paper are twofold. First, we introduce a novel image editing framework for seamlessly cloning an object from a source image into a target artwork so that the composite looks harmonious with consistent color, luminance, and internal texture. Second, we develop a new distance metric function which takes into consideration the luminance, texture, direction, local coherence, and semantic information of images, and solve it via a new parallel algorithm on the GPU to achieve real-time performance.

2 Related Work

Object cloning techniques have been extensively studied in order to reduce the difference in illumination and color between the cloned objects and the target image. A successful method is proposed by Pérez et al. [1], who introduce a variety of tools for seamless editing of image regions based on solving Poisson equations with Dirichlet boundary conditions. McCann and Pollard [9] develop a GPU-based image editing approach in the gradient domain by extending Poisson equations. They introduce an edge brush coupled with special blending modes to allow users with real-time local manipulations. Farbman et al. [2] propose a mean-value coordinate-based approach for seamless cloning of a source image patch into a target image. Rather than solving a large Poisson linear system, the value of the interpolation is given by a weighted combination of values along the boundary. The use of mean-value coordinates enables real-time cloning of large regions and interactive cloning of video streams. These methods effectively guarantee seamless boundaries between the cloned objects and the target image. However, when the styles between the interiors and exteriors of an object are quite different, unnatural compositions may arise. Xue et al. [10] compute the statistical measures of images and improve the realism of the composites by automatically adjusting these measures. These approaches are all designed for photorealistic images. Recently, Sunkavalli et al. [3] present an image harmonization framework for the blending between objects and target images. A multi-scale pyramid decomposition technique is employed to match contrast, texture, noise, and blur of the images for the production of harmonious composites. Unfortunately, this method still fails to harmonize complex texture styles which are common in non-photorealistic artworks. Image melding [4] can be used for image cloning and image harmonization by synthesizing a transition region. However, it suffers from the same limitation as the method of [3] when the source and target textures are too disparate.

Style transfer approaches try to transfer visual appearances between images based on feature-guided neighborhood matching techniques. Color transfer is a general form of color correction that transfers one image's color characteristics to another [11], [12], [13], [14].

Hertzmann et al. [5] explore the style transfer for the textural aspects of non-photorealistic media and propose a texture transfer framework called Image Analogies. With the help of a pair of training images, the approach automatically creates an analogous transferred result for an input image. Bénard et al. [15] extend the Image Analogies algorithm to create temporally coherent animation. Given an input photo and an artistic example image, image quilting [16] can synthesize an artistic result by stitching together small patches from the input artistic example. With a training set of images, Drori et al. [17] generate a new style by first adaptively partitioning images into fragments in the training set and then stitching together novel fragments. Ashikhmin [6] presents a fast texture transfer approach by extending the search space of the coherent synthesis. Lee et al. [7] propose a directional texture transfer algorithm to express the directional effect based on the feature flows of images. The output image obtained from the algorithm expresses not only the style feature of the example image but also the feature flow of the input image. However, these approaches either fail to provide real-time visual feedback or do not take semantic information into consideration.

Some other techniques are also related to the work of this paper. Fischer et al. [18] apply filtering-based stylization techniques to generate augmented reality images

(a) Artwork (b) Photo (c) (d) (e) Random (f) 1 pass (g) 2 passes

Fig. 3. Illustration of propagating passes. (c) and (d) show the corresponding direction fields of (a) and (b) respectively.

for reducing the visual realism of both the camera image and the virtual graphical objects. The non-photorealistic rendering approach effectively improves immersion in augmented reality. Zhao et al. [19] investigate the stroke placement problem and present a method to parameterize painterly rendering styles. Karsch et al. [20] propose a method to realistically insert synthetic objects into existing photographs with only a small amount of annotation. Their amazing results demonstrate that synthetic images are confusable with real scenes, even for people who believe they are good at telling the difference. The light estimation technique of Lopez-Moreno et al. [21] is able to recover complex lighting configurations in a single image and this method shows convincing image cloning results in photographs. These techniques motivate us to develop a new framework for editing artworks blending between objects by inserting objects from ubiquitous photos.

3 Framework Overview

As shown in Fig. 2, our framework takes an artistic image A and the objects O to be cloned as input, and generates a resultant image R with the user-desired objects cloned.

Our approach begins with a preprocessing step which tries to match the hue and the lightness of the photographic objects to the artwork. We decrease the ambient difference between O and A by histogram matching and reduce the number of O's distinct colors by bilateral filtering and luminance quantization [22]. Because the colors of O and A can be quite different, A may not always contain enough data to match O using RGB channels. Hence, we convert A and O from RGB color space to the YIQ color space and use the Y channel as luminance for the following similarity analysis. Then, we formulate a novel distance metric with style features and semantic information in the similarity analysis step. Some user interactions are optionally needed to get the sematic information when dealing with occlusions, shadow casting, semantic labeling, and direction field editing. Before the interaction, the artwork A is segmented into several parts which represent different layers. Details of the interactive editing tools are presented in Section 5. After that, a stylization step is performed to harmonize the cloned objects with the artistic image. A new parallel stylization technique is developed to achieve real-time feedback. Section 4 describes our artistic style transfer technique in detail. Finally, in the postprocessing step we convert O (with updated Y channel) from YIQ color space to RGB color space. Our framework provides two ways to create the colored final result. One way is to use the original color of O, which is used in most of our examples, such as Fig. 1(b). The other way is to use the color of A according to the map, such as the results in Fig. 5(e) and Fig. 15. To guarantee seamless boundaries between O and A the object mask is employed. The mask is blurred with a Gaussian filter and then used for α-blending.

Algorithm 1 Artistic style transfer

```
Initialize the correspondence M by blocks in random
Compute the energy value E using Formula (1)
repeat
    E' = E
    for each pixel p ∈ O do
        find q' ∈ A with minimal D(p, q')
        Update M(p) = q'
    end for
    Update E using Formula (1)
until (E' − E)/E < τ
return M
```

4 Artistic Style Transfer

4.1 Similarity Energy Minimization

The primary task of the artistic style transfer is to establish the correspondence M that maps each pixel $\mathbf{p}$ in O to a pixel $\mathbf{q}$ in A, *i.e.* $M(\mathbf{p}) = \mathbf{q}$. Based on the map, we can render O using the pixels coming from A. The resultant image R we expect contains not only the basic structure information of O but also the detailed texture information of A.

We define the energy function E as the sum of distances $D(\mathbf{p}, \mathbf{q})$ between each pixel $\mathbf{p}$ and its corresponding pixel $\mathbf{q}$. The energy function E is defined as:

$$E = \sum_{\mathbf{p} \in O} D(\mathbf{p}, M(\mathbf{p})) \tag{1}$$

where the distance D will be described in Subsection 4.2.

When $\mathbf{p}$ and $M(\mathbf{p})$ are similar, their distance $D(\mathbf{p}, M(\mathbf{p}))$ is small, and so does the energy E. In this paper, we take the transfer process as the minimization of the energy function E. Algorithm 1 shows the pseudo code of our framework. To decrease the energy, we

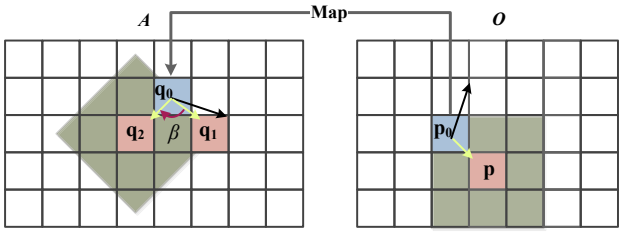


Fig. 4. Illustration of the direction alignment while searching candidates of $M(\mathbf{p})$. Without alignment we search candidates via $\mathbf{q_1}$ according to $M(\mathbf{p_0}) = \mathbf{q_0}$. With alignment we rotate vector $\overrightarrow{\mathbf{q_0}\mathbf{q_1}}$ with β degree in clockwise which is the angle between vectors $\mathbf{p_0}$ and $\mathbf{q_0}$ (black arrows), and then search candidates via $\mathbf{q_2}$ and rotate the neighborhoods of $\mathbf{q_2}$ for neighbor matching (the dark green region).

search more similar pixels $\mathbf{q}'$ to update $M(\mathbf{p})$ repeatedly. The iteration continues until the reduction rate of the energy is smaller than a threshold τ, which is set to 0.001 in all our experiments. The stylized result is obtained by sampling the mapped colors from A. The evolution of each iteration is illustrated in Fig. 3.

4.2 Distance Metric Definition

It is challenging to model artistic style features in artworks. However, style feature plays a critical role in generating artistic results for the cloned objects. In this paper, we propose a compositional feature which contains luminance, direction, texture, local coherence, and semantic information. The luminance, direction and texture features are measured between O and A. The local coherence feature tries to keep the local coherent appearance of R when the pixels of O are replaced by those from A.

Luminance For a corresponding pixel pair, if their neighborhoods are similar, the luminance difference between them is small. Ashikhmin [6] and Lee et al. [7] utilize the sum of the following two parts to define the luminance feature: the difference of neighborhood averages between O and A, and the difference of the standard deviation of luminance of the L-shaped neighborhoods in O and R. Our experiments show that the results, regardless of inclusion of the second part, are similar. For better performance, we define the distance of luminance feature as the difference of mean luminance:

$$L(\mathbf{p}, \mathbf{q}) = \|\overline{N(\mathbf{p})} - \overline{N(\mathbf{q})}\| \tag{2}$$

where $\overline{N(\mathbf{p})}$ and $\overline{N(\mathbf{q})}$ stand for the average of the circular neighborhood of $\mathbf{p}$ and $\mathbf{q}$, respectively.

Direction Most artworks have salient strokes strengthened by artists. These strokes are used to shape objects and characterize artworks. To better emulate an artistic style, the directions of the strokes in the cloned objects should follow the direction field of O instead of A. The directional texture transfer [7] expresses the directional effect by adding a directional factor which relies on the already synthesized neighborhoods in a scan-line order. Different from their approach, the direction feature in our framework is not added into the distance metric D as a directional factor. Instead, the directional effect is achieved by performing direction alignments. We employ the local structure estimation algorithm [23] to calculate the direction fields of O and A. Alternatively, we could employ the non-oriented MLS field algorithm [24] to compute the direction field.

We illustrate the different correspondences with and without alignment in Fig. 4. Let $\mathbf{p_0}$ and $\mathbf{q_0}$ be the known seeds, $\mathbf{p} = \mathbf{p_0} + (1, -1)$ and $\mathbf{q_1} = \mathbf{q_0} + (1, -1)$ be the corresponding neighboring pixels. In the case without alignment, we can search the mapping of $\mathbf{p}$ via $\mathbf{q_1}$ with the same relative coordinate $(1, -1)$ in the neighborhoods of $\mathbf{p_0}$ and $\mathbf{q_0}$, respectively. This is based on the idea of coherence [25], [26]. In our alignment, we perform necessary rotations before the feature matching test. Let β be the angle between the direction vectors of $\mathbf{p_0}$ and $\mathbf{q_0}$. We can find $\mathbf{q_2}$ by rotating $\overrightarrow{\mathbf{q_0}\mathbf{q_1}}$ with β degree in clockwise and we define $\mathbf{q_2} = ROT(\mathbf{q_1}, \beta)$. Therefore, we can search $M(\mathbf{p})$ via $\mathbf{q_2}$. We use a circular neighborhood to perform the luminance calculation since the luminance in such a region is the same no matter it is aligned or not.

Without loss of generality, we define the s-step neighboring pixel of $\mathbf{p}$ as $\mathbf{p_1} = \mathbf{p} + (i, j)$ where (i, j) denotes the relative coordinate from $\mathbf{p}$, s denotes the length of the jump step and $i, j \in \{-s, 0, s\}$. The user can tune the directional influence according to the style of an artwork by using the spherical linear interpolation between $\mathbf{q_1}$ and $\mathbf{q_2}$. The candidate pixel $\mathbf{q}$ for updating the map of $\mathbf{p}$ can now be determined by:

$$\mathbf{q_1} = M(\mathbf{p} - (i, j)) + (i, j) \tag{3}$$

$$\mathbf{q} = \frac{\sin((1-u)\beta)}{\sin(\beta)}\mathbf{q_1} + \frac{\sin(u\beta)}{\sin(\beta)}ROT(\mathbf{q_1}, \beta) \tag{4}$$

where $u \in [0, 1]$ is the interpolation parameter. Smaller values of u are set in examples of pointillism, watercolor, and some types of oil painting without salient directional strokes. Larger values are set in the examples of pencil sketch, pastel drawing and crayon drawing. For the pastel landscape drawing example shown in Fig. 1, we set $u = 1.0$.

Texture The luminance feature fails to handle image blocks with small luminance differences but different textures. The texture feature is measured by the difference in the texture between the neighborhoods of $\mathbf{p}$ and $\mathbf{q}$.

Similar to the method by Qu et al. [27], we use the statistical feature in Gabor wavelet domain [28] to measure the texture feature. We first construct a vector $\mathbf{V}$ using the mean and the standard deviation of the

(a) (b) (c) (d) (e)

Fig. 5. We take an object to be cloned (a) and the artwork in Fig. 1(a) as input, discoloration artifacts may arise if we stylize the cloned object using its own color (b). The stylized result using the colors in the crayon drawing without semantic information cannot solve this problem (c). With semantic information obtained by finding the most similar object (white region) (d), we can generate the result in (e).

magnitude of the transform coefficients with 2 scales and 4 orientations inside a 16×16 window:

$$\mathbf{V} = (\mu_{00}, \sigma_{00}, \mu_{01}, \sigma_{01}, \cdots, \mu_{13}, \sigma_{13}) \tag{5}$$

where μ_{ij} and σ_{ij} ($i \in [0,1], j \in [0,3]$) are the mean and the standard derivation of the transform coefficients. Then the distance of texture feature T is defined as the difference of vector $\mathbf{V}$:

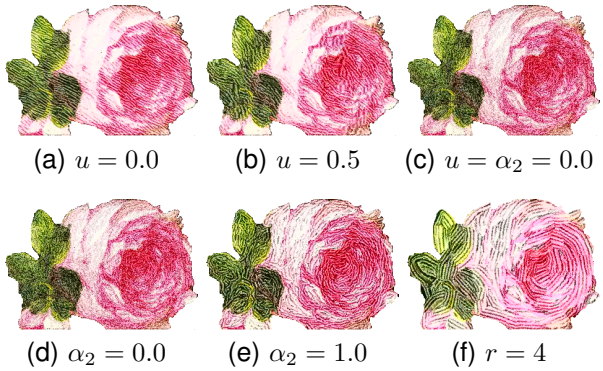
$$T(\mathbf{p}, \mathbf{q}) = \|\mathbf{V}(\mathbf{p}) - \mathbf{V}(\mathbf{q})\| \tag{6}$$

Local coherence If two pixels in A are neighbors, their mapped pixels in R are likely to be neighbors as well. This property is called local coherence [25]. In the coherent synthesis [25] and k-coherence search [26], the coherence is preserved in a serial scheme. That is, the candidates of pixels in R are selected according to the pixels that have been synthesized in the neighborhood. Different from them, because our synthesis scheme is parallel and iterative, the pixels already synthesized are always changing. In addition, the candidates of $\mathbf{p}$ come from the shifted pixels respected to its immediate neighbors in these methods. However, the candidates come from the shifted pixels respected to its s-step neighboring pixels in our parallel scheme. When the step s is large, many pixels are skipped over and the coherence should be further considered.

Therefore, we introduce a coherence item C to constrain our matching as follow:

$$C(\mathbf{p}, \mathbf{q}) = \frac{1}{n} \sum_{\mathbf{p}_{ij} \in N(\mathbf{p})} \min(\|\mathbf{q}_{ij} - M(\mathbf{p}_{ij})\|, r) \tag{7}$$

where N denotes the pixels of the neighborhood block, n is the number of pixels of N, and r is the radius of N. We set $r = 2$ in all experiments. The relationship of local coherence between $\mathbf{p}$ and $\mathbf{p}_{ij}$ in O corresponds to the relationship between $\mathbf{q}$ and $\mathbf{q}_{ij}$ in A, that is, they have the same relative coordinate (i, j). The truncation of distance by r is necessary because $\mathbf{q}_{ij}$ may be quite far from $M(\mathbf{p}_{ij})$.



(a) $u = 0.0$ (b) $u = 0.5$ (c) $u = \alpha_2 = 0.0$

(d) $\alpha_2 = 0.0$ (e) $\alpha_2 = 1.0$ (f) $r = 4$

Fig. 6. Effects of varying parameters. Each caption specifies the modified parameter value from the default ones: $u = 1.0$, $\alpha_2 = 0.1$, $r = 2$.

Semantics To guide synthesis [25], we take the semantic information into consideration and employ it as the weight of the aforementioned features. The semantic feature W describes the semantic similarity between the inserted object region and object regions in the artwork. In previous methods [5], [29], the regions of A and O are labeled and then constitute the label maps. The synthesis is guided by adding comparisons between neighbors from the label maps in the traditional neighborhood matching. We directly use the semantic weight W for each segmented region pair. W of all pairs are initialized with 1.0. Usually, a user may edit an artwork by inserting objects similar to the ones in the scene of the artwork.

Similar objects are expected to have similar appearances in the resulting image. Therefore, we employ the RepFinder algorithm [8] to find similar object regions by using O as the template.

In addition, we find the most similar region by comparing the average color difference between their boundary band matching regions and the object. The most similar region is assigned with 0.001 to increase the matching possibility. Take Fig. 5 for example, although the cloned tree is stylized (see Fig. 5(b)), it is easy to find the discoloration artifact. If we can color the cloned tree

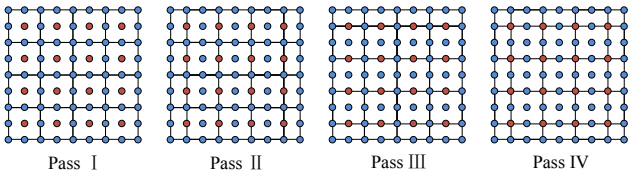


Fig. 7. Illustration of tile shifts. Half of the tile size shifts in horizontal and vertical directions guarantee the local coherence of boundary pixels.

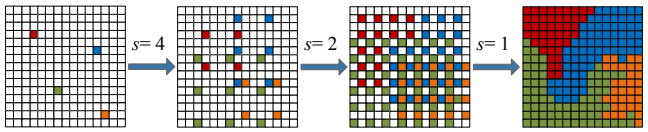


Fig. 8. Parallel stylization scheme. A pass is composed of three sub-passes if the maximal length equals to 4. Each sub-pass propagates from each seed to neighboring pixels in a variational logarithmic step length.

with the color of the trees in the crayon drawing, the cloning result will be more harmonious (see Fig. 5(e)). However, we can only get Fig. 5(c) without semantic matching where palpable artifacts are unavoidable. With semantic matching, the cloned result with the style of the trees in the artwork is more harmonious (see Fig. 5(e)). Another example is the oil drawing shown in Fig. 9, the apple is also semantically transferred from the one in the artwork.

The final distance D can now be defined as:

$$D(\mathbf{p},\mathbf{q}) = W(\mathbf{p},\mathbf{q})(L(\mathbf{p},\mathbf{q}) + \alpha_1 T(\mathbf{p},\mathbf{q}) + \alpha_2 C(\mathbf{p},\mathbf{q})) \quad (8)$$

where α_1 and α_2 are used for the balance of the distances of the three features. Empirically, $\alpha_1 = 0.005$ and $\alpha_2 \in [0.005, 0.1]$ can produce satisfactory results in our experiments.

Fig. 6 illustrates various effects by adjusting the parameters. Figs. 6(a) and (b) illustrate the effects of the direction feature parameter u. Figs. 6(c) and (d) do not take the local coherence into consideration whereas Fig. 6(e) improves the performance with the local coherence. Fig. 6(f) show the result by changing the size of the neighborhood.

4.3 Parallel Stylization Scheme

The neighborhood matching scheme in Step 6 of Algorithm 1 plays an important role in our approach. Traditional texture transfer techniques [5], [16], [6], [7] heavily rely on the scan-line scheme. Although they employ a coarse-to-fine multi-resolution approach to accelerate the convergence, the serial scanning makes it slow. Lefebvre and Hoppe [30], [31] present a parallel neighborhood-matching-based texture synthesis scheme. The high-quality synthesis is attained on the GPU using multi-resolution jittering together with coordinate upsampling and sub-pass correction. They use a fixed neighborhood for each resolution level. Different from them, we employ a parallel neighborhood matching scheme which performs in the original finest resolution to avoid the construction of image pyramids.

Our stylization scheme can be viewed as a variation of the jump flooding algorithm [32]. This scheme propagates quickly from each seed to neighboring samples in logarithmic steps. It has been adapted by Barnes et al. [33] to perform propagation over several iterations with a maximum jump step of 8 and 4 neighbors at each jump step. Their synthesis approach is in a coarse-to-fine manner and random search is employed in each pass. In this paper, we adapt the standard jump flooding scheme with two stages: the global optimization and the local optimization. The major difference between our approach and the existing parallel algorithms in [30], [31], [33] is that our approach operates on the original image without image pyramid generation and coarse-to-fine correspondence propagation.

In the global optimization, we set all the pixels in the image as seeds and flood them with four rounds. The jump steps are sequentially 2^{3k}, 2^{2k}, 2^k and 1, where k is $\lceil(\log_2 n)/3\rceil$. Although the attained result of the first stage is a highly noisy version of the object, it provides a good approximation for further propagations.

In the local optimization stage, we arrange four passes each of which is composed of four rounds and the corresponding step lengths are 8, 4, 2 and 1, respectively. We split the whole image into multiple 16×16-sized tiles and assign each tile with four seeds that carry smallest distances. We consider the fact that smaller distances corresponds to more similar feature styles.

In order to avoid visible incoherence along the boundaries of tiles, these boundary pixels should be updated appropriately by all their neighbors. Accordingly, we shift the tile by half of the tile size in horizontal and vertical directions, as illustrated in Fig. 7.

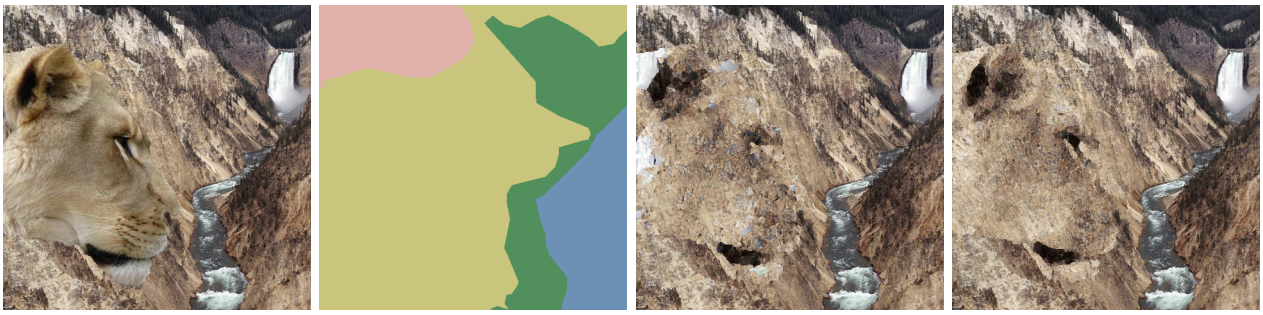
In each style matching round, we select the candidates for each pixel $\mathbf{p}$ from the shifted pixels which are respected to s-step neighboring pixels of $\mathbf{p}$. If one candidate is a seed and its distance is smaller than that of the current map correspondence, the mapped pixel is updated with this candidate. If a pixel is affected by multiple seeds, its correspondence is determined by the candidate which has the smallest distance. Consequently, the energy function E is minimized iteratively. We obtain the final solution when the reduction rate of the energy is smaller than a user-specified threshold τ. Fig. 8 shows an example of jump flooding with three rounds. Colored pixels in the leftmost chart are the original seeds with the smallest distance.

5 Interactive Editing Tools

Our framework provides not only basic operations such as scaling and moving objects, but also some premier op-

(a) Photos (b) Artwork (c) Segmentation (d) Casting shadow (e) Our cloning result

Fig. 9. Adding a coffee cup and an apple (a) into the still life oil painting (b), we can put them in arbitrary positions interactively. We put the apple behind the pear via layering based on the segmentation (c). We can also change the length and position of the shadow (d) by adjusting the frame with blue lines. The green point represents the highest point of the object and the red point is its corresponding shadow.



(a) Artwork/photo (b) Segmentation (c) Without Labeling (d) With Labeling

Fig. 10. Comparison between the results of without and with semantic labeling.

erations like occlusion, casting shadows, direction field editing, and semantic labeling. All the operations below are based on the interactive image segmentation [34].

Thanks to our interactive performance, all our operations become simpler and easier to use, which can be seen in the accompanied video. Empirically, most of the style-aware image cloning processes can be completed within one minute by taking advantage of these simple operations.

Layering We segment the artwork in advance and put the partitions in different layers. Users can move up and down the layers to change their occlusion relations with the cloned objects. They can also put the objects in arbitrary positions. In Fig. 9, the apple is occluded by the pear in the painting by layering.

Casting shadow Shadow provides viewers the cues for shape and depth perceptions. In cel animation, cartoon and computer-generated films, a broad set of techniques [35], [36], [37] have been employed to create shadow mattes. However, it is not trivial to add the shadow of the inserted object into the artwork since we do not know the geometry of the artwork and some artworks do not have clear lighting directions. Scientific studies show that the physics of shadow used by our visual brain is simpler than true physics and this fact has been used by artists [38]. As a result, artists can take some liberties when they draw shadows.

Therefore, we compute the shadow using a simplified shadow casting model and then project the shadow onto the image. We assume an orthogonal projection with a 45-degree angle between the image and the shadow receiver. Our model is based on a directional light and a plane shadow receiver. We use the mask of the inserted object to approximate the silhouette of its corresponding 3D model. As illustrated is Fig. 9(d), users indicate the light direction by adjusting a frame with blue lines, which represent an object on the ground and its shadow respectively. The yellow point is designed as the contact point between the inserted object and the shadow receiver. Finally, the shadow regions are darkened and stylized using our style transfer algorithm.

Semantic labeling The semantic labeling is used to enhance the result if imperfect matching happens using automatic semantic features. With the semantic labeling function, users can optionally specify the semantic similarity between the object and an arbitrary region in the artwork by clicking on the region. After the labeling, the sematic weight W of the user-specified region is set to 0.001 and that of other regions are assigned with 1.0. An example for semantic labeling is provided in Fig. 10 where the user specifies the rock region (yellow region) for the lion. The lion is expected to be hidden in the image with the texture of surrounding background. Without the semantic labelling, the texture of water

(a) Photo (b) Strokes/Segmentation (c) Before interaction (d) After interaction

Fig. 12. Comparison between the results of before and after direction field editing. The photo (a) is an render snapshot of the teapot mesh and is inserted into the drawing (Fig. 3(a)). We segment the teapot into several parts which have different direction (b). Thus, the direction of one part can be modified alone and directions of others remain unchanged. (c) and (d) are the results before and after the direction field editing.

(a) (b) (c) (d) (e)

Fig. 11. Comparison between the results of before and after direction field editing. Before the interaction, the direction field (b) has many undesired curls and the corresponding result (d) is the same as Fig. 5(e). After employing the direction field editing operation (a), the direction field is (c) and the corresponding result (e) integrates the cloned tree into the drawing harmoniously.

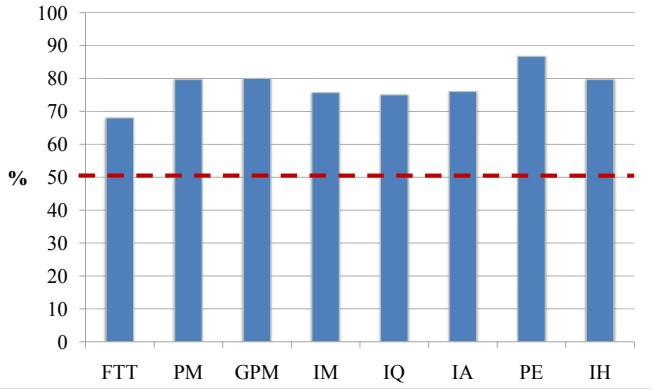


Fig. 13. Percentage of times which users chose ours over other algorithms. (1) Fast texture transfer (FTT), (2) PatchMatch (PM), (3) generalized PatchMatch (GPM), (4) image melding (IM), (5) image quilting (IQ), (6) image analogies (IA), (7) Poisson editing (PE), and (8) image harmonization (IH).

(white speckles) will be undesirably used to stylize the lion.

Direction field editing The direction field generated by the local structure estimation algorithm [23] is based on the gradient field of the image. In some cases, these automatically extracted direction fields are not appropriate for the artwork. See Fig. 11(b), the automatically computed direction fields of the cloned tree and the artistic trees are quite different. In another instance Fig. 12(c), the direction field of the teapot generated automatically fails to describe its geometrical characteristics. To improve the effectiveness of results like these, we allow users to edit the direction field by taking advantage of adjustable Bézier curves, as illustrated in Figs. 11(a) and 12(b). The directions of pixels on the Bézier curves are specified as the tangential directions of respective positions on the curves. We interpolate directions at the remaining pixels using Gaussian radial basis functions. As illustrated in Figs. 11 and 12, users can modify the direction field for the entire object or one part of the object based on the semantic segments with a few editing strokes.

6 User Study

We have devised a user study to objectively verify the effectiveness of our style-aware image cloning method. The user study consists of two cases. The first case compares the synthetic results of our method with several state-of-the-art methods to illustrate that our image cloning method achieves better effects. Eight out of ten samples in our user study are fully automatic. For the remaining two samples (Figs. 1 and 17), some user interactions are employed. To make fair comparisons, we do use the same segmentation for all methods. The segmentations in our method are mainly used for the layering and casting shadow operations. The layering operations designed for the occlusion effect are also used for other methods. The only user interaction not used in other methods is the casting shadow operation. The second case compares some original artworks with the synthetic ones produced by our method to justify that the synthetic parts match the original art style well. This case is designed to test the practicability of our method and the participants do not know the purpose of the study. We assume that participants prefer images with fewer noticeable style artifacts. It is expected that there are no remarkable differences between the two options. If so, it gives an evidence that our results do not introduce noticeable style artifacts.

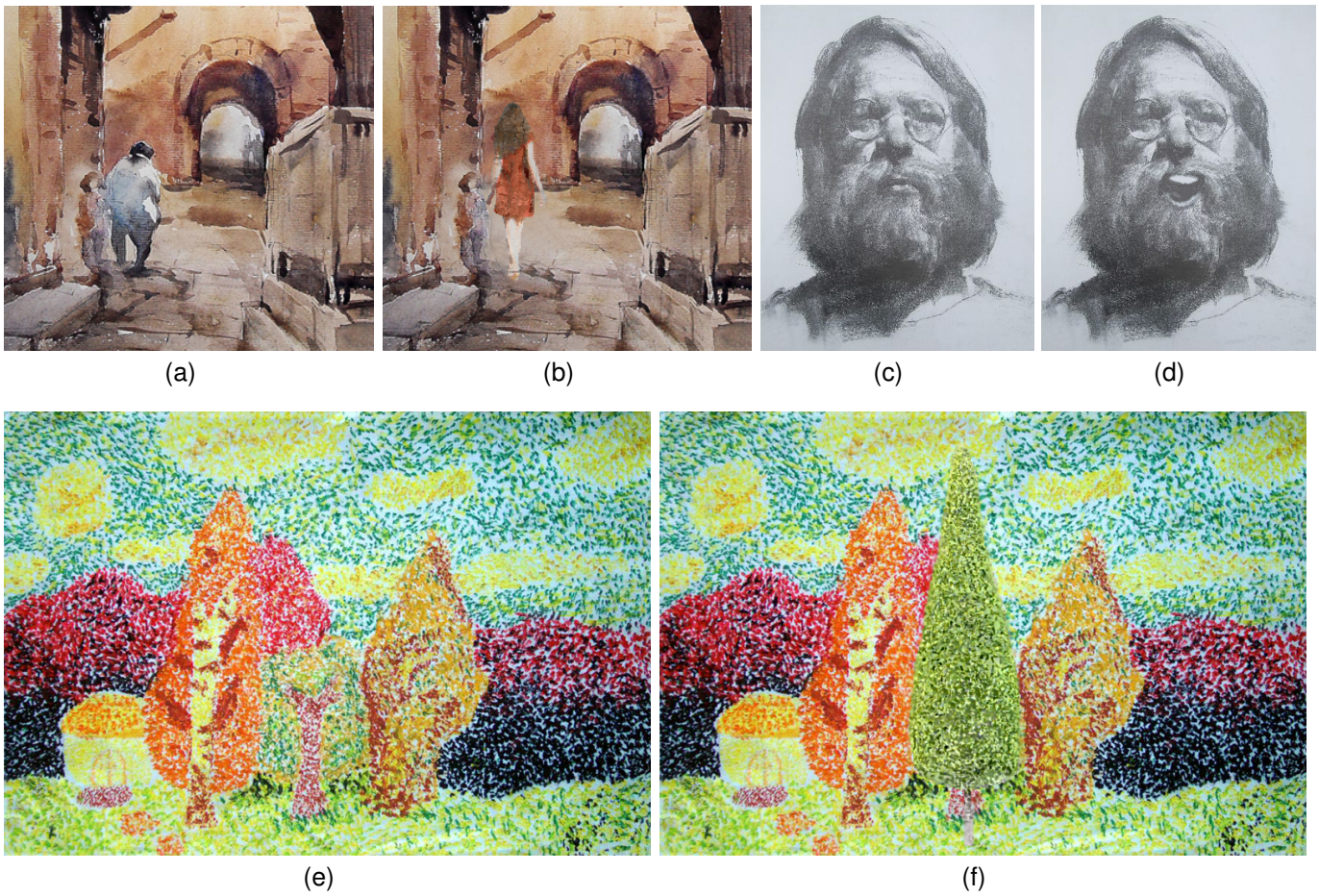


Fig. 14. Results designed for user study. (a), (c), (e) are original artworks, while (b), (d), (f) are our created results.

Study details. In the first case, the task consists of 10 pairs of synthesized images. The two images in each pair are produced by inserting the same objects into an artwork, and the inserted photorealistic objects were processed by our method and a method randomly chosen from (1) fast texture transfer [6], (2) PatchMatch [33], (3) generalized PatchMatch [39], (4) image melding [4], (5) image quilting [16], (6) image analogies [5], (7) Poisson editing [1], and (8) image harmonization [3]. We published our task on Amazon Mechanical Turk, a web-based marketplace that has been used for user studies [40], [41]. It allows requesters to offer paid "Human Intelligence Tasks" (HITs) to many non-expert workers. In one HIT, the order of images within each pair is random. The workers were asked to perform two-alternative forced choices (2AFCs), picking out the one with less artifacts from the two candidates, namely, the image looks harmonious as a whole. 114 workers completed a total of 240 HITs, taking 2 minutes 43 seconds on average.

For the second case, we prepared 4 pairs of images. Each pair contains a real-world artistic image and a synthesized one from our algorithm. Some parts of the original artworks were replaced with photorealistic objects in the synthesized images. We invited 30 volunteers who had different artistic background and areas of knowledge for this case. The 4 pairs were presented to participants in random order and random placements. Participants were asked to pick out the better looking one from each pair, no matter from which aspect they considered.

Results. We analyzed the data of the two cases separately. For the first case (see Fig. 13), when asked to choose the one with less artifacts from the artworks respectively created by our method and (1) the fast texture transfer method, our method gets 71.33% of the total times which the workers chose as shown in Fig. 13(1). It also gets 78.67% against (2) PatchMatch, 80.33% against (3) generalized PatchMatch, 78.00% against (4) image melding, 79.00% against (5) image quilting, 74.33% against (6) image analogies, 88.00% against (7) Poisson editing, and 79.33% against (8) image harmonization. By performing one-sample, one-tailed t-tests for all eight state-of-the-art methods, we found that workers preferred our method (p-values$\ll 0.001$).

In the second case of the user study, we applied our method to retouch artworks by replacing some parts of the original image with photorealistic objects while preserving stylistic harmony. We found that for sample 1, sample 2 (see Figs. 14(a), (b), (e) and (f)) and sample 4

Fig. 15. We generate different portraits ((b) and (d)) by replacing the original one (a) with face (f) (upper) and (f) (down). We also compare our results (b) and (d) with the image harmonization [Sunkavalli 2010] (c) and image melding [Darabi 2012] (e).

Fig. 16. Comparisons with other methods of map work. From left to right: (a) inputs (the map image is courtesy of Matthew Cusick); (b) results of our method; (c) image quilting [Efros and Freeman 2001]; (d) image analogies [Hertzman et al. 2001]; (e) image harmonization [Sunkavalli et al. 2010].

Fig. 17. Comparison with other methods. From left to right: (a) Our result; (b) Fast texture transfer [Ashikhmin 2003]; (c) Image analogies [Hertzmann 2001]; (d) Image quilting [Efros 2001]; (e) Patchmatch [Barnes 2009]; (f) Generalized Patchmatch [Barnes 2010].

(Figs. 15(a) and (b)), more than 50% volunteers chose our synthetic ones (53.3%, 60%, and 86.7% respectively). But for sample 3 (see Figs. 14(c) and (d)), the percentage is 36.7%, less than 50%. We performed the statistical analysis of this case using Pearson's χ^2 test and there was no significant difference between the two choices of our results and the original artworks ($p = 0.9106$). Through this user study, we can conclude that our approach can create harmonious artworks with no noticeable style artifacts.

7 Results and Discussions

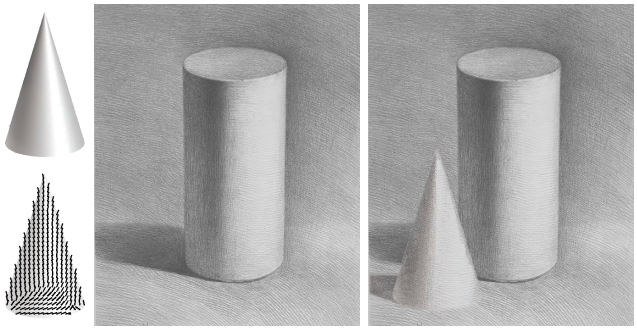
We have developed an interactive editing system to alter existing artworks using photorealistic or artificial objects. Examples with various styles are shown in Fig. 5 (crayon drawing), Fig. 14(a) (watercolor), Fig. 16 (map work), Fig. 17 (oil painting), and Fig. 18 (hatching drawing).

The image analogies [5] and patch-based texture transfer [16] are widely acknowledged approaches for style transfer. We compare our approach with them in Figs. 16 and 17. We extend their distance metric and add plausible shadow to get more harmonious results.

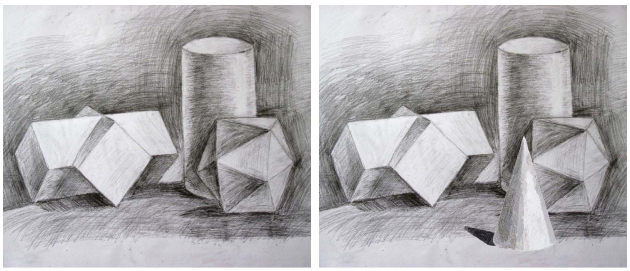
In addition to this, our framework can generate better

TABLE 1
Comparison of performance over other methods (in seconds)

example	size of artworks	size of inserted photos	FTT C++	IQ Matlab	PM CUDA	IH Matlab	IM Matlab	ours CUDA
Fig. 1	600×425	$312 \times 338, 144 \times 128$	0.270	15.273	0.154	7.452	743.333	0.038
Fig. 9	800×808	$308 \times 257, 258 \times 246$	0.330	38.694	0.161	12.425	1382.643	0.051
Fig. 16	372×468	192×184	0.110	4.998	0.070	6.330	264.255	0.024



(a) (b) Hatching drawing (c) Reasonable case



(d) Hatching drawing (e) Failure case

Fig. 18. Limitations for hatching drawings.

results (see Fig. 17) because we incorporate semantic information into the new distance metric. The employment of interaction in Fig. 17 is happened in layering and casting a plausible shadow.

PatchMatch [33] accelerates the nearest-neighbor matching by using a randomized patch search. The upgraded version [39] adds rotations and scales for computer vision applications. When the distance between matching patches are very small, which is tenable for most applications of photograph in computer vision, the results are nice. The comparisons with the approach are given in Fig. 17.

Image harmonization [3] matches the color, contrast, noise, texture and blur of the source to those of the target images. It produces pleasing results under the assumption that the target is a good model to match the source, and noise and texture are stochastic. When the assumption does not hold, artifacts may arise (see Fig. 16(e)). We have compared our approach with it through the example of human face transplanting in Fig. 15. The eyes in Fig. 15(c) are still very clear while our method Fig. 15(b) preserves the indistinct style as showed in Fig. 15(a).

Image melding [4] makes excellent combinations of the sources with different textures and structures, and its harmonization application can handle the failure case in image harmonization [3]. However, when the source and target are largely different, it may produce disharmonious artifacts (see Fig. 1(e)). We also compare its harmonization application with our approach in Fig. 15(e) by using a scene similar to that used in Fig. 15(c).

The comparisons with Poisson image editing are shown in Fig. 1(d). The cloned results of Poisson image editing look like photographs rather than artistic images. In contrast, our approach can clone the style of an artwork in a seamless way, and generates more harmonious results for such artworks.

Our system is implemented using NVIDIA CUDA programming environment [42]. We ran the program on a 3.40GHz Intel Core i7-2600 CPU and an NVIDIA GeForce GTX 590 GPU. We compare our method with other representative methods on the same machine for performance comparison (see Table 1). Our approach costs less than 0.1 seconds for a cloned object with 180,000 pixels. The most time-consuming interaction time is spent on segmentation, which costs less than one minute and it is the same for all methods. The editing operations used in Figs. 1 and 9 are layering and casting shadow (see the supplemental video). We use Adobe Photoshop CS5 to achieve the occlusion effect of layering for other methods. Fig. 16 is created automatically.

Our method can deal with most kinds of artworks, including Vincent Van Gogh's paintings. However, it may fail for hatching drawings (see Fig. 18). For such works, tonal or shading effects are created by painting closely spaced parallel lines or cross lines. Therefore, the directions of these lines are not directly related to the variation of shadings. A reasonable result can be generated by setting u to 0 (see Fig. 18(c)).

8 Conclusion and Future Work

We have presented a novel interactive rendering framework for cloning photo-realistic or artificial objects into real-world artworks seamlessly. The harmonization between the cloned objects and the artwork is achieved by transferring the style features encoding luminance, texture, direction, local coherence, and semantic information in the artistic images. Extensive experiments have been conducted to demonstrate the effectiveness of the proposed method. Our new image cloning approach can facilitate designers to create new artworks by utilizing

existing images with a high fidelity. Even amateurs can also enjoy themselves through creating various non-photorealistic images with our method.

In order to reduce the illumination difference between the cloned object and the artwork, our approach adjusts the color of the cloned image by histogram matching. However, such an adjustment cannot simulate the light interactions between the cloned object and the artwork. As a result, the lights in the artwork cannot illuminate the cloned object. Although we have simulated shadow casting by projecting the contours of cloned objects through some user interactions, a better solution which can realistically insert cloned objects into existing artworks accounting for their lighting interactions should be developed. The work on rendering synthetic objects into legacy photographs by Karsch et al. [20] provides some inspirations for our future work.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their constructive comments. Many thanks also to Liwen Hu, Tianmin Zou, Junjie Chen and Ke Wang for their dedicated help. Xiaogang Jin was supported by the National Natural Science Foundation of China (Grant nos. 61472351, 61328204). Yingqing Xu was supported by the National Basic Research Program of China (Grant No. 2012CB725304) and the National Natural Science Foundation of China (Grant Nos. 61272234, 61373072). Hanli Zhao was supported by the National Natural Science Foundation of China (Grant No. 61100146).

REFERENCES

[1] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," *ACM Trans. Graphics*, vol. 22, no. 3, pp. 313–318, 2003.
[2] Z. Farbman, G. Hoffer, Y. Lipman, D. Cohen-Or, and D. Lischinski, "Coordinates for instant image cloning," *ACM Trans. Graphics*, vol. 28, no. 3, pp. 67:1–67:9, 2009.
[3] K. Sunkavalli, M. K. Johnson, W. Matusik, and H. Pfister, "Multi-scale image harmonization," *ACM Trans. Graphics*, vol. 29, no. 4, pp. 125:1–125:10, 2010.
[4] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen, "Image melding: combining inconsistent images using patch-based synthesis," *ACM Trans. Graphics*, vol. 31, no. 4, pp. 82:1–82:10, 2012.
[5] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, "Image analogies," in *Proceeding of ACM SIGGRAPH'01*. ACM, 2001, pp. 327–340.
[6] M. Ashikhmin, "Fast texture transfer," *IEEE Comput. Graph. Appl.*, vol. 23, no. 4, pp. 38–43, 2003.
[7] H. Lee, S. Seo, S. Ryoo, and K. Yoon, "Directional texture transfer," in *Proceeding of the 8th International Symposium on Non-photorealistic animation and rendering (NPAR)*, 2010, pp. 43–48.
[8] M. Cheng, F. Zhang, N. J. Mitra, X. Huang, and S. Hu, "Repfinder: finding approximately repeated scene elements for image editing," *ACM Trans. Graphics*, vol. 29, no. 4, pp. 83:1–83:8, 2010.
[9] J. McCann and N. S. Pollard, "Real-time gradient-domain painting," *ACM Trans. Graphics*, vol. 27, no. 3, pp. 93:1–93:7, 2008.
[10] S. Xue, A. Agarwala, J. Dorsey, and H. Rushmeier, "Understanding and improving the realism of image composites," *ACM Trans. Graphics*, vol. 31, no. 4, pp. 84:1–84:10, 2012.
[11] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Comput. Graph. Appl.*, vol. 21, no. 5, pp. 34–41, 2001.
[12] X. Xiao and L. Ma, "Gradient-preserving color transfer," *Computer Graphics Forum*, vol. 28, no. 7, pp. 1879–1886, 2009.
[13] H. Huang, Y. Zang, and C.-F. Li, "Example-based painting guided by color features," *Vis. Comput*, vol. 26, no. 6-8, pp. 933–942, 2010.
[14] Y. Xiao, L. Wan, C.-S. Leung, Y.-K. Lai, and T.-T. Wong, "Example-based color transfer for gradient meshes," *IEEE Trans. Multimedia*, vol. 15, no. 13, pp. 549–560, 2013.
[15] P. Bénard, F. Cole, M. Kass, I. Mordatch, J. Hegarty, M. S. Senn, K. Fleischer, D. Pesare, and K. Breeden, "Stylizing animation by example," *ACM Trans. Graphics*, vol. 32, no. 4, pp. 119:1–119:12, 2013.
[16] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proceeding of ACM SIGGRAPH '01*, 2001, pp. 341–346.
[17] I. Drori, D. Cohen-Or, and H. Yeshurun, "Example-based style synthesis," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003, pp. 143–150.
[18] J. Fischer and D. Bartz, "Stylized augmented reality for improved immersion," in *Proceedings of the IEEE Conference on Virtual Reality (VR)*, 2005, pp. 195–202.
[19] M. Zhao and S.-C. Zhu, "Customizing painterly rendering styles using stroke processes," in *Proceeding of the 9th International Symposium on Non-photorealistic animation and rendering (NPAR)*, 2011, pp. 137–146.
[20] K. Karsch, V. Hedau, D. Forsyth, and D. Hoiem, "Rendering synthetic objects into legacy photographs," *ACM Trans. Graphics*, vol. 30, no. 6, pp. 157:1–157:12, 2011.
[21] J. Lopez-Moreno, E. Garces, S. Hadap, E. Reinhard, and D. Gutierrez, "Multiple light source estimation in a single image," *Comput. Graph. Forum*, vol. 32, no. 8, pp. 170–182, 2013.
[22] H. Winnemöller, S. C. Olsen, and B. Gooch, "Real-time video abstraction," *ACM Trans. Graphics*, vol. 25, no. 3, pp. 1221–1226, 2006.
[23] J. E. Kyprianidis and H. Kang, "Image and video abstraction by coherence-enhancing filtering," *Comput. Graph. Forum*, vol. 30, no. 2, pp. 593–602, 2011.
[24] J. Chen, G. Guennebaud, P. Barla, and X. Granier, "Non-oriented MLS gradient fields," *Comput. Graph. Forum*, vol. 32, no. 8, pp. 98–109, 2013.
[25] M. Ashikhmin, "Synthesizing natural textures," in *proceeding of the symposium on Interactive 3D graphics and games (I3D)*, 2001, pp. 217–226.
[26] X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo, and H.-Y. Shum, "Synthesis of bidirectional texture functions on arbitrary surfaces," *ACM Trans. Graphics*, vol. 21, no. 3, pp. 665–672, 2002.
[27] Y. Qu, T.-T. Wong, and P.-A. Heng, "Manga colorization," *ACM Trans. Graphics*, vol. 25, no. 3, pp. 1214–1220, 2006.
[28] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 8, pp. 837–842, 1996.
[29] P. Panareda Busto, C. Eisenacher, S. Lefebvre, and M. Stamminger, "Instant texture synthesis by numbers," in *Proceeding of Vision, Modeling, and Visualization Workshop (VMV)*, 2010, pp. 81–85.
[30] S. Lefebvre and H. Hoppe, "Parallel controllable texture synthesis," *ACM Trans. Graphics*, vol. 24, no. 3, pp. 777–786, 2005.
[31] ——, "Appearance-space texture synthesis," *ACM Trans. Graphics*, vol. 25, no. 3, pp. 541–548, 2006.
[32] G. Rong and T.-S. Tan, "Jump flooding in gpu with applications to voronoi diagram and distance transform," in *Proceeding of the symposium on Interactive 3D graphics and games (I3D)*, 2006, pp. 109–116.
[33] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: a randomized correspondence algorithm for structural image editing," *ACM Trans. Graphics*, vol. 28, no. 3, pp. 24:1–24:11, 2009.
[34] F. Meyer, "Color image segmentation," in *Proceeding of IEEE International Conference on Image Processing and its Applications (IPA)*, 1992, pp. 303 – 306.
[35] L. Petrović, B. Fujito, L. Williams, and A. Finkelstein, "Shadows for cel animation," in *Proceeding of ACM SIGGRAPH '00*, 2000, pp. 511–516.
[36] F. Pellacini, P. Tole, and D. P. Greenberg, "A user interface for interactive cinematic shadow design," *ACM Trans. Graphics*, vol. 21, no. 3, pp. 563–566, 2002.
[37] E. Sugisaki, H. S. Seah, F. Tian, and S. Morishima, "Interactive shadowing for 2d anime," *Comp. Anim. Virtual Worlds*, vol. 20, no. 2-3, pp. 395–404, 2009.
[38] P. Cavanagh, "The artist as neuroscientist," *Nature*, vol. 434, no. 7031, pp. 301–307, 2005.

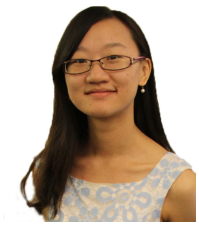
[39] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein, "The generalized patchmatch correspondence algorithm," in *Proceedings of the 11th European conference on computer vision conference on Computer vision (ECCV)*, 2010, pp. 29–43.
[40] J. Lu, F. Yu, A. Finkelstein, and S. DiVerdi, "Helpinghand: example-based stroke stylization," *ACM Trans. Graphics*, vol. 31, no. 4, pp. 46:1–46:10, 2012.
[41] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?" *ACM Trans. Graphics*, vol. 31, no. 4, pp. 44:1–44:10, 2012.
[42] NVIDIA, "Cuda c programming guide (version 4.2)," 2012. [Online]. Available: http://docs.nvidia.com/cuda/cuda-c-programming-guide/

Yandan Zhao received the B.Sc. degree in computer science from Jilin University, P. R. China, in 2008. She is currently working toward the Ph.D degree in computer science at the State Key Laboratory of CAD&CG, Zhejiang University. Her main research interests include texture synthesis and non-photorealistic rendering.

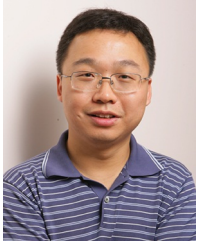
Xiaogang Jin received the B.Sc. degree in computer science and the M.Sc. and Ph.D degrees in applied mathematics from Zhejiang University, P. R. China, in 1989, 1992, and 1995, respectively. He is a professor in the State Key Laboratory of CAD&CG, Zhejiang University. His current research interests include implicit surface computing, special effects simulation, digital geometry processing, texture synthesis, crowd animation, cloth animation, computer-generated marbling, and non-photorealistic rendering. He is a member of the IEEE.

Yingqing Xu received his B.Sc. from the Department of Mathematics of Jilin University, and his Ph.D from the Institute of Computing Technology, Chinese Academy of Sciences (CAS). He is a professor in Department of Information Art & Design, Tsinghua University. His current research interests include natural user interface design, computer graphics, computer vision, e-Heritage and virtual reality.

Hanli Zhao is an associate professor of Wenzhou University, Wenzhou, China. He received his B.Sc. degree in software engineering from Sichuan University in 2004 and his Ph.D degree in computer science from Zhejiang University in 2009. His research interests include non-photorealistic rendering and general purpose GPU computing.



Meng Ai received the B.Sc. degree in computer science from Dongbei University, P. R. China, in 2011. She is currently working toward the M.Sc. degree in computer science at the State Key Laboratory of CAD&CG, Zhejiang University. Her main research interests include mesh editing, facial animation, non-photorealistic rendering, and image processing.



Kun Zhou is a Cheung Kong Distinguished Professor in the Computer Science Department of Zhejiang University, and a member of the State Key Lab of CAD&CG, where he leads the Graphics and Parallel Systems Group. Prior to joining Zhejiang University in 2008, Dr. Zhou was a Leader Researcher of the Internet Graphics Group at Microsoft Research Asia. He received his BS degree and PhD degree in computer science from Zhejiang University in 1997 and 2002, respectively. His research interests include shape modeling/editing, texture mapping/synthesis, real-time rendering, and GPU parallel computing.