Artistic Rendering of Human Portraits Paying Attention to Facial Features

Mahdi Rezaei

Dept. of Computer Science, The University of Auckland, New Zealand
mrez010@aucklanduni.ac.nz

Abstract. Artistic painting of human portraits are more challenging than landscapes or flowers. Challenges are eye and nose areas where we need to avoid alterations from their natural appearance. Shades or darkness around eyes, or shininess at the nose tip may negatively impact the rendering result if not properly dealt with. The proposed computerized method attempts to be adaptive to those sensitive areas by utilizing a face analysis module. First, the program detects meaningful face segments, and then it utilizes a blending of various filtering parameters to make the final portrait as clear as possible while still supporting an 'artistic painting' in overall.

Keywords: Non-photorealistic rendering, artistic filter, pointillism, Glass pattern, curves and strokes style, facial features.

1 Introduction

Photorealistic rendering aims at creating digital images that look like a photograph. In contrast, *non-photorealistic rendering* (NPR) produces images of simulated worlds, in styles that are different to realism [1]. An artistic filter implements a non-photorealistic rendering technique for transforming a given image with some 'artistic effect' (Fig. 1). Those effects aim to emulate the styles followed by painters, such as impressionism or cubism. Digital NPR bring together art and image processing [2]. Nevertheless, a real painter is normally better at determining what visual style has the best artistic effect, since creativity itself cannot be easily simulated by a computer.

NPR rendering for a flower, a building, or a landscape are considerably easier than a human face. NPR program easily fails on facial features. A program may

Fig. 1. A portrait photograph and a digital NPR example of this photo [1]

A.L. Brooks (Ed.): ArtsIT 2011, LNICST 101, pp. 90–99, 2012.

have difficulties in understanding why a shade around one eye should not lead to a different representation compared to the other eye,or a specularity on a nose should not lead to bended 'brush strokes' around the specularity. The paper proposes an NPR approach that combines options for creative artistic rendering (based on prior work [4]) with a new focus on human portrait particularities, resulting altogether in an artistic painting style.

2 Related Works

A variety of painterly rendering techniques were developed in the early 1990s, starting with Haeberli's pioneering work in [6], which introduced computerized paintings with an ordered collection of strokes described by size, shape, colour, and orientation. The method tried to simulate an impressionistic style. Hertzmann et al. described a two-phase framework called 'image analogies' [3]. In the first phase (the 'design phase'), they used a pair of images, one original image and one filtered image as the training data. Then, in the second phase (the 'application phase'), and following by texture synthesis, the learned filter is applied to some new target images to develop analogous filtered images.

Papari and Petkov [5] proposed an idea to replace natural texture in an input image, with a synthetic painterly texture that is generated by means of a continuous Glass pattern, while its geometrical structure is controlled by the gradient orientation of the input image.

Valente and Klette [4] developed a triangular user interface that allows the user to select different styles of painting according to the users taste (Fig. 2). Each corner of the triangle represents one style, and any point inside the triangle means a combination of two painting style.

DiPaola provided an interdisciplinary method that uses a parametrised approach to approximate a knowledge domain for painterly rendering of portraits [7]. The knowledge domain uses fuzzy knowledge rules gained from interviews with oil portrait painters, data from the traditional 'portrait painter process' combined with human vision techniques and semantic models of the face and upper torso. Except Dipaola's, none of the other NPR works focused on sensitive facial features so far.

Fig. 2. Triangular GUI for filter blending

3 Painting Styles and Filter Design

As the first step we briefly review on simulation of three painting styles namely *curves-and-strokes*, *pointillism*, and *Glass pattern* which are provided by [4]. Then in section 4, we go for facial feature detection (eye and nose regions), and finally we apply a softened filtering approach in eye and nose regions to generate a more pleasant and sensible artistic renderings.

3.1 Curves-and-Strokes Filter

This filter generates an output image with an appearance of hand-painted brushes. The important point in this method is how to put strokes on the canvas by an appropriate brush size in different areas of images. The algorithm is a three layer painting, takes the source image, and then determines the following information:

- Image size (W_i, H_i)
- Face detection and then detecting face width (W_f)
- Proportional size of detected *face* to the entire *image size* $(p\%)$

Details of face detection and localization are described in Section 4. Based on the p value and inspiring by [3], we select three brush sizes $R_1 = 0.02 \cdot W_i$, $R_2 = 0.01 \cdot W_i$, $R_3 = 0.006 \cdot H_i$. With those brush sizes we got a satisfying appearance. The method starts putting the strokes on the canvas with the largest brush (R_1). To a painter, a painting seems more artistic and professional, if we use mainly larger brushes and just use smaller brushes only in sensitive regions to show details. Therefore, at first, we paint the whole image with the largest brush R_1. Then, applying a convolution of the created painting with the source image, the algorithm determines the mean colour difference to the source image. If the difference is greater than a threshold T, then next smaller brush repaints the canvas, otherwise it remains unchanged:

```
function paint(sourceImage, Rm ... Rn)
        {
           canvas = a new constant color image
           // paint the canvas
                for (i=n ; i > m ; i--)
                from largest brush to smallest do
                  {
                   // Apply Brush Ri
                   Canvas = sourceImage * I(Ri);
                   // Threshold in painted layer
                   Threshold (canvas, SourceImage, Ri, Ti)
                        If (Ti <= T) then
                            Exit (loop);
                  }
           return canvas;
        }
```

Following the above algorithm, we will have two benefits of having a more artistic appearance, as well as a logarithmically decrease in computational cost. Figure 3 shows a sample result. The overall output looks pleasant, however, the eyes seem abnormally deviated from their natural appearance, and the nose also a

Fig. 3. Curved brush strokes on a sample portrait with bright eyes and light skin

bit bended. a simple way to fix this problem, is using a smaller brush sizes for the whole painting, to sustain more similarity to the reference image. In spite of the fact that this basic solution can solves the eye and nose problem, but the overall level of artistic painting decrease. As a better approach we tried to keep painting the images at first with larger brushes, then using smaller strokes and brushes *just* for the eye region, as well as applying some image enhancement in the nose area. Section 4, provides the detail for eye and nose detection method.

3.2 Pointillism

For emulating pointillism, it is helpful to apply Seurat's theoretical work [9]. There are two main reasons for computerizing pointillism. The painting process of pointillism is highly exhausting, so unloading the main proportion of the manual painting task is very desirable. Secondly, we are able to approximate pointillism by simpler point-like strokes. For emulating pointillism, we break up the image into three layers, each of which distributes the points in different ways.

The background layer fills up the canvas with large points which have minimal colour distortion. The middle layer adds smaller dots of stronger colours to areas where the brightness of the picture differs from the original by a threshold T_p. The final layer provides some edge enhancement to prevent details so far hidden by the larger dots. At each layer, dots are painted in a random order by using a Z-buffer, where each point's Z-value is initialized by a random number.

Fig. 4. Illustration of a naive emulation of a pointillistic painting. Defected eyes and nose issues (e.g. 'holes') can be seen on the right.

As an advantage comparing to curved strokes method, this filter uses only two parameters that need to be altered. The values that need to be defined is the brush size R and the decreasing rate of size for each layer (default $d = 25\%$). To make it simple, uniform circles of plain colours are used in the implementation. The filter emulates Seurat's painting style by breaking the image into a series of points, restricting the colour palette, and incorporating the idea of divisionism. Figure 4 shows a pointillistic painting. In this sample we have multiple faces, so we consider the average width of detected face as W_f.

3.3 Glass Patterns

This filter generates impressionistic images in the style of Van Gogh; inherent geometric structures are modeled as Glass patterns and transformed into the target image via a random point set. The geometric transformation uses information about the image's gradient to create an effect of impressionistic whirls around image contours. The algorithm is a modified version of the work by Papari and Petkov [5]. Similar to curves-and-strokes and pointillism, we implemented a code based on OpenCV that defines two parameters of Glass patterns (GP), the step size (h) and the standard deviation based on the proportion of the face size to the image size, $\rho = \frac{W_f}{W_I}$, and a grid histogram deviation, respectively. The initial step in the Glass pattern method is an edge preserving smoothing (EPS), which eliminates small textures from the input image while keeping the main object contours. We denote the output of EPS by $I_{EPS}(r)$. The next step is the generation of synthetic painterly textures (SPT) which simulates oriented brush strokes. Assume $\nabla_\sigma I_{EPS}$ as the scale-dependent gradient of I_{EPS}, which is created by convolution of I_{EPS} with the gradient of the Gauss function g_σ:

$$\nabla_\sigma I_{EPS} = I_{EPS} * \nabla_{g_\sigma}$$

Finally we apply a typical Glass pattern such as shown in Fig. 5 at random locations of the image with different strengths of S between 0 to 1. The whirl value θ_w can be a random or a fixed value equals $\pi/4$, $\pi/2$, $3\pi/4$, or π.

The implemented filter gives the look of an impressionistic oil painting by shifting pixels around the canvas in a pattern of impressionistic whirls. The Glass pattern is successful in providing 'pleasant' visual effects for small images, but, because colour values rotate around some fix points, it does not apply well to large portraits, and again some refinements in eye areas are compulsory (Fig. 6).

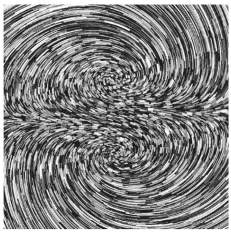

Fig. 5. A sample whirled Glass pattern

Fig. 6. Naive impressionistic-style emulation for dark skin subject, showing distortion in the eye regions

4 Haar Classifiers for Face, Eye and Nose Detection

For the facial component detection, we apply Haar-like feature matching based on the Viola-Jones object detection ideas [8] and our previous work [10]. Figure 7 shows mapping the Haar wavelet into a 2D image processing concept and a corresponding Haar feature in common black-white representation.

4.1 Detector Structure

The detector combines three techniques: (1) the use of a comprehensive set of Haar-like features that are in analogy to base functions of the Haar transform (Fig. 8), (2) the application of a boosted algorithm to select a limited set of appropriate Haar features for classifier training, and (3) forming a cascade of strong classifiers by merging week classifiers.

The detection process of a query object (e.g. a face) is based on some important characteristics of the object and the value distributions in dark or light regions of a feature that models expected intensity distributions. For example, the feature in Fig. 9, left, relates to the idea that in all faces there are darker regions of eyes compared to the bridge of the nose. Similarly, Fig. 9, right, models that the central part of an eye (the iris) is darker than the sclera.

4.2 Integral Image

To determine the presence or absence of Haar-like features, we use *integral images*. For pixel $p = (x, y)$, the *integral value*

$$I(p) = \sum_{1 \le i \le x \wedge 1 \le j \le y} P(i, j)$$

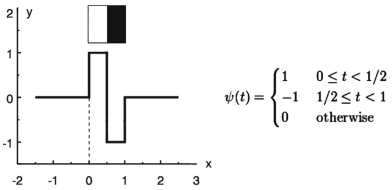

Fig. 7. Haar wavelet specification

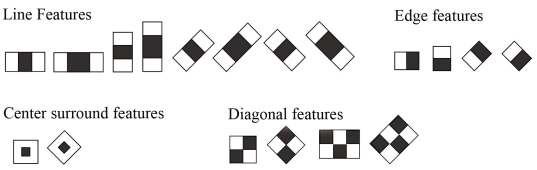

Fig. 8. Main categories and shapes of Haar-like features

is the sum of all pixel values $P(q)$, where pixel $q = (i, j)$ is above and left of p.

Considering $p_1, \ldots, p_4$ as the corners of rectangle D, the sum of all pixel values for D equals (see Fig. 10)

$$I(D) = I(p_4) + I(p_1) - I(p_2) - I(p_3)$$

Then, we follow with a Haar-like feature such as $R_1R_2R_3$. After calculating $I(R_i)$, values at rectangular regions are weighted by reals $\omega_i > 0$, defining regional values $\omega_i \cdot I(R_i)$. In combination, regional values define the *feature value* for a given *Haar-like feature*. For the above example, we have that $I(F_k) = \omega_1 \cdot I(R_1) - \omega_2 \cdot I(R_2) + \omega_3 \cdot I(R_3)$. Since the region R_2 is black, thus the weight ω_2 is negative, opposite to those of R_1 and R_3.

5 Cascaded Classifiers and Experimental Results

In order to classify object regions of a face such as eye and nose, we use a sliding window to search the image area in different sizes to match different Haar-like features. Figure 11 shows the structure of a cascaded classifiers. We used AdaBoost machine learning to select appropriate Haar-like features among thousands of possible Haar-like features, for each stage of the classifier. Applying an image data set of positive face images, the machine learning algorithm analyses the provided features F_k and adjusts weights W_k, thresholds T_k, and other parameters to maximize the performance according to the goal of detecting the faces. Heavily weighted filters come first, to eliminate non-face image regions as quickly as possible. The initial classifier simply rejects non-object (none-face) regions, if the basic predefined features do not exist.

If an image region includes this feature set with a value greater than the predefined threshold τ, then the system goes for the next stage, otherwise it rejects the region as being a non-object. more details in our previous work [10].

Fig. 9. Application of two sample rectangular features for face and eye detection

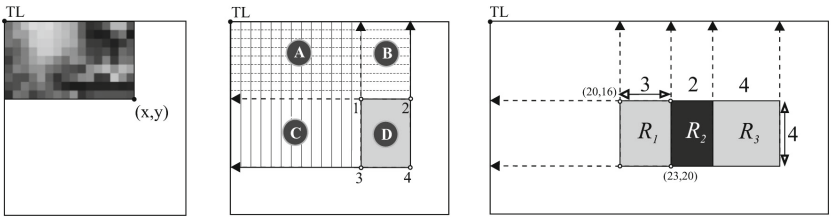

Fig. 10. Calculating integral values for a pixel, a rectangle, and a Haar-like feature

6 Filter Blending

Assume $Img1$ as the input portrait. Before applying an artistic filter, we run the face, eye and nose detector. Then we save an unchanged copy of these detected regions as $Eye1L$, $Eye1R$, $Nose1$ (original eyes and nose). We also save the position ans size of these regions as (x_i, y_i, w_i, h_i). Then, we apply artistic rendering with default parameters set of A to the original image ($Img1$) and we save the new image as $Img2$. afterwards, we apply artistic rendering with modified parameters set of B (as defined in 3.1., 3.2., and 3.3.) on three regions of $Eye1L$, $Eye1R$, $Nose1$ and save the resulted regions as $Eye2L$, $Eye2R$, and $Nose2$. Parameters set of B actually applies smoother effects than parameters set of A. Finally, we remove harsh regions of eye and nose from $Img2$, and replace new smoothed eyes and nose at the same places of x_i, y_i. The final result is $Img3$.

7 Experimental Results and Conclusions

Figures 12, 13, and 14 show the final images for three discussed artistic rendering. The paper utilized a hybrid approach of artistic rendering and facial feature detection to emulate portrait painting in NPR style. The experimental results for three techniques of curves-and-stokes, pointillism, and Glass patterns are satisfying in terms of eye and nose improvements while keeping the overall painting interesting in appearance. As future work, the method could be expanded for other artistic styles. Also, developing a method to ensure 'non-uniform' borders would make the rendered image more similar to a real painter's artwork.

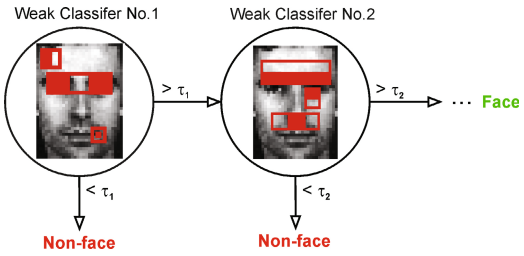

Fig. 11. Haar-feature matching inside the weak classifiers

Fig. 12. Curve and Stroke Painting. (a): Initial output with max brush size $R_M = 7$, min brush size $R_n = 3$. (b): Detected face/eye, and, blurred nose with Gaussian aperture width of $p = 11$ (d), (e) centre: Before modification. (d), (e) right: Eye and nose modification with $R_{f_M} = 0.85R_M$ and $R_{f_n} = 0.33R_n$. (c): Final result.

Fig. 13. Pointillistic Painting. (a): Initial output with brush size of $R = 5$ and stroke strength of $s = 1$ (b): Detected faces, eyes and noses (d), (e), (f) centre: Before modification. (d), (e) right: Eyes after modification with $b_e = 0.58R$ and $s_e = s$. (f) right: Nose modification with $b_n = 0.7R$ and $s_n = 0.8s$. (c): Final result.

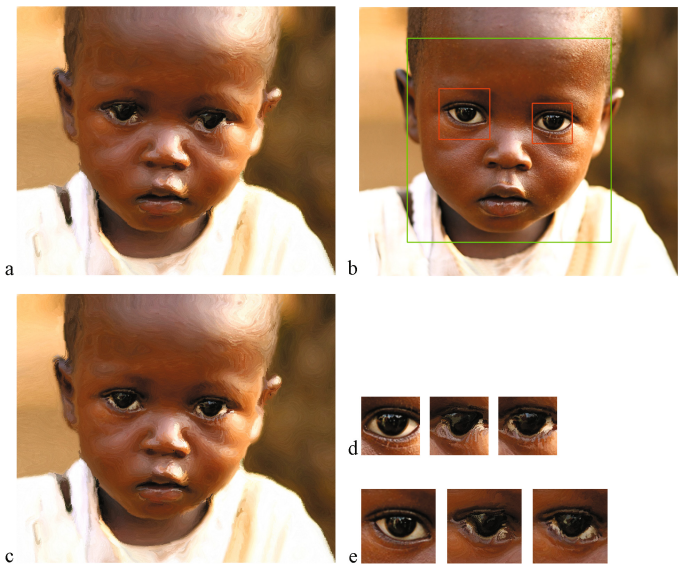

Fig. 14. Glass pattern painting. (a): Initial output with GP step size of h=0.4, $\theta_w = \pi/2$, and s= 0.5 (b): Detected face and eyes. (d), (e) centre: Before modification, (d), (e) right: After modification with $h_f = 0.3h$ and $s_f = 0.8s$. (c): Final result. Note: no nose detection and modification is required in Glass pattern painting.

References

1. Mignotte, M.: Unsupervised Statistical Sketching for Non-photorealistic Rendering Models. In: International Conference on Image Processing, Barcelona, Spain, vol. III, pp. 573–576 (2003)
2. Gooch, B., Gooch, A.: Non-Photorealistic Rendering. AK Peters Ltd. Publisher (2001)
3. Hertzmann, A., Jacobs, C., Oliver, N., Curless, B., Salesin, D.: Image analogies. In: Proc. SIGGRAPH, pp. 327–340 (2001)
4. Valente, C., Klette, R.: Artistic Emulation - Filter Blending for Painterly Rendering. In: Proc. Fourth Pacific-Rim Symposium on Image and Video Technology, Singapore, pp. 462–467 (2010)
5. Papari, G., Petkov, N.: Continuous Glass patterns for painterly rendering. IEEE Trans. Image Processing 18, 652–664 (2009)
6. Haeberli, P.: Paint by numbers: Abstract image representations. In: Proc. SIGGRAPH, pp. 207–214 (1990)
7. DiPaola, S.: Exploring a Parametrised Portrait Painting Space. Int. Journal of Art and Technology 2, 82–93 (2009)
8. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR, pp. 511–518 (2001)
9. Yang, C.K., Yang, H.L.: Realization of Seurat's pointillism via non-photorealistic rendering. The Visual Computer 24, 303–322 (2008)
10. Rezaei, M., Klette, R.: 3D Cascade of Classifiers for Open and Closed Eye Detection in Driver Distraction Monitoring. In: Real, P., Diaz-Pernil, D., Molina-Abril, H., Berciano, A., Kropatsch, W. (eds.) CAIP 2011, Part II. LNCS, vol. 6855, pp. 171–179. Springer, Heidelberg (2011)