# Learnable Stroke Models for Example-based Portrait Painting

Tinghuai Wang[1]
Tinghuai.Wang@surrey.ac.uk
John Collomosse[1]
J.Collomosse@surrey.ac.uk
Andrew Hunter[2]
Andrew.Hunter@hp.com
Darryl Greig[2]
Darryl.Greig@hp.com

[1] Centre for Vision Speech and Signal Processing
University of Surrey
Guildford, UK

[2] Hewlett Packard Laboratories.
Stoke Gifford,
Bristol, UK

**Abstract**

We present a novel algorithm for stylizing photographs into portrait paintings comprised of curved brush strokes. Rather than drawing upon a prescribed set of heuristics to place strokes, our system learns a flexible model of artistic style by analyzing training data from a human artist. Given a training pair — a source image and painting of that image — a non-parametric model of style is learned by observing the geometry and tone of brush strokes local to image features. A Markov Random Field (MRF) enforces spatial coherence of style parameters. Style models local to facial features are learned using a semantic segmentation of the input face image, driven by a combination of an Active Shape Model and Graph-cut. We evaluate style transfer between a variety of training and test images, demonstrating a wide gamut of learned brush and shading styles.

## 1 Introduction

The stylization of photographs into high quality digital paintings remains a challenging problem in computer graphics. In recent years, sophisticated *painterly rendering* algorithms have been proposed that rely increasingly upon Vision to interpret structure and drive the rendering process [4, 26]. Although such algorithms generate a pleasing aesthetic for many image classes e.g. scenic shots, they typically perform poorly on portraits. The human visual system has a strong cognitive prior for portraits, and is particularly sensitive to distortion or loss of detail around facial features [19]. Yet such artifacts are frequently observed when applying general purpose painterly algorithms to photographs of faces. High quality rendering of faces is important, as many scenarios for artistic stylization focus upon movie post-production, or consumer media collections, which predominantly contain people.

This paper contributes a new stroke-based rendering (SBR) algorithm for stylizing photographs of faces into portrait paintings. SBR algorithms create paintings by compositing a sequence of curved spline strokes on a 2D canvas. In contrast to SBR algorithms that encode various rendering heuristics to target a particular artistic style [14, 26], our algorithm learns the style of a human artist *by example*. Given a photograph, and an ordered list of strokes (and related attributes) captured from a training session in which an artist paints that photograph, we are able to learn the artist's style and render previously unseen photographs of faces into portraits with a similar aesthetic. To enable our two core contributions; robust learning of styles, and the synthesis of high quality portraits, we harness Computer Vision to parse visual structure from the source image and drive our rendering process.

Our algorithm is aligned with Image Analogies [13] and derivative techniques [17] that learn non-parametric models of image filters from a pair of unfiltered and filtered greyscale images. Such systems are able to learn filters, including edge preserving filters reminiscent of a painterly effect, by sampling pairs of corresponding patches from the two images. The learned filter is applied by looking up patches from the new image. Our approach differs as we train at the level of the stroke, learning how the placement and appearance of each brush stroke is modulated according to underlying visual features in the training image. As such, our approach is specialized to the task of painting, enabling a wide variety of artistic styles. We specialize further to portraits by learning stroke models independently within semantic regions of the face, identified using an Active Shape Model (ASM) and Graph Cut. Image features are composed using a Markov Random Field (MRF) model to ensure spatial coherence of stroke style during learning and rendering. To the best of our knowledge our system is the first to explore portrait stylization by example at the level of the stroke.

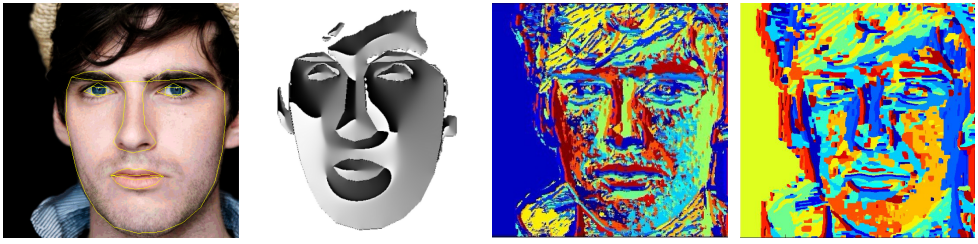
## 1.1 Related Work

The majority of SBR painterly algorithms target generic scenes. Early systems synthesized painterly renderings by incrementally compositing virtual brush strokes whose color, orientation, scale, and ordering were derived from semi- [11] or fully automated processes [14] driven by local image gradient. Later, global methods were proposed to optimize the placement of all strokes — minimizing heuristic functions that encouraged retention of edge detail [15] or perceptually salient visual structures [6]. Mid-level representations for painting based on regions have inspired the composition of higher-level features for artistic rendering. Collomosse and Hall [5] use salient regions as compositional elements to simulate Cubism. Song *et al.* [22] classify regions into one of several canonical shapes to create abstract shape renderings. In contrast to these heuristic approaches, example-based rendering (EBR) algorithms harness machine learning to model the artistic stylization process — typically by densely sampling corresponding patches in a source and stylized training image pair ($A$ and $A'$ respectively). Stylization of a target image $B$ proceeds by matching patches on an approximate nearest neighbor (ANN) basis with those sampled from $A$ during training [13]. The corresponding patch from $A'$ is then composited into stylized version of $B$, to create mapping $B \mapsto B'$ said to be *analogous* to $A \mapsto A'$.

Although some literature exists on portrait sketching [8, 25] and caricatures [10, 24] prior work dedicated to painterly portraits is sparse. DiPaola [9] attempts to map the knowledge domain of the human portrait painter. However, this preliminary work places emphasis on methodology rather than delivering a concrete rendering system. Zhao and Zhu [27] are arguably closest to our work, presenting an "example-based" method to paint portraits. As with our algorithm, strokes are captured during training from a human artist. However to create a new image [27], the training strokes are simply warped from the training face to the new face, using a triangular mesh established over facial features. The system does not learn a model of the painting process, and so can not generalize beyond its training data to produce new paintings. Adopting a warping, rather than rendering, strategy leads to stroke distortion, and cannot emphasize shadow or highlights as we do.

# 2 Feature Extraction and Style Modeling

We synthesize our portraits using a greedy stroke-based rendering (SBR) algorithm described in Sec. 3. As with many spline-based SBR algorithms [7, 12, 14], we derive stroke attributes (e.g. spline shape, thickness, color) using data sampled locally from the source photograph (e.g. orientation, edge strength, color). However, central to our approach is the introduction of *transfer functions* on these values, i.e. between sampling and rendering. These functions are learned by observing a training photograph and corresponding painting (stroke sequence) captured from an artist. For example, when training a particular painting style we might observe a hue shift in the shadow on a cheek, or long thin strokes when painting fine edges local to an eye. When rendering a new portrait, we can introduce similar stylistic effects in stroke



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

Figure 1: Feature extraction, from left to right: (a) Source with ASM fitted; (b) Parsed region map and interpolated orientation field in facial area; Codeword map formed by (c) hard quantization and (d) proposed MRF method.

attributes (e.g. stroke thickness or color) by setting up transfer functions that manipulate the underlying color, edge magnitude or orientation fields that drive the SBR process. These functions are learned local to particular visual structures, within particular semantic regions of the face.

We first describe the process by which we parse and represent structure from photographs of the face (Sec. 2.1). This representation enables us to learn transfer functions local to visual structures within facial features (Sec. 2.2).

## 2.1 Facial Feature Extraction

Portraiture demands a careful composition of facial features (e.g. eyes, mouth) each differing in depictive style. For both training and rendering, we perform segmentation on the input photograph to parse semantic regions corresponding to facial features and so label each pixel. In addition we extract mid- and low-level structure to guide the learning process using texture and luminance information. Each pixel of the image is thus assigned a tuple $\{\mathcal{R}, \mathcal{W}, \mathcal{L}\}$ reflecting the local semantic feature, texture, and luminance. In addition we compute an orientation field $\Theta$ from edges and salient facial features.

### 2.1.1 Semantic Segmentation ($\mathcal{R}$)

We fit an Active Shape Model (ASM) [8] to the input image, comprising landmarks local to the eyes, eyebrows, nose, mouth and outline of the facial region. Polygons connecting these landmarks form the contours of a subset semantic facial regions (Fig. 1(a)) which we use as a basis for deriving a more complete facial representation. The ASM is insufficiently flexible to accurately reflect the shape of each facial feature. We extract a spatial and color prior from fitted ASM regions to to drive a Graph-Cut segmentation local to the bounding box of each feature [2]. This refinement is performed for the mouth, eyes and eyebrows — where precision is particularly important in a portrait. Regions of high shape diversity such as the forehead, ears and neck cannot be represented in the ASM and are addressed using further segmentation. Skin tone is learned from ASM fitted features using a GMM; pixels on image borders train a background GMM. After applying Graph-Cut, pixels classified as foreground exterior to the ASM facial area are labeled as the forehead, ears and neck respectively according to their spatial relationship. The remaining pixels (hair, clothes and background) are treated as one region. Thus the portrait is parsed into 8 regions $r \in \mathcal{R}$.

### 2.1.2 Codebook of Visual Structure ($\mathcal{W}$)

Artists emphasize different types of image feature differently; for example long thin strokes along edges. We wish to learn rather than prescribe such heuristics and densely sample SIFT [18] at each pixel to capture this contextual structure. A dictionary of 20 visual words is obtained via $k$-means over all descriptors sampled [21] (compact codebooks produce spatially

coherent regions). Each descriptor is assigned a unique word $w \in \mathcal{W}$ in the dictionary. The dictionary is also used later when rendering a new image. However descriptor-codeword assignment on a nearest-neighbor basis produces spatially incoherent results (Fig. 1(c)). We apply a Markov Random Field (MRF) model to optimize the labeling $f$ which assigns codeword $w \in \mathcal{W}$ to each pixel with SIFT descriptor $s \in \mathcal{S}$. Let $\tilde{\mathcal{S}} \subseteq \mathcal{S}$ be the set of corresponding SIFT descriptors of codebook. The energy of labeling is given by:

$$E(f) = \sum_{i \in \mathcal{I}} D_i(w_i) + \sum_{i \in \mathcal{I}, j \in \mathcal{N}_i} V_{i,j}(w_i, w_j). \tag{1}$$

where $\mathcal{N}_i$ denotes the set of four-connected neighbors of pixel $i$. $D_i(w_i)$ is the cost of assigning codeword $w_i$ to pixel $i$, with SIFT descriptor $\tilde{s}_i$ and closest local descriptor $s_i$:

$$D_i(w_i) = \min(\mu ||s_i - \tilde{s}_i||, \tau). \tag{2}$$

where constants $\mu$ (0.01) and $\tau$ (100) shape the response of the Euclidean distance function $||.||$. $V_{i,j}(w_i, w_j)$ is the cost of assigning codewords $w_i$ and $w_j$ to neighboring pixels:

$$V_{i,j}(w_i, w_j) = \min(\mu ||s_i - s_j||, \tau). \tag{3}$$

We use max-product belief propagation (BP) [23] to solve for the MRF over a four-connected pixel lattice. Fig. 1(d) illustrates the resulting codeword map, with improved coherence versus nearest-neighbor (Fig. 1(c)).

### 2.1.3 Orientation ($\Theta$)

We construct an edge orientation field $\Theta$ to encode the boundaries of salient features in the face as well as incidental occurrences of edges such as wrinkles in the face. Highly anisotropic regions such as the hair are also emphasized.

We create a sparse edge map $M(x,y) = \{0,1\}$ consisting of the contours of semantic facial regions (from the ASM) and salient edges from the Sobel operator, from which we interpolate an orientation field. Given this edge map, we compute a sparse field from the gradient of edge pixels $\theta[x,y] \mapsto \text{atan}\left(\frac{\delta M}{\delta x} / \frac{\delta M}{\delta y}\right), \forall_{x,y} M(x,y) = 1$. We define a dense orientation field $\Theta_{\Omega^-}$ over all coordinates within the facial region $\Omega^-$, minimizing:

$$\underset{\Theta}{\text{argmin}} \int\int_{\Omega^-} (\bigtriangledown \Theta - \mathbf{v})^2 \quad s.t. \;\; \Theta|_{\delta\Omega^-} = \theta|_{\delta\Omega^-}. \tag{4}$$

i.e. $\triangle\Theta = 0$ over $\Omega^-$ s.t. $\Theta_{\delta\Omega^-} = \theta|_{\delta\Omega^-}$ for which a discrete solution was presented by Perez *et al.* [20] solving Poisson's equation with Dirichlet boundary conditions. $\mathbf{v}$ represents the first order derivative of $\theta$.

### 2.1.4 Intensity ($\mathcal{L}$)

We represent intensity level by quantizing the luminance channel into 8 bins, assigning the bin number $l \in \mathcal{L}$ to each pixel. A similar MRF optimization to eq. 1 enhances spatial coherence, posing quantization as a pixel-labeling problem of assigning each pixel $i \in \mathcal{I}$ with a value from the set of quantization levels, $\mathcal{L}$. Let $\mathcal{V}$ be the set of intensity values in the luminance channel, and $\tilde{\mathcal{V}} \subseteq \mathcal{V}$ be the set of intensity values in edges from hard quantization:

$$E(f) = \sum_{i \in \mathcal{I}} D_i(l_i) + \sum_{i \in \mathcal{I}, j \in \mathcal{N}_i} V_{i,j}(l_i, l_j). \tag{5}$$

$$D_i(l_i) = \min(||v_i - \tilde{v}_i||, \lambda). \tag{6}$$

$$V_{i,j}(w_i, w_j) = \min(||v_i - v_j||, \lambda) + \min(\mu ||s_i - s_j||, \tau). \tag{7}$$

where $D_i(l_i)$ is the cost of assigning quantization level $l_i$ to pixel $i$, which provides the local evidence of the labeling. $V_{i,j}(l_i, l_j)$ is the cost of assigning quantization levels $l_i$ and $l_j$ to two neighboring pixels, enforcing spatial coherence with constants as Sec.2.1.2.

(a)  (b) 

Figure 2: Learning stroke orientation. (a) strokes in the direction of intensity gradient, trained by Fig.5(f). (b) scribbled strokes in Fig.5(c) cause diverse, swirling orientations.

## 2.2 Learning Stroke Style

Our painterly rendering algorithm (Sec. 3) composits a sequence of curved spline brush strokes to create a painting. Each stroke is represented using an Catmull-Rom (interpolating) piecewise cubic spline, comprising $n$ control points $c_{1\ldots n}$. Stroke properties such as geometry (i.e. stroke length and position), as well as stroke thickness and color, are determined using information sampled local to $c_{1\ldots n}$ in the image.

Our training process operates by observing these properties in strokes within manually created training paintings. The system learns the mapping of these properties to pixels derived information within the training photograph (such as color, orientation) local to each stroke's control points $c_{1\ldots n}$. i.e. we observe the stroke property set $\mathcal{P}$ given features $\mathcal{F}$ present in the image local to these points. The mapping is learned by modeling the distribution $p(\mathcal{F}|\mathcal{P})$ independently for each facial region ($\mathcal{R}$). The rendering process then estimates the stroke parameters $\mathcal{P}$ given features $\mathcal{F}$ observed in a new image via: $p(\mathcal{P}|\mathcal{F}) \propto p(\mathcal{F}|\mathcal{P})p(\mathcal{P})$ where we assume all stroke properties are equally likely (uniform prior). $p(\mathcal{F}|\mathcal{P})$ is learned independently for each $\mathcal{W}$ or $\mathcal{W} \times \mathcal{L}$ pair as follows.

### 2.2.1 Color Transfer Model

Particular features or visual structures may cause an artist to shift toward particular shades or hues; for example, complementary pairs of colors are often used by artists to emphasize light and shadow. We learn a color transfer function $\mathcal{F}_c : \{\mathcal{R}, \mathcal{W}, \mathcal{L}\} \mapsto \{\Delta a, \Delta b\}$ for each three-tuple, where $\{\Delta a, \Delta b\}$ indicates the deviation of training stroke color from the training source image in the $a$ and $b$ channels of *CIELab* space. By learning for each three-tuple we sample a color transfer model for various illumination levels of each category of visual words ($\mathcal{W}$) — which are in turn, learned independently for each semantic region ($\mathcal{R}$).

For a given tuple we learn the transfer function as follows. Our system accumulates the color deviation $\{\Delta a_{c_i}, \Delta b_{c_i}\}$ in $a$ and $b$ channels of stroke color at each control point $c_i$ from the underlying image to the tuple entry $\{r_{c_i}, w_{c_i}, l_{c_i}\}$. Color deviation of each tuple entry is averaged after painting to form a $2D$ vector $\{\Delta a_{(r,w,l)}, \Delta b_{(r,w,l)}\}$, which encodes how the painter uses color to account for varying local structures and luminance.

### 2.2.2 Stroke Orientation Model

We model the orientation deviation of strokes using orientation field $\Theta$ (subsec. 2.1.3). Portraits of a specific style exhibit characteristic patterns of stroke orientations local to visual structure. As with color (subsec. 2.2.1) we learn orientation non-parametrically as a transfer function, encoding deviation between $\Theta$ and stroke orientation (Fig. 2).

We compute the local stroke orientation at control point $c_i$ using the coordinates of two consecutive points as $\theta[c_i] \mapsto \text{atan}(c_{i-1} - c_i)$. Thus the deviation of stroke orientation from underlying orientation is computed as $\theta[c_i] - \Theta_\Omega[c_i]$, observations of which are per tuple entry $\{r_{c_i}, w_{c_i}\}$. The mean $\mu$ and standard deviation $\sigma$ of orientation deviations are computed for each tuple entry yielding a Gaussian model $\mathcal{N}^o_{(r,w)}(\mu, \sigma^2)$.

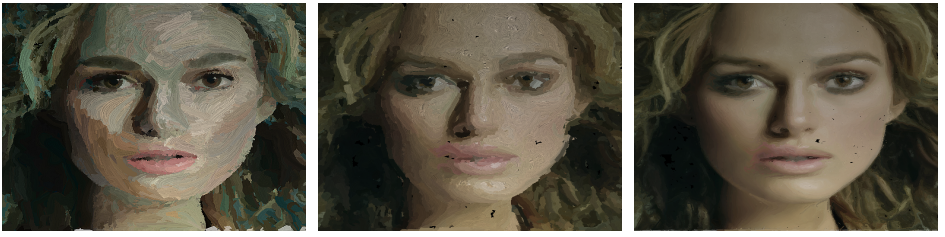


Figure 3: Portrait rendering result of our proposed algorithm (left), result of Hertzmann's multi-layer algorithm with brush radii $R = \{16, 8, 4\}$ (middle) and $R = \{8, 4, 2\}$ (right). Note for all portrait renderings **we recommend zooming in the PDF to 400% to view details** .

### 2.2.3 Stroke Density Model

The density of painted strokes can vary according to the visual salience of depicted features, and their appearance. The control points of training strokes correspond to tuple entries $\{r, w\}$; A count $O_{(r,w)}$ is maintained for each tuple. The count is normalized by area on a per region basis, i.e. over all $\mathcal{W}$ for a given $\mathcal{R}$.

### 2.2.4 Stroke Thickness and Length Models

We accumulate the thickness of strokes, creating a count on each tuple entry $\{r, w\}$ corresponding to the classification of the control point $c_i$. For control point, the count is incremented by the ratio of the stroke thickness to the width of the face, to account for the scale variation of facial area. An identical procedure is undertaken to record stroke length. After the painting is finished, we calculate the mean $\mu$ and standard deviation $\sigma$ of stroke thickness per tuple entry to form a Gaussian model $\mathcal{N}^s_{(r,w)}(\mu, \sigma^2)$. A Gaussian model $\mathcal{N}^l_{(r,w)}(\mu, \sigma^2)$ is similarly learned for stroke length.

## 3 Portrait Rendering

Our algorithm accepts as input a source portrait photograph to render, composite orientation field $\Theta_\Omega$, and learned mappings $\mathcal{F} \mapsto \mathcal{P}$, i.e. the mappings from parsed image features ($\mathcal{F}$) at high- ($\mathcal{R}$), mid- ($\mathcal{W}$) and low- ($\mathcal{L}$) levels to translations in parameter space for Orientation ($\{\mathcal{R} \times \mathcal{W}\} \mapsto \mathcal{N}^o(\mu, \sigma^2)$); Thickness ($\{\mathcal{R} \times \mathcal{W}\} \mapsto \mathcal{N}^s(\mu, \sigma^2)$); Length ($\{\mathcal{R} \times \mathcal{W}\} \mapsto \mathcal{N}^l(\mu, \sigma^2)$); Density ($\{\mathcal{R} \times \mathcal{W}\} \mapsto \mathcal{O}$); and Color ($\{\mathcal{R} \times \mathcal{W} \times \mathcal{L}\} \mapsto \{\Delta a, \Delta b\}$).

Curved strokes are 'grown' bi-directionally from seed locations. Seed positions follow the stroke density model $O_{(\mathcal{R},\mathcal{W})}$ learned in Sec. 2.2.3. Given a point $c_i$ and the associated tuple entry $\{r_{c_i}, w_{c_i}\}$, the probability that a stroke seed is generated at point $c_i$ is $O_{(r_{c_i}, w_{c_i})}$.

Each stroke is grown bi-directionally from a seed control point $c_0$, with two additional control points being placed a short 'hop' distance away — the direction of the hop is determined by vectors with orientation $\theta[c_0]$ and $\theta[c_0] + \pi$, after [14]. Orientation $\theta[c_0]$ is computed based on sampling from the learned orientation model $\mathcal{N}^o_{(r_{c_0}, w_{c_0})}(\mu, \sigma^2)$ trained in Sec. 2.1.3. $\{r_{c_0}, w_{c_0}\}$ is the tuple entry associated with $c_0$ in the parsed face representation. We use a truncated Gaussian $\mathcal{N}^o_{(r_{c_0}, w_{c_0})}(\mu, \sigma^2, a = -\sigma, b = \sigma)$. The maximum length $l_{max}$ of the current stroke initiated from $c_0$ is generated from the truncated Gaussian $\mathcal{N}^l_{(r_0, w_0)}(\mu, \sigma^2, a = -\sigma, b = \sigma)$ learned in Sec.2.2. The thickness is similarly sampled from truncated Gaussian $\mathcal{N}^s_{(r_0, w_0)}(\mu, \sigma^2, a = -\sigma, b = \sigma)$. Note the orientation of each stroke fragment is updated on each new control point $c_i$ following $\mathcal{N}^o_{(r_{c_i}, w_{c_i})}(\mu, \sigma^2, a = -\sigma, b = \sigma)$ whilst the maximum length and size of the stroke are fixed by the initial control point.

Figure 4: Comparisons. Top: portraits of source image (left) using hard quantized (middle) features, and our MRF approach (right). Bottom: our algorithm (left) and [27] (middle and right) which warps pre-painted strokes to new faces.

The stroke grows from the initial seed $c_0$ to point $c_{-1}$ and $c_1$ along $\theta[c_0]$ and $\theta[c_0]+\pi$ respectively with a minimum length of $L_{min}$ (2 pixels in our system); this process iterates until any of the following criteria are violated. Growth halts if the curvature change between a pair of consecutive stroke fragments is larger than a threshold ($\frac{pi}{2}$), or the new control point belongs to a different semantic region than $c_0$. Regions are painted independently in order of area (largest first) and textured to enhance their painted appearance [16].

## 3.1 Color Transfer

After all the control points of a new stroke are generated, the associated code word $\tilde{w}_{c_0}$ with the highest occurrence, and thus the tuple entry $(r_{c_0}, \tilde{w}_{c_0})$ can be found to impose a color transform on the stroke. As in the training process of Sec. 2.2, we quantize the averaged $L$ channel of all the control points in *CIELab* space as $\tilde{l}_{c_0}$. All together, we identify the tuple entry $\{r_{c_0}, \tilde{w}_{c_0}, \tilde{l}_{c_0}\}$ to index the color deviation model learned from Sec. 2.2.1.

Given the associated tuple entry $\{r_{c_0}, \tilde{w}_{c_0}, \tilde{l}_{c_0}\}$, the color deviation model of the current stroke is a $2D$ vector $\{\Delta a_{(r_{c_0}, \tilde{w}_{c_0}, \tilde{l}_{c_0})}, \Delta b_{(r_{c_0}, \tilde{w}_{c_0}, \tilde{l}_{c_0})}\}$. Let $\bar{L}_{c_0}$, $\bar{a}_{c_0}$, and $\bar{b}_{c_0}$ be the average *Lab* channels over the control points respectively, the color $C_s : \{\tilde{L}_{c_0}, \tilde{a}_{c_0}, \tilde{b}_{c_0}\}$ of the stroke which originates from $c_0$ is: $\tilde{L}_{c_0} \leftarrow \bar{L}_{c_0}$, $\tilde{a}_{c_0} \leftarrow \bar{a}_{c_0} + \Delta a_{(r_{c_0}, \tilde{w}_{c_0}, \tilde{l}_{c_0})}$, $\tilde{b}_{c_0} \leftarrow \bar{b}_{c_0} + \Delta b_{(r_{c_0}, \tilde{w}_{c_0}, \tilde{l}_{c_0})}$.

## 3.2 Null Models

The sparseness of the training data can result in no model being recorded for particular tuple (i.e. a null model). Tuples coded to null models may be encountered during rendering, and it is necessary to perform a lookup to identify the closest tuple with a model. This also promotes spatially coherent appearance. For color transfer we must find the nearest tuple in $\{\mathcal{R} \times \mathcal{W} \times \mathcal{L}\}$ space; for the remainder of the properties in $\{\mathcal{R} \times \mathcal{W}\}$ space.

# 4 Results and Discussion

We evaluate the proposed system, demonstrating improved preservation of salient features over the state of the art [14, 27] and the broad gamut of styles achievable. Our system is trained using a single photograph/painting pair. Brush strokes are captured using a bespoke UI resembling a basic Photoshop environment with stroke color and size selection. Strokes are painted on top of the original training photograph to provide a point of reference.

## 4.1 Comparison with Baseline Methods

We first compare our rendering algorithm with the general purpose painterly algorithm of Hertzmann [14], commonly used as a baseline for painterly stylization. Fig. 3 provides a visual comparison, showing that this generic painting method either destroys important facial details through blurring or over-paint Fig. 3 (middle) or produces photorealistic renderings without insufficient painterly effect Fig. 3 (right) that tend back towards photorealism.

Fig. 4 (bottom) compares our results with Zhao and Zhu [27] where artists' strokes are reproduced 'verbatim' and warped to fit the face. This stroke-warping approach relies upon on a global facial model but, as the stroke positions are prescribed rather than algorithmically generated, they are unable to depict visual structures such as shadows or wrinkles. Detail in regions such as the eyes distinguishes our system's capability to adapt to fine facial features. The phenomenon of "ghost teeth"[27] on the lips in Fig. 4 (bottom, right) is similarly caused by re-using captured strokes rather generating them. By contrast the result from our system (learned from 5e, with teeth) successfully depicts detail in the eyes but also deals with changes in geometry, transferring the smiling face with teeth to faces with no teeth showing. Note [27] also does not provide a solution to render hair, as we do, and paints this as a post-process. Source images for [27] are neither available for direct comparison, nor is an author implementation available.

## 4.2 Style and Coherence

We evaluate the benefit of our MRF based feature composition which enforces spatial coherence in codeword ($\mathcal{W}$) and intensity ($\mathcal{L}$). Optimizing these fields to enforce spatial coherence directly influences the coherence of $\mathcal{P}$ and so aesthetics, versus naive nearest neighbor approaches to quantization (Fig. 4, top).

In order to evaluate style learning we asked 10 participants with levels of artistic experience from professional to hobbyist, to paint portraits using our UI (Fig. 5, a-f). The corresponding synthesized renderings are in Fig. 5, k-l. Figs. 5(a)-5(b) and Figs. 5(g)-5(h) show a couple of training examples and rendering results respectively demonstrating correct learning and reproduction of color shading. In Fig.5(a)/Fig.5(g) a warm color is palette is learned, with different reds and pinks being used to emphasize light and shadow in different ways for different facial features. Complementary color shifts are learned in Fig.5(b)/Fig.5(h), where hints of orange and purple are used to emphasize light and shadow. Only shadows with similar visual structure to the training image are painted in complementary color; most evident local to the cheeks and eyes. In Fig. 5(c)/5(i), thick, short strokes are laid down as color blobs causing a circular stroke and coarse abstraction. Fig. 5(d)/5(j) uses medium thick strokes to depict fine details of portrait but a more randomized orientation within flat regions e.g. forehead. Training Fig. 5(e) causes thick strokes with medium length to increase the abstraction level around the hair in Fig. 5(k) whilst retaining finer strokes to depict the facial region of portrait. Fig. 5(f) trains hair texture using long strokes which results in corresponding long, sweeping strokes in the hair region of Fig. 5(l).

Fig. 6 shows results of re-rendering training source Figs.5(e)-5(k) using the style models learned from those training examples. This demonstrates a painting used to train a system may be approximately reproduced from the learned style parameters. We are not trying to reconstruct the training painting exactly, which is only possible by simply warping the training strokes as Zhao and Zhu [27] did — rather we aim to reconstruct the visual style

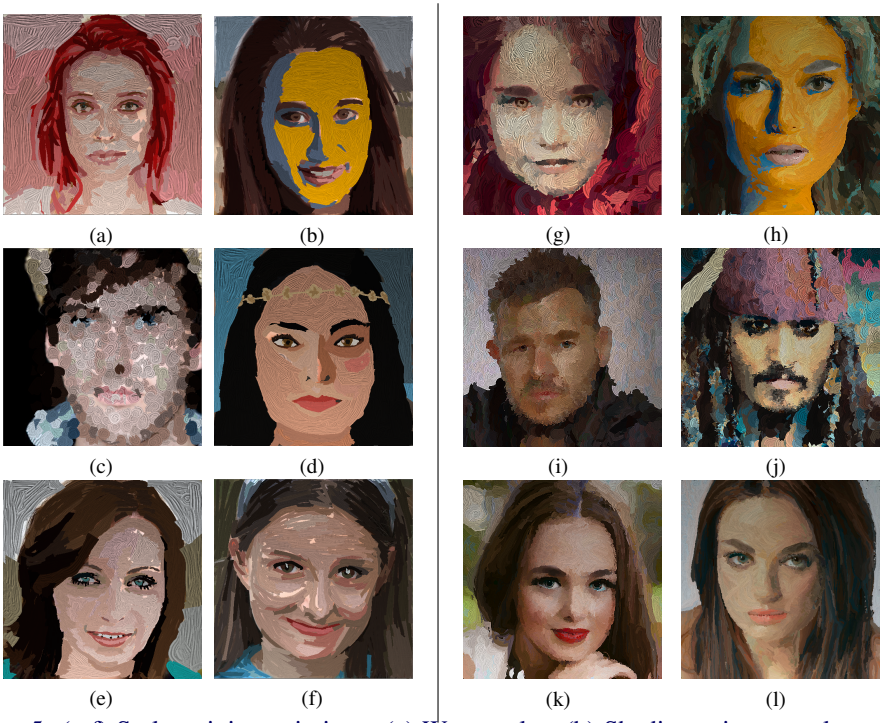



Figure 5: (a-f) Style training paintings: (a) Warm color; (b) Shading using complementary color; (c) Blobby strokes; (d) Thick, haphazard strokes; (e) Natural color, medium thick strokes; (f) Long strokes for hair. (g-l) Renderings using models learned from (a-f).

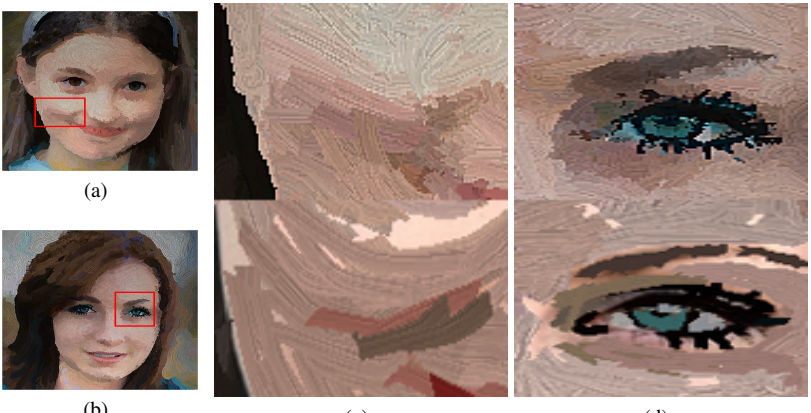



Figure 6: Portrait renderings of test images using models learned from the same images (a) Fig. 5(f); (b) Fig. 5(e); (c) Closeup on (a) and corresponding section in training painting; (d) Closeup on (b) and the corresponding section in training painting.

of the training image. The rendering results exhibit similar styles, e.g. color tones, stroke orientation, thickness, and length, with the corresponding training paintings shown in Figs. 5(f) and 5(e). For example, the horizontal strokes on the the girl's cheek (Fig. 6(c)) in Fig. 6(a) have similar shading in Fig. 5(f); the style to depict the shade areas of hair and skin, even the eyelash (Fig. 6(d)) in Fig. 6(b), echo the corresponding training painting Fig. 5(e).

# 5 Conclusion

We have presented a user trainable algorithm for stylizing photographs into portrait paintings. We learn a flexible non-parametric model of artistic style by analyzing the global and local geometry as well as tone of brush strokes placed local to image features. Portraiture has previously proven challenging for painterly rendering. We are able to depict faces without the loss of salient detail exhibited by more general painterly methods [12] and without relying on a pre-painted arrangement of strokes to warp over the face [27]. In contrast to warping pre-capture stroke maps, we algorithmically place strokes as a function of image content presenting the first portrait painting algorithm to learn stroke style parameters by example. In the future we would consider parsing using 3D morphable models [1] to provide depth cues that might capture additional style variation with geometry.

# References


[1] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Proc. SIGGRAPH*, pages 187–194, 1999.

[2] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *Proc. ICCV*, pages 105–112. IEEE, 2001.

[3] H. Chen, Z. Liu, C. Rose, Y. Xu, H.-Y. Shum, and D. Salesin. Example-based composite sketching of human portraits. In *Proc. NPAR*, pages 95–153, 2004.

[4] J Collomosse and P Hall. Salience-adaptive painterly rendering using genetic search. *Intl. Journal on Artificial Intelligence Tools (IJAIT)*, 4(15):551–576, August 2006.

[5] J. Collomosse and P. M. Hall. Cubist style rendering from photographs. *IEEE Trans. Vis. Comput. Graphics*, 4(9):443–453, 2003.

[6] J. Collomosse and P. M. Hall. Genetic paint: A search for salient paintings. In *Proc. EvoMUSART*, pages 437–447, 2005.

[7] J. Collomosse, D. Rowntree, and P. M. Hall. Stroke surfaces: Temporally coherent artistic animations from video. *IEEE TVCG*, 11(5):540–549, 2005.

[8] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models - their training and application. *Comput. Vis. Image Underst.*, 61:38–59, January 1995.

[9] Steve Dipaola. Painterly rendered portraits from photographs using a knowledgebased approach. In *In Proc SPIE Human Vision and Imaging*, 2007.

[10] Bruce Gooch, Erik Reinhard, and Amy Gooch. Human facial illustrations: Creation and psychophysical evaluation. *ACM Trans. Graph.*, 23(1):27–44, 2004.

[11] Paul Haeberli. Paint by numbers: Abstract image representations. In *Proc. SIGGRAPH*, pages 207–214, 1990.

[12] J. Hays and I. Essa. Image and video based painterly animation. In *Proc. ACM NPAR*, pages 113–120, 2004.

[13] A. Hertzmann, C. Jacobs, N. Oliver, B. Curless, and D. Salesin. Image analogies. In *Proc. ACM SIGGRAPH*, pages 327–340, 2001.

[14] Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *Proc. SIGGRAPH*, pages 453–460, 1998.

[15] Aaron Hertzmann. Paint by relaxation. In *Proc. CGI*, pages 47–54, 2001.

[16] Aaron Hertzmann. Fast paint texture. In *Proc. NPAR*, pages 91–96, 2002. doi: 10.1145/508530.508546.

[17] Hochang Lee, Sanghyun Seo, Seungtaek Ryoo, and Kyunghyun Yoon. Directional texture transfer. In *Proc. NPAR*, pages 43–50, 2010. ISBN 9781450301251. doi: 10.1145/1809939.1809945.

[18] David G. Lowe. Object recognition from local scale-invariant features. In *Proc. ICCV*, pages 1150–1157, Washington, DC, USA, 1999. IEEE. ISBN 0-7695-0164-8.

[19] E. McKone, N. Kanwisher, and C. Duchaine. Can generic expertise explain special processing for faces? *Trends in Cognitive Science*, 11:8–15, 2007.

[20] P. Perez, M. Gangnet, and A. Blake. Poisson image editing. In *SIGGRAPH*, pages 313–318, 2003.

[21] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation. *IJCV*, 77:157–173, May 2008.

[22] Y. Song, P. M. Hall, P. Rosin, and J. Collomosse. Arty shapes. In *Proc. CAe*, pages 65–72, 2008.

[23] Marshall F. Tappen and William T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In *ICCV*, pages 900–907, 2003.

[24] M. Tominaga, S. Fukuoka, K. Murakami, and H. Koshimizu. Facial caricaturing with motion caricaturing in PICASSO system. In *Proc. ICAIM*, page 30, 1997.

[25] Xiaogang Wang and Xiaoou Tang. Face photo-sketch synthesis and recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(11):1955–1967, 2009.

[26] Kun Zeng, Mingtian Zhao, Caiming Xiong, and Song-Chun Zhu. From image parsing to painterly rendering. *ACM Trans. Graph.*, 29(1):2:1–11, 2009.

[27] Mingtian Zhao and Song-Chun Zhu. Portrait painting using active templates. In *Proc. ACM NPAR*, pages 117–124, 2011.