Cubist Style Rendering from Photographs

J. P. Collomosse and P. M. Hall

Abstract— ***The contribution of this paper is a novel non-photorealistic rendering (NPR) technique, influenced by the style of Cubist art. Specifically we are motivated by artists such as Picasso and Braque, who produced art work by composing elements of a scene taken from multiple points of view; paradoxically such compositions convey a sense of motion without assuming temporal dependence between views. Our method accepts a set of two-dimensional images as input, and produces a Cubist style painting with minimal user interaction. We use salient features identified within the image set, such as eyes, noses and mouths as compositional elements; we believe the use of such features to be a unique contribution to NPR. Before composing features into a final image we geometrically distort them to produce the more angular forms common in Cubist art. Finally we render the composition to give a painterly effect, using an automatic algorithm. This paper describes our method, illustrating the application of our algorithm with a gallery of images. We conclude with a critical appraisal and suggest the use of "high-level" features is of interest to NPR.***

Keywords— ***Non-photorealistic rendering, Cubism, Salient features.***

I. Introduction

We present a novel method of non-photorealistic rendering (NPR), which draws upon techniques from both the graphics and the vision communities to produce stylised compositions *reminiscent* of Cubist art. We are influenced by artists such as Picasso and Braque, who produced art work by composing elements of a scene taken from multiple points of view. Paradoxically the Cubist style conveys a sense of motion without assuming temporal dependence between views.

Our motivation is to render motion from two-dimensional (2D) images, such as photographs. The Cubist approach is especially interesting, as it requires the identification of salient features within images to produce an abstract composition. By *salient feature* we mean an image region containing an object of interest, such as an eye or nose; a composition made from elements of low salience would tend to be uninteresting. To the best of our knowledge, "high level" salient features have not been used previously to produce non-photorealistic renderings. An attempt to emulate Cubist art provides a useful contribution to NPR, not only because it produces pleasing renderings, but also because it forces us to study salience, with potentially wider benefits.

We therefore set out to investigate whether aesthetically pleasing art, reminiscent of the Cubist style, could be synthesised. Specifically we were interested in achieving a suitably high degree of automation. We considered the following specific questions:

J. P. Collomosse and P. M. Hall are with the Department of Computer Science, University of Bath, Bath, England.
Email: {jpc | pmh}@cs.bath.ac.uk

- How is salience to be defined so that it operates over a wide class of input images?
- How should salient features be selected from amongst many images, and how should the selected features be composed into a single image?
- How should the angular geometry common in Cubist art be reproduced?
- How should the final composition be rendered to produce a painted appearance?

Resolution of the first two questions provides the basic mechanism by which a Cubist-like image can be formed; resolution of latter two questions enhances aesthetic quality.

Our method accepts one or more 2D images as input. Salient features within each image are then identified using statistical analysis. This can produce disconnected features, which requires correction; in our case by minimal user interaction. These features are geometrically distorted. A subset is then selected and composited, ensuring that non-selected features do not inadvertently appear in the final composition – naïve composition allows this to happen. The composition process is stochastic, and a new image may be produced on each new run of the method. In rendering a painting from the final composition our painting algorithm treats brush strokes in a novel way, so that salient features are not obscured.

The remainder of the paper is organised as follows. In the next section we explain our research in the context of related literature. We then give a detailed explanation of our algorithm in section III, illustrating our approach with a gallery of images in section IV. We conclude the paper in section V with a critical appraisal, and discuss the direction of future work.

II. Related Work

Non-photorealistic rendering (NPR) is an area of increasing interest within the field of computer graphics, and many techniques have been developed to render both 2D and 3D source data for the purposes of both aesthetics and visualisation. We restrict our attention to the subset of methods developed to synthesise drawing and painting techniques, which we refer to as "artificial drawing" [1].

Many artificial drawing techniques exist, capable of emulating a wide variety of artistic media such as 'pen-and-ink', charcoal, and paint. We briefly summarise the field in order to illustrate the context of our work, and direct the

reader to a number of more detailed literature surveys [2], [3], including a SIGGRAPH course [4], and a more recent text [5].

The majority of early artificial drawing techniques concentrate upon the simulation of traditional artistic media. Pen-and-ink is perhaps the most thoroughly researched, and a number of methods have been proposed that allow the interactive rendering of surfaces through hatching [6], [7], [8], texturing [9], and stippling [10]. Techniques for the interactive illustration of both 3D surfaces [2], [11], [12] and 2D canvas [13], [14] have been developed; other systems allow the virtual engraving and sculpture of more exotic media such as wood [15]. Physical simulations of sticky paint [16], watercolour washes and glazes [17], and a variety of artists brushes [14], [18], [19] have been used to create impressive artwork in interactive digital environments. Whilst painting is of interest to us when rendering the final composition, we are more concerned with the formation of novel images through composition of features.

A multitude of methods have more recently been developed to automate the placement of hatches [20], [21] and strokes [22], [23], [24], [25]. These contrast with earlier work such as that of Haeberli [13], whose paint systems allowed a user to, for example, interactively generate impressionist style "paintings" through a labour intensive process. In a similar fashion, sketch rendering systems were initially developed as interactive tools [7] but were later advanced toward greater automation [20], [21]; some methods are now capable of rendering in real-time [26], [27]. Many techniques address automation through implementation of simplified artistic heuristics, basing decisions upon image processing operations. Most of such systems are guided by local edge information [23], [24], or local statistical measures such as variance [25]. In some systems interaction is augmented by automation [9], [19]. Our method also relies on a combination of user interaction and vision techniques to assist the rendering process, but bases decisions upon the *global* 'salience', or importance, of image pixels as well as using gradient criteria.

The techniques described all share the similarity that they are concerned with emulating the use of artistic media such as canvas or brush, to create a natural or 'hand-painted' effect; in an image processing sense, they operate at a low level. We argue that a higher level of abstraction is necessary to synthesise the compositions common in Cubism. In our research we address the issue of media, but distinguish between the synthesis of a Cubist-style composition and the rendering of that composition. We concentrate upon the emulation of a style rather than an artistic medium, using the high level salient features within an image. In particular, we are concerned with the construction of novel image through composition of high-level features, rather than the rendering of an existing photograph with, for example, painted strokes.

Whilst there has been significant development of artificial drawing methods, the majority of these are concerned with rendering static scenes. Litwinowicz [23], and later Hertzmann [28], extended Haeberli's work [13] to produce temporally coherent animations – treating each frame as a separate rendering, but preserving stroke information to enhance temporal coherence. Our aim is different; to render motion in a single two-dimensional image. Strohotte approaches this issue using streak-lines and similar marks [7]. We approach the problem from the point of view of Cubism; using angular forms and multiple viewpoint composition to convey a sense of time and motion in our work [29].

The portrayal of motion in an abstract manner is currently under-studied. An empirical study [30] suggests that superimposing several frames of animation at specific intervals can affect perception of time. A set of interactive tools [31] recently proposed by Klein *et al* allows a user to paint with 'stroke-solids' to produce NPR video; this interactive system can yield output ostensibly similar to earlier stages of our work, but differs greatly in both level of interaction and abstraction (we construct compositions using identified high-level features rather that allowing a user to interactively subdivide blocks of video). The literature remains sparse concerning techniques for two-dimensional rendering of motion, and the synthesis of abstract artistic styles.

III. Cubist style images from photographs

In this section we present our rendering algorithm. Our method accepts a set of 2D source images, and produces a single 2D image rendered in the Cubist style.

We initially register all source images so that objects of interest, such as faces, fall upon one another when we composit. We threshold upon colour to partition foreground from background, and translate images so that first moments of foreground are coincident. Finally we clip the images to a uniform width and height. This step creates correspondence between source images on a one-to-one basis: pixels at the same (i, j) location in any image correspond. The remaining algorithm stages are of greater interest, and we describe each of them in turn in the following subsections.

A. Identifying salient features

We wish to find a set of salient features amongst the registered images. These images should be unrestricted in terms of their subject (for example, a face or guitar). In addition, we want our salient features to be relatively "high-level", that is they correspond to recognisable objects, such as noses or eyes. This implies we need a definition of salience that is both general and powerful; such a definition does not currently exist in the computer vision literature, or elsewhere. However, we can make progress by choosing a definition of salience that is sufficiently general

for our needs, and allow user interaction to provide power where it is needed.

We locate salient features within a single image by modifying a technique due to Walker *et al* [32], who observe that salient pixels are uncommon in an image. The basic technique is to model the statistical distribution of a set of measures associated with each pixel, and to isolate the outliers of this distribution. The pixels corresponding to these outliers are regarded as salient.

To compute these measures, $\vec{x}$, over each pixel we convolve the image with a set of origin-centred 2D Gaussian derivative filters. Specifically we use 5 first and second order directional derivatives: $\partial G(x,y;\sigma)/\partial x$, $\partial G(x,y;\sigma)/\partial y$, $\partial^2 G(x,y;\sigma)/\partial x^2$, $\partial^2 G(x,y;\sigma)/\partial y^2$, and $\partial^2 G(x,y;\sigma)/\partial x \partial y$. These filters smooth the image before computing the derivative; they respond well to edge and other signals of characteristic scale σ, but as Figure 1 shows, our method is more general than edge detection. We filter using octave intervals of σ, as such intervals contain approximately equal spectral power. In our implementation we use σ values of 1, 2, 4 and 8; thus with each pixel we associate a vector $\vec{x}$ of $20 = 5 \times 4$ components. For an image of M pixels we will have M vectors $\vec{x} \in \Re^n$, where for us $n = 20$. We assume these points are Gaussian distributed, which we represent using an eigenmodel [33]; a simple and convenient model that works acceptably well in practice. The eigenmodel provides a sample mean μ; a set of eigenvectors each a column in orthonormal matrix $\mathbf{U}$; each eigenvector has a corresponding eigenvalue along the diagonal of Λ. An eigenmodel allows us to compute the squared Mahalanobis distance of any point $\vec{x} \in \Re^n$:

$$d^2(\vec{x}) = (\vec{x} - \mu)^T U \Lambda U^T (\vec{x} - \mu) \quad (1)$$

The Mahalanobis distance provides a convenient way of deciding which sample points are salient; we use a simple threshold, $d^2(\vec{x}) > 9$, empirically chosen to give reasonable results. Notice that because we look for statistical outliers we can record pixels in flat regions as being salient, if such regions are rare; a more general method than using variance magnitude [31].

In practice salient pixels tend to form spatially coherent clusters, which tend to be associated with interesting objects in the image, including conceptually high level features such as eyes (Figure 1). However, our method is general purpose, and therefore has no specific model of eyes, or indeed or any other high level feature. It is therefore not surprising that what a human regards as a salient feature may be represented by a set of disconnected salient clusters.

Given that the general segmentation problem, including perceptual grouping, remains unsolved we have two choices: either to specialise the detection of salient regions to specific classes of source images, such as faces or houses; or to allow the user to group the clusters into features. We adopt the latter approach for its power and simplicity: powerful

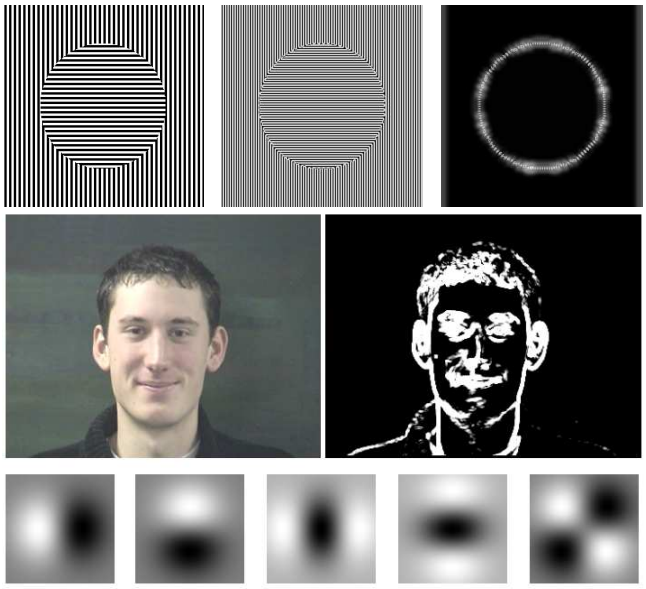

Fig. 1. Top: An illusory circle, edge detected, and salience mapped; showing salience mapping can "pick out" the circle where edge detection fails. Middle: a typical source image (left) and corresponding salience map (right); lighter pixels are more salient. Below: the 5 differential convolution kernels, at one σ value, used to create measures $\vec{x}$.

because we rely on human vision, and simple not only to implement but also to use. We allow the user to point to clusters in order to group them, and compute the convex hull [34] of each group. We identify a *salient feature* as all pixels within the convex hull (each cluster is included in at most one feature, and a cluster need not be included in any feature). This mode of interaction is much simpler for the user than having to identify salient features from images *ab initio*; that is with no computer assistance. Feature grouping is also likely to be consistent between source images, because our salience measure provides an objective foundation to the grouping.

The union of the salient features identified in each source image forms the set of salient features we require, which we call $\mathcal{F}$. In addition to grouping clusters into features, the user may also label the features. These labels partition the set of all salient features $\mathcal{F}$ into equivalence classes, such as "eyes", providing a useful degree of high-level information (these classes represent a simple model of the object in the picture). We make use of $\mathcal{F}$, and associated equivalence classes, throughout the remaining three stages of our algorithm.

B. Geometric Distortion

We now wish to distort the identified features, in $\mathcal{F}$, to produce the more angular forms common in Cubist art. Our approach is to construct a continuous vector field $\mathcal{V}$ over each source image, which is a sum of the contributions made by distorting the set of all features $f \subseteq \mathcal{F}$ belonging to that image. That is, we define a vector-valued distortion function $\vec{g} : \Re^2 \mapsto \Re^2$, so that for every point $\mathbf{u} \in \Re^2$, we

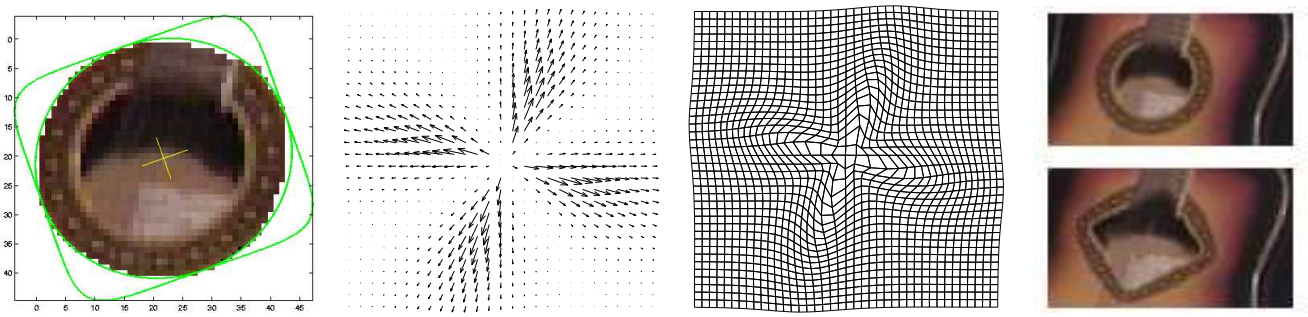

Fig. 2. Four stages of a geometric warp where $\alpha' = 0.3$. From left to right: (a) the source and target superquadrics, fitted about a salient feature; (b) the continuous forward vector field; (c) the mesh of quadrilaterals (mapped pixel areas); (d) the final distorted image.

have $\vec{g}(\mathbf{u}) = \mathbf{u} + \mathcal{V}(\mathbf{u})$ where

$$\mathcal{V}(\mathbf{u}) = \sum_{\phi \in f} \vec{d}_\phi(\mathbf{u}) \tag{2}$$

To define a particular distortion function $\vec{d}_\phi(.)$ we fit a superquadric about the perimeter of feature ϕ, then transform that fitted superquadric to another of differing order; thus specifying a distortion vector field $\vec{d}_\phi(\Re^2)$. We now describe the details of this process.

The superquadric class of functions may be represented in Cartesian form by the equation

$$\left(\frac{x}{a}\right)^{\frac{2}{\alpha}} + \left(\frac{y}{b}\right)^{\frac{2}{\alpha}} = r^{\frac{2}{\alpha}} \tag{3}$$

where a and b are normalised constants ($a + b = 1$) which influence the horizontal and vertical extent of the superquadric respectively, and r is an overall scaling factor. We observe that (3) reduces to the general equation for a closed elliptic curve when $\alpha = 1$, toward a rectangular form as $\alpha \to 0$, and toward a four-pointed star as $\alpha \to \infty$.

We use a parametric form of (3) determined by an angle θ about the origin, by which we correlate points on the perimeter of one superquadric with those on another.

$$x = \frac{r\cos\theta}{\left(\left|\cos\theta/a\right|^{\frac{2}{\alpha}} + \left|\sin\theta/b\right|^{\frac{2}{\alpha}}\right)^{\frac{\alpha}{2}}} \tag{4}$$

$$y = \frac{r\sin\theta}{\left(\left|\cos\theta/a\right|^{\frac{2}{\alpha}} + \left|\sin\theta/b\right|^{\frac{2}{\alpha}}\right)^{\frac{\alpha}{2}}} \tag{5}$$

We calculate the distortion for a given feature by fitting a general superquadric of order α, and warping it to a target superquadric of new order α'. Features whose forms differ from this target superquadric are therefore distorted to a greater degree than features that already approximate its shape; thus each feature boundary converges toward the geometric form specified by α'. Typically we choose $\alpha' < 1$ to accentuate curves into sharp angles. The initial superquadric is fitted about the bounding pixels of the feature using a 6-dimensional optimisation technique described by Rosin [35]. We find this method suitable for our purpose due to its relatively high tolerance to noise.

Recall the distortion function $d_\phi(.)$; we wish to produce a displacement vector $\mathbf{v}$ for a given point $\mathbf{u} = (\mathbf{u}_x, \mathbf{u}_y)$. We first calculate the points of intersection of line $\vec{\mathbf{Ou}}$ and the two superquadric curves specified by α and α', where $\mathbf{O}$ is the origin of both superquadrics (these origins are coincident). We derive the intersections by substituting a value for $\theta = atan(\mathbf{u}_y/\mathbf{u}_x)$ into equations (4) and (5). We denote these intersection points by β and β' respectively (see Figure 3). The vector $\beta' - \beta$ describes the maximum distortion in direction θ. We scale this vector by passing the distance (in superquadric radii) of point $\mathbf{u}$ from the origin, through a non-linear transfer function $\mathcal{T}(.)$. So, for a single feature ϕ:

$$d_\phi(\mathbf{u}) = \mathcal{T}\left(\frac{|\mathbf{u} - \mathbf{O}|}{|\beta - \mathbf{O}|}\right)(\beta' - \beta) \tag{6}$$

The ideal characteristics of $\mathcal{T}(x)$ are a rapid approach to unity as $x \to 1$, and a slow convergence to zero as $x \to \infty$. The rise from zero at the origin to unity at the superquadric boundary maintains internal continuity, ensuring a topologically smooth mapping within the superquadric. Convergence to zero beyond unit radius mitigates against noticeable distortion to surrounding areas of the image that do not constitute part of the feature. The Poisson distribution function (equation 7) is a suitable $\mathcal{T}$, where $\Gamma(.)$ is the gamma function [36] and λ is a scaling constant.

$$\mathcal{T}(x) = \frac{\lambda^{x\mathrm{e}^\lambda}}{\Gamma(x)} \tag{7}$$

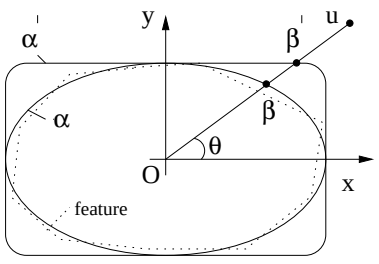

Fig. 3. The fitted and target superquadrics, described by α and α' respectively. Intersection with line $\vec{Ou}$ is calculated using angle θ.

Recall from equation (2) that we sum the individual vector fields of each feature belonging to a specific source image, to construct the overall vector field for that image. With this field defined, we sample those points corresponding to the corners of every pixel in the source image, and so generate their new locations in a target image. This results in a mesh of quadrilaterals, such as that in Figure 2*c*. Mapping each pixel area from the original bounded quadrilateral to the target bounded quadrilateral yields the distorted image.

The distortion process is repeated for each source image, to produce a set of distorted images. At this stage we also warp the bounding polygon vertices of each feature, so that we can identify the distorted salient features $\mathcal{F}'$. For reasons of artistic acumen, we may wish to use different values of α' for each equivalence class (as mentioned in Section I). For example, we may wish to make eyes appear more angular, but leave ears to be rather more rounded.

We draw attention to issues relating to the implementation of our method; specifically that the feature distortion stage can be relatively expensive to compute. This bottleneck can be reduced by: (a) precomputing the transfer function $\mathcal{T}(.)$ at suitably small discrete intervals, and interpolating between these at run-time; (b) using a fast but less accurate method of integrating distorted pixel areas such as bilinear interpolation. In both cases we observe that the spatial quantisation induced later by the painterly rendering stage mitigates against any artifacts that may result.

C. Generation of Composition

We now describe the process by which the distorted salient features are selected from $\mathcal{F}'$ and composited into a target image. Specifically we wish to produce a composition in which:

- Salient features are distributed evenly, and in similar proportion to the equivalence classes of the original image set.
- Features do not overlap each other.
- The space between selected salient features is filled with some suitable non-salient texture.
- Non-salient regions are "broken up" adding interest to the composition, but without imposing structure that might divert the viewer's gaze from salient regions.

A subset of the distorted salient features $\mathcal{F}'$ are first selected via a stochastic process. These chosen features are then composited, and an intermediary composition produced by colouring uncomposited pixels with some suitable non-salient texture. Large, non-salient regions are then fragmented to produce the final composition.

C.1 Selection and Composition

We first describe the process by which a subset of distorted salient features in $\mathcal{F}'$ are selected. We begin by associating a scalar with every feature; specifically the area of the feature weighted by the fractional size of the equivalence class to which it belongs. We treat each scalar as an interval, and concatenate intervals to form a range. This range is then normalised to span the unit interval. We choose a random number from a uniform distribution over $[0, 1]$, which falls in a particular interval, and hence identifies the corresponding feature. Features of larger area from larger equivalence classes tend to be selected in preference to others, which is a desirable bias in our stochastic process. The selected feature is removed from further consideration, and included in a set $\mathcal{C}$, which is initially empty.

Fig. 4. Features selected from the set $\mathcal{F}'$ via the stochastic process of Section III-C.1 (outlined in yellow). Notice that the number of facial parts has a natural balance despite our method having no model of faces; yet we still allow two mouths to be included. Pixels not yet composited are later coloured with some suitable non-salient texture.

This selected feature may intersect features in other images, by which we mean at a least one pixel (i, j) in the selected feature may also be in some other feature in some other image (recall the registration process aligns pixels to correspond on a one-to-one basis). Any features that intersect the selected feature are also removed from further consideration, but are not placed in the set $\mathcal{C}$.

This process is biased toward producing a set $\mathcal{C}$ containing features whose equivalence classes are similar in proportion to the original source images. For example, if the original subject has two eyes and a nose, the algorithm will be biased toward producing a composition also containing two eyes and a nose, but deviation is possible, see Figure 4.

The second step of our process is concerned with the composition of the chosen features in $\mathcal{C}$ to produce the final image. We begin this step by copying all chosen features into a target plane, producing a result such as Figure 4. In order to complete the composition we must determine which image pixels have not yet been composited, and colour them with some suitable non-salient texture.

An initial approach might be to compute a distance transform [37] for each non-composited pixel, which determines its distance to the nearest feature. The corresponding pixel in the distorted source image containing this nearest feature is used to colour the uncomposited pixel. This produces similar results to a Voronoi diagram, except that we seed each Voronoi segment with a region rather than a point. Unfortunately this initial approach is unsatisfac-

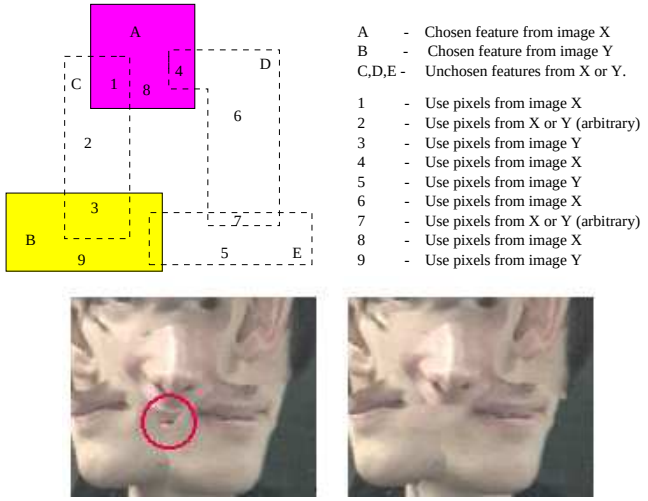

Fig. 5. (a) potential intersections between features (top); (b) final compositions without (left) and with (right) the second stage of processing.

tory: under some circumstances regions may be partially textured by unchosen salient features, and images such as Figure 5*b* may result. To mitigate against partial mouths and similar unappealing artifacts requires greater sophistication, which we introduce by performing a second set of intersection tests.

We copy each of the unchosen features onto the image plane, and test for intersection with each of the chosen features $\mathcal{C}$. If an unchosen feature u intersects with a chosen feature c, we say that *'c holds influence over u'*. Unchosen features can not hold influence over other features. By examining all features, we build a matrix detailing which features hold influence over each other. If an unchosen feature u is influenced by exactly one chosen feature c, we extend feature c to cover that influenced area. We fill this area by copying pixels from corresponding positions in the distorted image from which c originates. Where an unchosen feature is influenced by several chosen features, we arbitrarily choose one of these chosen features to extend over the unchosen one (Figure 5*a*, region 2). However, we do not encroach upon other chosen regions to do this – and it may be necessary to subdivide unchosen feature areas (Figure 5*a*, regions 1, 3 and 4). Only one case remains: when two unchosen features intersect, which are influenced by features from two or more differing source images (Figure 5*a*, region 7). In this case we arbitrarily choose between those features, and copy pixels from the corresponding distorted source image in the manner discussed.

We now perform the previously described distance transform procedure on those pixels not yet assigned, to produce our abstract composition.

C.2 Finer Segmentation

The composition produced at this stage (Figure 6*a*, left) is often composed of pieces larger than those typically found in the Cubist paintings. We wish to further segment non-salient regions to visually 'break up' uninteresting parts of the image, whilst avoiding the imposition of a structure upon those areas.

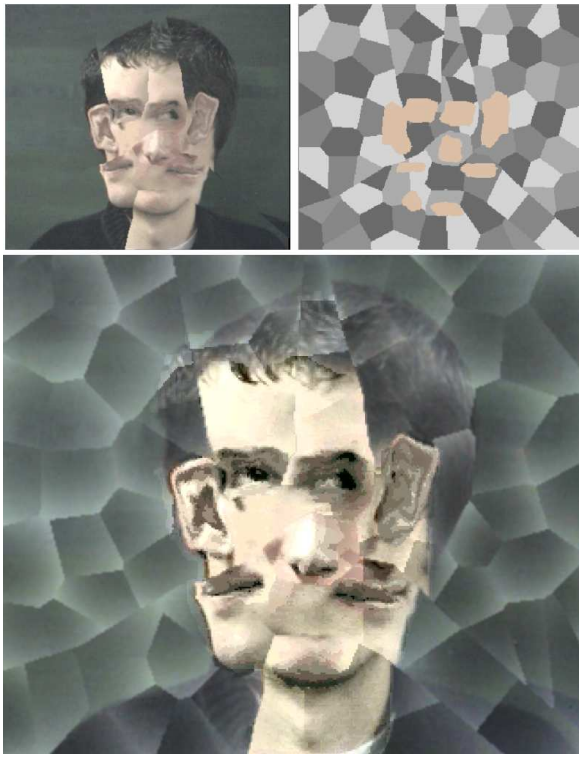

Fig. 6. (a) Composition after application of steps in Section III-C.1 exhibiting large non-salient segments (left) and a uniquely coloured finer segmentation (right) (b) Results of finer segmentation and shading of non-salient areas in the composition.

We initially form a binary mask of each non-salient segment using information from the previous distance transform stage of Section III-C.1, and calculate the area of each segment. We then average the area of the chosen salient features $\mathcal{C}$, to produce a desired 'segment size' for the composition. Each non-salient segment is fragmented into n pieces, where n is the integer rounded ratio of that segment's area to the desired segment size of the composition. To perform the segmentation we produce a dense point cloud of random samples within the binary mask of each non-salient segment. Expectation maximisation [38] is used to fit n Gaussians to this point cloud. We then calculate the Gaussian centre to which each pixel within a given mask is closest; a Voronoi diagram is thereby constructed, the boundaries of which subdivide the non-salient segment being processed into multiple non-salient *fragments*.

Each of the non-salient fragments must now be shaded to break up the composition. We choose a point, or 'epicentre' along each fragment's boundary, and decrease the luminosity of pixels within that fragment proportional to their distance from the epicentre (see Figure 7). The result is a modified intensity gradient across each fragment, simulating light cast over a fragment's surface.

In practice it is desirable that no two adjacent fragments have an intensity gradient of similar direction imposed upon them; doing so induces a noticeable regular structure in non-salient areas, which can divert the viewer's attention from the more interesting *salient* features elsewhere in the composition. Placement of the epicentre at a random location upon the boundary produces too broad a range of possible gradient directions, causing shading to appear as noise. We therefore restrict shading to a minimal set of directions, calculated in the following manner.

A region adjacency graph [37] is constructed over the entire composition; each non-salient fragment corresponds to a node in the graph with vertices connecting segments adjacent in the composition. We then assign a code or 'colour' to each node in the graph, such that two directly connected nodes do not share the same colour. Graph colouring is well-studied problem in computer science, and an minimal colour solution is known to be NP-hard to compute. We therefore use a heuristic based approximation which is guaranteed to return a colouring in P-time, but which may not be minimal in the number of colours used (see Appendix for details). The result is that each fragment is assigned an integer coding in the interval $[1, t]$, where t is the total number of colours used by our approximating algorithm to encode the graph.

The result of one such colouring is visualised in Figure 6*a*. The epicentre of each fragment is placed at the intersection of the fragment's boundary and a ray projected from the centroid of the fragment at angle θ from vertical (Figure 7), where θ is determined by:

$$\theta = 2\pi \left(\frac{segment\ code}{t} \right) \tag{8}$$

This expression guarantees placement of the epicentre at one of t finite radial positions about the boundary of the segment, as the segment coding is an integer value.

The introduction of additional segmentation, and therefore edge artifacts, into non-salient areas of the composition can have the undesired effect of diverting a viewer's gaze from salient features present in the picture. We mitigate against this effect in two ways. First, we convolve the non-salient regions of the image with a low-pass filter kernel such as a Gaussian. This has the effect of smoothing sharp edges between fragments, but conserving the more gradual intensity differential over each non-salient fragment's surface. This also proves advantageous in that the jagged edges of lines partitioning fragments are smoothed. Second, we use a variation upon histogram equalisation [37] to boost contrast within the foreground of the composition (determined during image registration), causing features such as eyes or noses to 'stand out' from the softened segmentation boundaries. Specifically, we calculate the transfer function between the luminosities of the source and equalised compositions. For each pixel in the composition we then interpolate between these luminosities proportional to that pixel's salience; thus contrast is boosted in more salient areas of the composition.

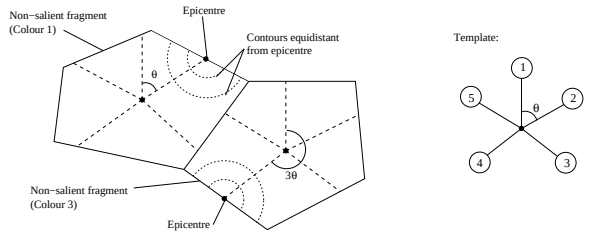

Fig. 7. The geometry of two adjacent non-salient fragments, and a single template determining the location of the epicentre within a fragment. The epicentre of a fragment of colour i lies at the intersection of that fragment's boundary and the ith template spoke.

This produces a final composition such as that of Figure 6*b*. We draw attention to the fact that major segmentation lines (produced by the steps of Section III-C.1) and salient features remain unaffected by this final segmentation of the composition.

D. Image Rendering

The final stage of our algorithm is concerned with creating a painterly effect on the generated composition. We use a novel painting technique adapted from earlier work [13], [23], to which there are two sub-stages: colour quantising, and brush stroke generation. The latter sub-stage makes use of our salience measure to ensure that lower salience strokes, do not encroach upon higher salience regions. This mitigates against loss of detail in the final painting. The rendering process is both entirely automatic and deterministic, deriving stroke parameters from the composition image to produce the painting.

The colour quantising step should be performed prior to composition, but is described here to be alongside our other rendering components. We use variance minimisation quantisation [39], to reduce the colour depth of three independent areas within the image: the distorted salient features ($\mathcal{F}'$); the foreground of each distorted image; and the background of each distorted image. Distinction between foreground and background is made by thresholding upon a simple characteristic property of the image, such as hue or intensity (as was performed during image registration).

Our motivation to quantise follows the observation that an artist typically paints with a restricted palette, and often approximates colours as a feature of the Cubist style [40]. We allow a level of control over this effect by differentiating the level of quantisation over the various image components, and have found that heavy quantisation of the features and foreground, contrasted by a lesser degree of background quantisation can produce aesthetically pleasing effects.

At this stage we optionally introduce false colour to the image. Artists such as Braque and Gris often painted in this manner, contrasting shades of brown or grey with yellows or blues to pick out image highlights. We use a look-up table based upon a transfer function, which generates a hue and saturation for a given intensity, calculated from the original input colour. Typically we define this function by specifying several hue and saturation values at various intensities, and interpolate between these values to produce a spectrum of false colour to populate the look-up table.

The second step of the rendering process concerns the generation of "painted" brush strokes. We use the composition produced thus far as a reference image (in a manner similar to Haeberli [13]), and sample at regular spatial intervals to produce three dimensional brush strokes; inverted cones with superquadric cross-section. These strokes are shaded according to the reference image at the point of sampling. Each stroke is z-buffered and the result is projected orthogonally onto the (2D) image plane to generate the final image (Figure 8*a*).

There are seven parameters to each stroke; a, b, r, α (see equation 3), colour $j(c)$, orientation angle θ and height h. Each of these parameters is automatically set. Parameter α determines the form of the stroke, and is preset by the user. Low values (< 1) of α create cross-sections of a rectangular form, giving the image a chiselled effect, whilst higher values of α produce jagged brush styles. Function $j(c)$ transforms, or 'jitters', the stroke colour c by some small, uniformly distributed random quantity, limited by a user defined amplitude ϵ. By increasing ϵ, impressionist results similar to Haeberli's [13] can be produced, and some interesting effects can be created by blending the styles of Cubism and Impressionism. Further brush styles can also be produced by texturing the base of each cone with an intensity displacement map, cut at a random position from a sheet of texture. In particular, we find that this process greatly enhances the natural, 'hand-painted' look of the resulting image. The remaining five stroke parameters (a, b, r, θ, and h) are calculated by an automated process which we now describe.

Stroke height h is proportional to image salience at the point of sampling. We use the technique explained in Section III-A to construct a salience map of the reference image. Higher salience image pixels tend to correspond to the features and detail within the image, and so produce strokes of greater height that rise above the canopy of lower salience strokes in the z-buffer. In effect, the least salient strokes are laid down first, much as an artist might use a wash to generate wide expanses of colour in an image, and fill in the details later; the strokes are laid down in an implicit order. Without this sensitivity to salience, the rendering procedure can obscure regions of high salience with strokes of lower salience, causing loss of detail (see Figure 8*b*).

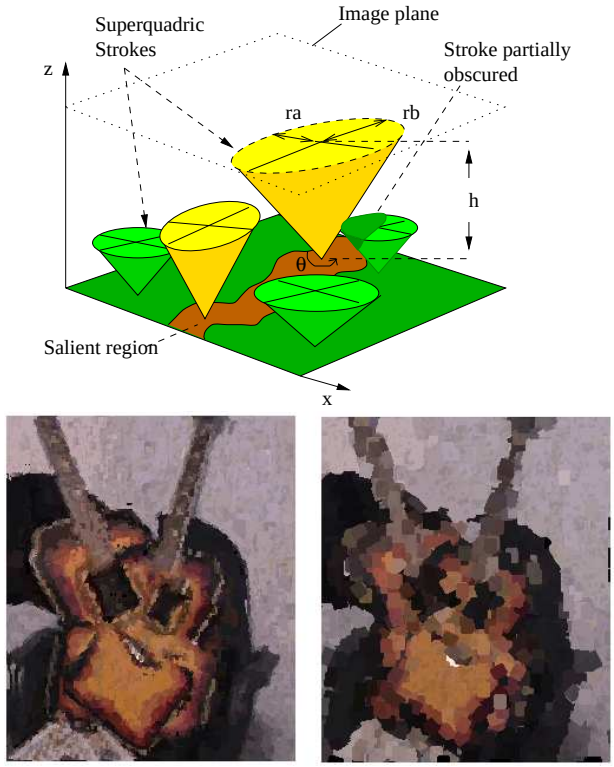

Fig. 8. (a) Strokes take the form of inverted cones with superquadric cross-section, and are z-buffered (top) to produce the final image; (b) Strokes applied with (left) and without (right) salience adaptation. We make the qualitative observation that salient detail is conserved with salience adaptation.

We also derive gradient information from the reference image, by convolving the intensity image with a Gaussian derivative of first order. The scale of the base of the cone r is set inversely proportional to the magnitude of the gradient. This causes small, definite strokes to be painted in the vicinity of image edges, corresponding to the detail in the image. Larger strokes are used to shade non-edge areas. Again, this mimics the manner in which an artist may paint large strokes to colour areas, and sharply define the edges of an image. In many images such detailed areas also tend to be salient. Hence our method tends to draw low, fat cones in regions of low salience, and tall, narrow cones in regions of high salience.

Stroke orientation θ is set using the gradient orientation, the larger axis of the superquadric being aligned tangentially. However in areas where the edge magnitude is low, orientation derived in this manner becomes less reliable. We therefore vary the eccentricity of the superquadric (a, b) in relation to magnitude of the edge gradient at the position sampled. If the gradient is low, then $a \approx b$, and orientation becomes less important as the superquadric is not greatly expressed in either horizontal or vertical directions. Where the edge gradient is high, then $a > b$ and the superquadric stretches out along the edge. One emergent property of this approach is that strokes along an edge

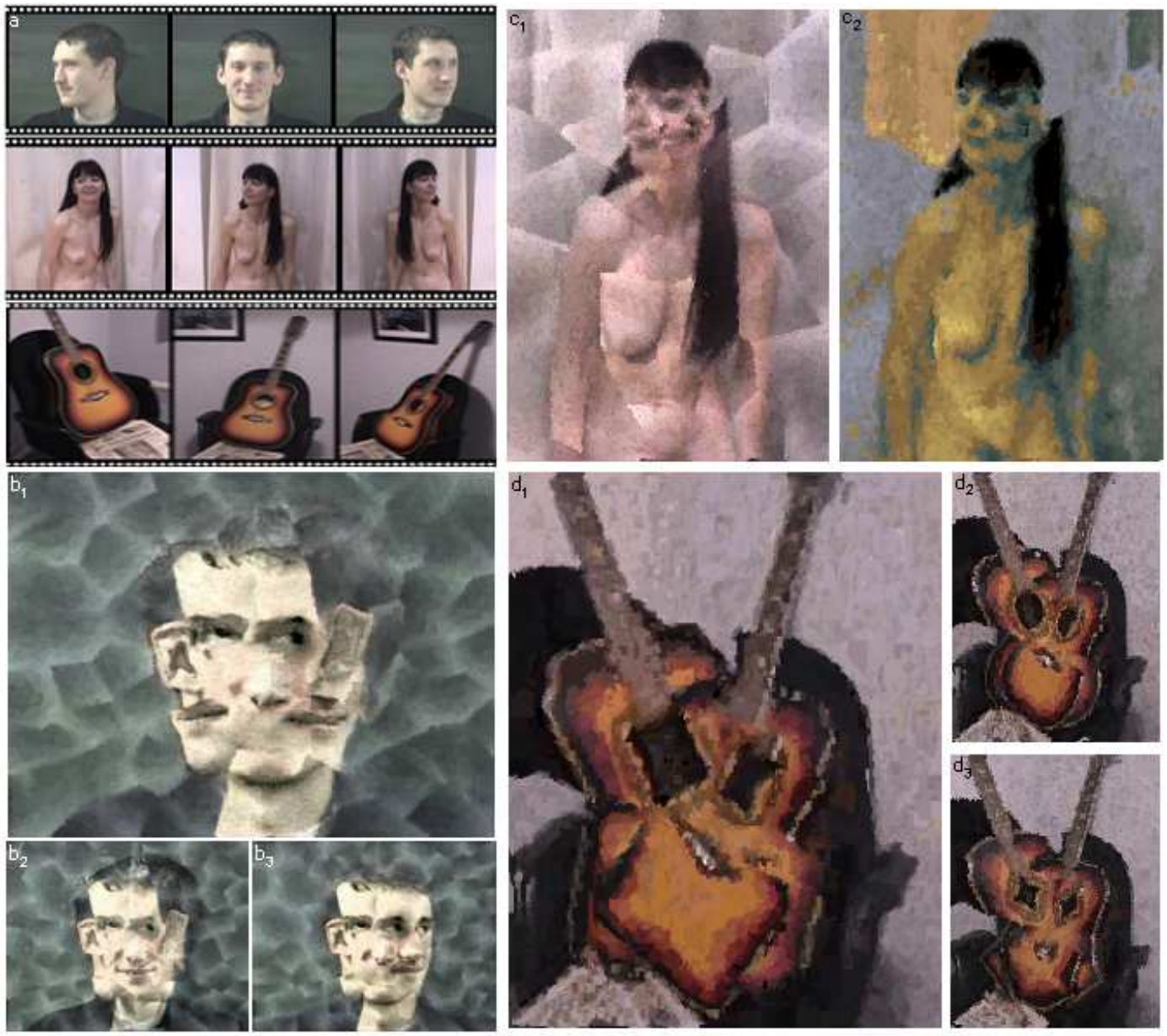

Fig. 9. A gallery of images illustrating the application of our rendering algorithm.

tend to merge. This, combined with the colour quantisation performed earlier, causes edge highlights to appear as though produced by fewer, longer strokes. This is typical of the manner in which an artist might manually render such highlights, and adds aesthetic quality to the image.

IV. Gallery of Images

We present the results of applying our algorithm to three image sets; a portrait, a guitar, and a nude. These subjects were popular choices for artists of the Cubist period, and we use them to demonstrate the processes of composition, distortion, and painting respectively.

The original source image sets were captured using a digital video camera, and are given in Figure 9*a*. Figure 9*b* presents the results of processing the portrait images; salient features were the ears, eyes, nose and mouth. Figures $9b_1$ and $9b_2$ were created by successive runs of the algorithm, using identical distortion parameters; the stochastic nature of feature selection produces varying compositions in the same visual style. Figure $9b_3$ demonstrates the consequence of relaxing the constraints which maintain proportion between equivalence classes during composition; equivalence classes are no longer proportionally represented. In Figure $9b_2$ we demonstrate the results of applying the histogram equalisation described in Section III-C.2 to the composition background. This can produce a sharper contrast within salient areas of the background (such as the hair or jumper), and may be artistically preferable to some.

The nude has been rendered with minimal distortion; salient features were the eyes, nose, mouth, chest and arm. False colour has been introduced to Figure $9c_2$, using the complementary colours of blue and orange to contrast highlight and shadow. Many abstract artists also make use of complementary colour pairs in this manner. Finer fragmentation of non-salient regions was not performed when generating this result.

Paintings produced from the guitar images are presented in Figure $9d$; salient features were the hole, neck, bridge and chair arms. Figures $9d_1$, $9d_2$, and $9d_3$ are identical compositions rendered with different distortion and painting parameters. The values of distortion parameter α' for each of the renderings is 0.5, 1, and 2 respectively. Notice how the hole in the guitar changes shape, from rectangular to star-like. By changing only these parameters, a varied range of styles are produced. The finer segmentation of non-salient regions was not performed on these renderings, to allow clear demonstration of distortion effects.

V. Discussion and Conclusions

We have shown that Cubist-like art can be produced by computer, almost entirely automatically. Input images are unrestricted in content, we have demonstrated our work with portraits, nudes, and still-life — chosen to reflect typical subject matter of the Cubists themselves. The composition of each output image is stochastically decided, and so a potentially original rendering is possible with each new pass over the same input set. Distortion and final painting is wholly deterministic, except for a "jitter" the amplitude of which can be set to zero.

The thrust of our work was motivated by an artistic genre, rather than a particular technique. Cubism was chosen because it forced us to think carefully about grouping and high-level feature composition, and allowed us to produce novel images that depict motion in a still image, an area which is as yet under-researched. A wide variety of professional artists (from traditional painters to computer animators) judge our output to be of high aesthetic quality. Indeed, some professional artists have advised us already on output, and are keen work on related future projects with us. Such judgements are important, given the absence of any specific ground truth.

Our definitions of salience and high-level features, are the key technical contributions. We correlate salience with rarity, which we have found to be general, but requires interaction for grouping. A more restrictive definition may remove the need for interaction, at the cost of generality. For example statistical cluster-based grouping might permit automatic segmentation of compact features, but cope poorly with elongated features. Typically interactive grouping takes less than one minute of user time, and so we are content with our method for the time being at least. While our specific salience definition is new to computer graphics, others have recognised the value of some kind of importance measure [41] within artificial drawing, and in this sense we continue a trend. Our use of high-level features, by which we mean eyes, mouths (in general any identifiable object or sub-part of an object) is arguably more unique. The use of such high-level features is the single most important element of our work; without them we would not be able to produce our novel Cubist-like compositions, nor could we render preferentially. In this work 'preferential rendering' means boosting the contrast of eye regions, for example, but it is easy to imagine other work in which preferential treatment of important areas is desirable.

Specific elements of our method can no doubt be improved upon. Whilst the algorithm may, in principal, handle an arbitrary number of source images, we assume that the foreground and background of those images have distinct colour signatures allowing us to perform registration. Although a more sophisticated registration method may be required for heavily cluttered scenes, we note that our salience measure is able to isolate objects of interest even in the presence of substantial clutter (Figure 1). Salient features are currently selected for composition based upon a set of constraints, one of which ensures that features do not intersect. Since the shape of features may change during geometric distortion, it follows that distortion must occur *prior to* the selection stage. An interesting alternative might be to combine the distortion and selection stages; features could be distorted to conceal artifacts such as partial mouths (Figure 5b) rather than colouring such artifacts with non-salient texture (as in Section III-C.1).

We might seek to improve composition by enforcing straight-line boundaries between segments, as the Cubists once did, whereas now they can appear curved. One way to do this would a linear discriminant analysis [42], but the additional algorithmic complexity is unattractive to us. We could consider undertaking a compositional analysis in order to more aesthetically place our high level features, and progress yet further toward emulating Cubism; we believe such an analysis is a considerable challenge that is not necessary to demonstrate the synthesis of Cubist-*like* renderings are possible. We might revisit the way in which we apply paint, so that it appears more in the tradition of a particular artist, but there is no compelling reason to focus our work in such a way at this stage: the manipulation of high-level features is the only necessary step to producing images that can be classified as "Cubist" or, at least, "Cubist influenced". We believe that the most productive avenues for future research will not be in incremental refinements that make our images tend ever closer toward traditional Cubism, rather the best avenues to follow will be those exploring alternative uses for high-level features.

Acknowledgements

The authors would like to thank Nanomation Ltd. for early discussions, and the anonymous reviewers for their

constructive comments. This work was supported by EPSRC grant GR/M99279.

References

[1] P. Hall, "Non-photorealistic rendering by Q–mapping," *Computer Graphics Forum*, vol. 1, no. 18, pp. 27–39, 1999.

[2] J. Lansdown and S. Schofield, "Expressive rendering: a review of non-photorealistic rendering techniques," *IEEE Computer Graphics and Applications*, vol. 15, no. 3, pp. 29–37, May 1995.

[3] C. Reynolds, "Non-photorealistic, painterly, and 'toon rendering," Worldwide web.

[4] S. Green, D. Salesin, S. Schofield, A. Hertzmann, and P. Litwinowicz, "Non-photorealistic rendering," *SIGGRAPH '99 Non-Photorealistic Rendering Course Notes*, 1999.

[5] G. Gooch and A. Gooch, *Non-photorealistic rendering*, A. K. Peters, U.S.A., July 2001.

[6] G. Elber, "Interactive line art rendering of freeform surfaces," *Computer Graphics Forum*, vol. 3, no. 18, pp. 1–12, September 1999.

[7] T. Strothotte, B. Preim, A. Raab, J. Schumann, and D. R. Forsey, "How to render frames and influence people," in *Proceedings Computer Graphics Forum (Eurographics)*, Oslo, Norway, 1994, vol. 13, pp. C455–C466.

[8] C. Rossl and L Kobbelt, "Line-art rendering of 3d-models," in *Proceedings of Pacific Graphics*, 2000, pp. 87–96.

[9] M. P. Salisbury, M. T. Wong, J. F. Hughes, and D. H. Salesin, "Orientable textures for image-based pen-and-ink illustration," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, Los Angeles, USA, 1997, pp. 401–406.

[10] O Deussen, S. Hiller, C. van Overveld, and T. Strothotte, "Floating points: A method for computing stipple drawings," in *Proceedings Computer Graphics Forum (Eurographics)*, 2000, vol. 19, pp. 41–50.

[11] P. Hanrahan and P. Haeberli, "Direct wysiwyg painting and texturing on 3d shapes," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, August 1990, vol. 24, pp. 215–223.

[12] E. Daniels, "Deep canvas in disney's tarzan," in *Proceedings Computer Graphics (ACM SIGGRAPH, Abstracts and Applications)*, 1999, p. 200.

[13] P. Haeberli, "Paint by numbers: abstract image representations," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, 1990, vol. 4, pp. 207–214.

[14] B. Baxter, V. Scheib, M. C. Line, and D. Manocha, "DAB: Interactive haptic painting with 3d virtual brushes," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, 2001, pp. 461–468.

[15] S. Mizuno, M. Okada, and J. I. Toriwaki, "Virtual sculpting and virtual woodcut printing," *The Visual Computer*, vol. 2, no. 14, pp. 39–51, 1998.

[16] T. Cockshott, J. Patterson, and D. England, "Modelling the texture of paint," *Computer Graphics Forum*, vol. 11, no. 3, pp. 217–226, September 1992.

[17] C. Curtis, S. Anderson, J. Seims, K. Fleischer, and D. H. Salesin, "Computer-generated watercolor," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, 1997, pp. 421–430.

[18] S. Strassmann, "Hairy brushes," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, 1986, vol. 20, pp. 225–232.

[19] S. C. Hsu and I. H. H. Lee, "Drawing and animation using skeletal strokes," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, Orlando, USA 1994, pp. 109–118.

[20] G. Winkenback and D. H. Salesin, "Computer-generated pen-and-ink illustration," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, Orlando, USA, 1994, pp. 91–100.

[21] W. Leister, "Computer generated copper plates," *Computer Graphics Forum*, vol. 13, no. 1, pp. 69–77, March 1994.

[22] B. Meier, "Painterly rendering for animation," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, 1996, pp. 447–484.

[23] P. Litwinowicz, "Processing images and video for an impressionist effect," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, Los Angeles, USA, 1997, pp. 407–414.

[24] A. Hertzmann, "Painterly rendering with curved brush strokes of multiple sizes," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, 1998, pp. 453–460.

[25] M. Shiraishi and Y. Yamaguchi, "An algorithm for automatic painterly rendering based on local source image approximation," in *First Internation Sympsoium on Non-Photorealistic Animation and Rendering (NPAR)*, Annecy, France, 2000, pp. 53–58.

[26] L. Markosian, M. A. Kowalski, S. J. Trychin, L. D. Bourdev, D. Goldstein, and J. F. Hughes, "Real-time nonphotorealistic rendering," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, Los Angeles, USA, 1997, pp. 415–420.

[27] E. Praun, H. Hoppe, M. Webb, and A. Finkelstein, "Real-time hatching," in *Proceedings Computer Graphics (ACM SIGGRAPH)*, 2001, pp. 581–586.

[28] A. Hertzmann and K. Perlin, "Painterly rendering for video and interaction," in *First Internation Sympsoium on Non-Photorealistic Animation and Rendering (NPAR)*, 2000, pp. 7–12.

[29] A. Butler, C. Van Cleave, and S. Stirling, *The Art Book*, The Phaidon Press, 1994.

[30] A. Chen, K. Knudtzon, J. L. Stumpfel, and J. K. Hodgins, "Artistic rendering of dynamic motion," in *Proceedings Computer Graphics (ACM SIGGRAPH Sketches)*, 2000.

[31] A. W. Klein, P. J. Sloan, R. A. Colburn, A. Finkelstein, and M. F. Cohen, "Video cubism," Tech. Rep., Microsoft Research, May 2001.

[32] K. N. Walker, T. F. Cootes, and C. J. Taylor, "Locating salient object features," in *Proceedings BMVC*, 1998, vol. 2, pp. 557–567.

[33] P. Hall, D. Marshall, and R. Martin, "Merging and splitting eigenspaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 22, pp. 1024–1049, September 2000.

[34] C. B. Barber, D. P. Dobkin, and H. T. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions of Mathematical Software*, , no. 22, pp. 469–483, 1996.

[35] P. L. Rosin, "Fitting superellipses," *IEEE Transactions of Pattern Analysis and Recognition*, vol. 22, no. 7, pp. 726–732, July 2000.

[36] W. H. Press, *Numerical Recipes in C: the art of scientific computing*, Cambridge University Press, 1992.

[37] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*, P.W.S. Publishing, 1999.

[38] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Royal Statistical Society Series B*, vol. 39, pp. 1–38, 1997.

[39] X. Wu, "Colour quantization by dynamic programming and principal analysis," *ACM Transactions on Graphics*, vol. 4, no. 11, pp. 348–372, October 1992.

[40] R. Hughes, *The Shock of the New*, Thames and Hudson, 1991.

[41] A. Santella and D. DeCarlo, "Abstracted painterly renderings using eye-tracking data," in *Second International Symposium on Non-Photorealistic Animation and Rendering (NPAR)*, 2002.

[42] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, Wiley-Interscience, 2nd edition edition, 2001.

[43] G. J. Chaitin, M. A. Auslander, A. K. Chandra, J. Cocke, M. E. Hopkins, and P. W. Markstein, "Register allocation via colouring," *Computer Languages*, vol. 6, pp. 47–57, 1981.

Appendix

We give details of the P-time approximation used to colour our region adjacency graph in Section III-C. The algorithm is guaranteed to return a graph where no two directly connected nodes are assigned the same colour. The algorithm may use a greater *total number* of colours than the minimum possible solution. The algorithm trades accuracy for speed of execution; a precise solution is known to be NP-hard. However, the heuristics implicit in this approximation have been empirically shown to guide the algorithm toward a close approximation to the minimum colouring. In our application, adjacency graphs are often encoded using eight or fewer colours. The algorithm originates in machine-language compilation literature, where a limited set of registers are allocated to potential larger set of program variables using a graph colouring approach [43].

Require: graph G of n nodes denoted by G_i $[1 \leq i \leq n]$

1: Integer $maxcols \leftarrow n$

2: Integer $mincols \leftarrow 4$

```
3:  for totalcols = mincols to maxcols do
4:    Graph W ← G
5:    Ordered List L ← ∅
6:    repeat
7:      R ← index of node W_R with greatest count of
        immediate neighbours, that count not exceeding
        totalcols − 1
8:      if R ≠ ∅ then
9:        remove node R from graph W
10:       add node R to list L
11:     end if
12:   until W == ∅||R == ∅
13:   if W == ∅ then
14:     Define empty sets C_i where [1 ≤ i ≤ totalcols]
15:     for idx = n to 1 do
16:       for i = 1 to totalcols do
17:         if G_idx is not a direct neighbour of any nodes
            in C_i then
18:           C_i ← C_i ∪ G_idx
19:         end if
20:       end for
21:     end for
22:   end if
23: end for
```

The algorithm terminates with all members of C_i being assigned some colour i, such that two directly connected nodes do not share the same i. The algorithm operates by repeatedly 'loosening' the graph; each iteration of the inner *repeat* loop eliminating the colourable node with highest connectivity. This is the heuristic that guides the algorithm, and seems intuitively correct. Were we to pursue the consequences of eliminating each node at each iteration of the inner *repeat* loop, one branch of the resulting decision tree would terminate at a minimal solution. However this would produce a combinatorial search of all possible colourings.

John Collomosse received the B.Sc. in computer science from the University of Bath in 2001, and is currently a doctoral student at the same institution. His research focuses upon the application of computer vision to the problem of image based non-photorealistic rendering. He is a student member of the BMVA and Eurographics UK.

Peter Hall is a Lecturer (Assistant Professor) in the Department of Computer Science, University of Bath, he was appointed in 1999. His first degree is in Physics with Astrophysics; and his Ph.D. is in Scientific Visualisation. Concurrent with his Ph.D. he worked in medical imaging. His interests in the visual arts and in seeing combine to produce research interests in the intersection of computer vision and computer graphics. He holds subsidiary interests in computer learning using statistical methods, with special emphasis on incremental learning methods. Current work includes on-line modelling walking motions, perceptually based texture classification, content based video indexing, and methods for learning to draw based on machine vision.