Department of Computer Science

Technical Report

Stroke Surfaces: A Spatio-temporal Framework for Temporally Coherent Non-photorealistic Animations

J. P. Collomosse, D. Rowntree and P. M. Hall

Technical Report 2003-01 **June 2003**

ISSN 1740-9497

Contact Address:

Department of Computer Science

University of Bath

Bath, BA2 7AY

United Kingdom

URL: http://www.cs.bath.ac.uk

ISSN 1740-9497

Stroke Surfaces: A Spatio-temporal Framework for Temporally Coherent Non-photorealistic Animations

J. P. Collomosse[1], D. Rowntree[2] and P. M. Hall[1]

[1] Department of Computer Science, University of Bath, Bath, England.
[2] Nanomation Ltd., 6 Windmill Street, London, England.

Technical Report CSBU 2003-01 (June 2003)

Abstract

The contribution of this paper is a novel framework for the automated synthesis of non-photorealistic animations from video sequences. Our approach is unique in that we interpret the source video sequence as a spatio-temporal voxel volume, with time as the third dimension. Video frames are segmented into homogeneous regions, and heuristic associations between regions formed over time to produce a collection of conceptually high level spatio-temporal objects. These objects carve sub-volumes through the video volume delimited by continuous isosurface "Stroke Surface" patches. By manipulating objects in this representation we are able to synthesise a wide gamut of artistic effects, which we allow the user to stylise and influence through a parameterised "Video Paintbox". In addition to novel temporal effects unique to our method we demonstrate the extension of 'traditional' static NPR styles to video including painterly, sketchy and 'toon shading effects. An application to advanced rotoscoping is also identified. The high level of analysis afforded by our spatio-temporal approach allows us to maintain a high degree of temporal coherence; a property scarce in current NPR video techniques all of which process video at a low level (on a per pixel, per frame sequential basis). The paper concludes with a critical appraisal and discussion of future applications for the Stroke Surface representation, including potential for video compression.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Animation

Keywords: Non-photorealistic Animation, Spatio-temporal video processing, Cartoon, Video Paintbox, Stroke Surfaces

1. Introduction

In this paper we present the "Video Paintbox", a novel framework for the automated synthesis of non-photorealistic animations from video sequences. Our approach is unique in that we use a high level interpretation of the source video, treating the image sequence as a spatio-temporal voxel volume; effectively a stack of sequential frames in which time forms the third dimension. A spatio-temporal segmentation of the video is initially performed, resulting in multiple sub-volumes carved by objects moving over time. We demonstrate that by manipulating video in this representation, we are able to re-render footage in a wide range of temporally coherent non-photorealistic styles. In particular we demonstrate how a reference frame may be attached to objects, enabling the extension of traditional stroke and texture based NPR techniques to video (for example painterly effects). In addition we demonstrate novel temporal video effects unique to our framework, and identify an application to advanced rotoscoping. Users select from the gamut of available rendering styles by varying parameters on the paintbox.

Our work is motivated by a desire to render video in cartoon-like styles, a problem which decomposes into two separable sub-goals: 1) producing temporally coherent stylised shading effects in the video; 2) emphasising motion within the image sequence. This paper addresses the former of these issues and complements work recently published by the authors in [1, 2], which address the latter issue.

Animation is a costly, labour intensive process and the ability to shoot footage for later automatic rendering in artistic styles would see clear application in the entertainment industry. Indeed, a number of recent movies such as "Waking Life" and "What Dreams May Come" have included clips of cartoon shaded and painterly video respec-

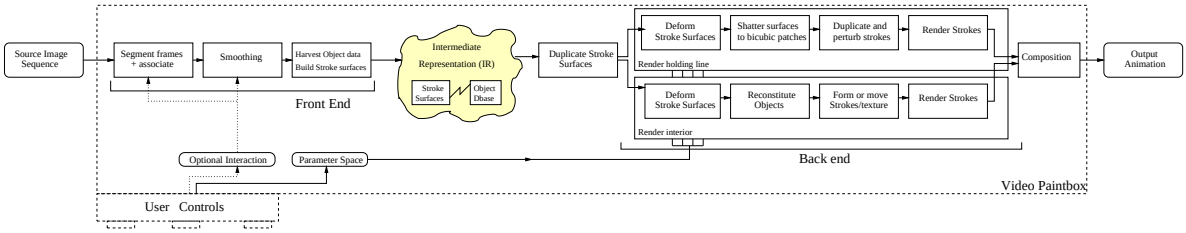

Figure 1: *Illustrating information flow within the rendering framework. The front end analyses the source video to produce an intermediate representation (IR), and is largely automated through Computer Vision techniques. The back end renders the IR in one of a variety of artistic styles, selected by user parameters.*

tively; although many hundreds of man hours of manual interaction or correction were required to generate such effects [3].

Whilst image-based NPR techniques have become increasingly common in the literature, few algorithms currently exist for the synthesis of non-photorealistic animations from 2D image sequences. Those that do typically produce animations exhibiting poor temporal coherence, manifested as an uncontrolled flickering within the rendered animation, termed *swimming*. Artificial drawing techniques are predominantly stroke-based, and temporal incoherence occurs principally when the motion of strokes (or more generally, motion within the resulting animation) does not agree with the perceived motion of content within the source image sequence. This observation is reinforced by perceptual research for example the Gestalt 'common fate' cue [4], where objects moving in a similar manner become grouped. Conflicts between the motion of phantom objects perceived due to grouping, and physical objects, contribute to the distracting nature of swimming. Wong *et al* [5] observe that rapidly flickering dots are perceived to 'pop-out' from their neighbours; explaining the confused sense of depth apparent in an animation with poor stroke coherence. Unfortunately this pop-out effect manifests most strongly at around 6Hz, close to the aesthetically optimal frame rate for coherent painterly animations determined by Hertzmann *et al* [6].

One might naively hope to produce non-photorealistic animations by applying existing artificial drawing processes (for example static painterly or sketchy filters) to independent frames of the video sequence. This approach typically fails to produce coherent animations for two reasons. First, many artificial drawing algorithms employ a pseudo-random element to mask the deterministic nature of the machine. For example, many painterly rendering techniques [7, 8, 9] randomise the order of the painting of individual strokes, thus suppressing a regularity that 'flattens' the appearance of the painting. Second, those techniques that do operate deterministically (for example the painterly technique of [10]) are still influenced to some degree by noise and so are also non-deterministic in some sense. Other unsuccessful solutions include fixing the spatial attributes of strokes and varying stroke colour according to source frames. This gives the impression of motion 'behind a shower door' and causes severe loss of salient detail in the rendering as strokes are no longer aligned tangential to salient edges within the image.

Consequently attempts have been made to match stroke motion with the motion of the source video content. The current state of the art is to paint strokes upon the first frame of video, and to translate strokes from frame to frame based upon an estimate of the motion field between frames. Current techniques use either optical flow [8, 6] or frame differencing operations [6] to produce this motion estimate. The commonality between all current algorithms is that the rendering of video proceeds on a per frame sequential basis. In a temporal sense such algorithms are local, greedy approaches and as such have clear disadvantages in producing a globally optimum (in this case temporally coherent) solution. Errors and inaccuracies accumulate with each frame processed and propagate forward to all subsequent frames in the video stream. Moreover, errors can accumulate quite rapidly as motion estimation techniques such as optical flow make many simplifying assumptions (e.g. no occlusion) which are often violated in real image sequences [11]. Each frame is rendered prior to analysis of subsequent frames (and then only taking into account the single previous frame). If one were processing video for interaction or real-time animation then a frame by frame approach would be justified (an example is Hertzmann's "Living Painting" [6]). However the motivation of this paper is to produce post-production video effects. If one does not require real-time processing, then it seems intuitively correct to use all information available (i.e. all video frames) when rendering, to improve aesthetics of the animation.

We therefore argue for a higher level analysis of image sequences for NPR than is present in existing methods. Spatially, we operate at a higher level by segmenting images into regions. These regions are homogeneous in their visual attributes, for example colour and motion. This contrasts with the majority of NPR methods which operate as image filters at the pixel level (though some recent static painterly techniques use segmentation [12, 13]). Since attributes are set on a per region rather than per object basis, contradictory visual cues do not arise, for example where stroke motion differs within a given object. Temporally we work at a higher level, for example by smoothing region attributes over adjacent frames to mitigate incoher-

ence; this is in contrast to all existing NPR video methods. We believe the paradigm of processing video at this higher level to be a unique and valuable approach to the problem of synthesising NPR animations from video.

2. Overview of the Video Paintbox

The Video Paintbox consists of a single rendering framework which may be broken into a front and back end (Figure 1). The front end is responsible for the parsing of the source video to create an intermediate representation, and is largely automated through application of Computer Vision techniques. This abstracted video representation is then passed to the back end, where it is rendered in one of a range of artistic styles. We wish for minimal user action with the front end, which must robust and general. In contrast, the user is given control over the back end of the system via a set of parameters which influence the style of the resulting animation. We describe the operation of front and back end components in Sections 3 and 4 respectively.

3. Front end: Building the Representation

We now describe in detail the front end of the video paintbox, responsible for generating the intermediate representation. Frames are independently segmented into connected homogeneous regions using standard 2D Computer Vision techniques. The criterion for homogeneity we have chosen is colour (after [12]), but one might equally well segment on the basis of texture or motion; the nature of the video content influences such a choice. For each frame, associations are created between that frame's regions and those of frames adjacent to it. The result is a series of connected homogeneous sub-volumes carved from the spatio-temporal video volume, describing the trajectory of features in the video. These sub-volumes are smoothed and represented in terms of their interfacing surface patches, which we term *stroke surfaces*. Edge detail within the regions is similarly computed and encapsulated in the stroke surface representation. A supplementary database is also maintained, containing information such as the colour and local motion parameters of objects. Each stroke surface maintains a winged edge structure containing pointers to the two objects on either side of it; these reference the supplementary database. The stroke surfaces and database counterpart collectively form the intermediate representation (IR) which is passed to the back end for rendering (Section 4).

3.1. Video Segmentation

We assume motion within the image sequence to be smooth, that is, free from scene changes (for example cuts and cross-fades). A plethora of algorithms exist (mostly colour-histogram based [14]) which can segment a video sequence into such 'cut-free' chunks; such a segmentation may be considered a pre-processing step to our method.

When creating associations between frames, an important property of a segmentation algorithm is that of robustness; segmentations of adjacent frames must yield similar class maps to permit region association. For the purpose of segmentation algorithm selection we have empirically evaluated the coherence of six contemporary algorithms on twenty short clips of artificial and natural scenes. For our purpose, a natural scene is one of complex, cluttered content (e.g. Figure 2, *SOFA*), whilst an artificial scene is typically an uncluttered, low complexity scene with few homogeneous colour regions (e.g. Figure 2, *BEAR*). We measured the coherence of a segmented sequence as follows. Coherence between adjacent frames was computed as the mean squared error between distance transforms of each frame's segmented region boundaries. The coherence of an entire sequence was then evaluated as this mean error taken over all frames. The colour segmentation algorithms we tested were the Recursive histogram split [15], Split and Merge [15], Colour Segment Code (CSC) [16] in both RGB and HSV space, EDISON [17] (a synergistic approach based on edge and colour mean shift [18] operators), and JSEG [19]. Results indicated the EDISON system to be preferable in the case of natural scenes, and that artificial scenes were best segmented using the CSC algorithm operating in HSV space.

We note that an alternative methodology would be to segment the video in one pass as a single volume. Whilst such an methodology is in keeping with our spatio-temporal approach, there are reasons to avoid a 3D segmentation:

- Attributes such as the shape, colour or shading of a region are permitted to evolve gradually over time with our 2D+time approach. Such variation is difficult to accommodate within the framework of a single 3D segmentation (without introducing further models and constraints)
- Small, fast moving objects may form disconnected volumes in 3D, resulting in temporal over-segmentation. However these discontinuities do not arise with our proposed 2D+time association scheme between frames.
- Many spatial phenomena, such as texture or edges, are not easily extensible to a 3D volume representation without some form of constraining motion model [20] which is incompatible with our general application. Colour segmentation can also become problematic, since the more coherent algorithms employ a synergistic approach to segmentation (e.g. EDISON uses edges as well as colour to determine region boundaries).
- For pragmatic reasons. The problem of 2D image segmentation has received extensive study from the Computer Vision community, in contrast to 3D segmentation (exceptions lie within medical imaging, but do not deal with the problem domain of video imagery). However the modular nature of our framework (Figure 1) is loosely coupled with the segmentation technology used, and we allow for algorithm substitution as improved techniques appear in the literature.

After each video frame has been independently segmented, associations are created between regions in adjacent frames to produce video volumes. We now explain this association process in some detail.

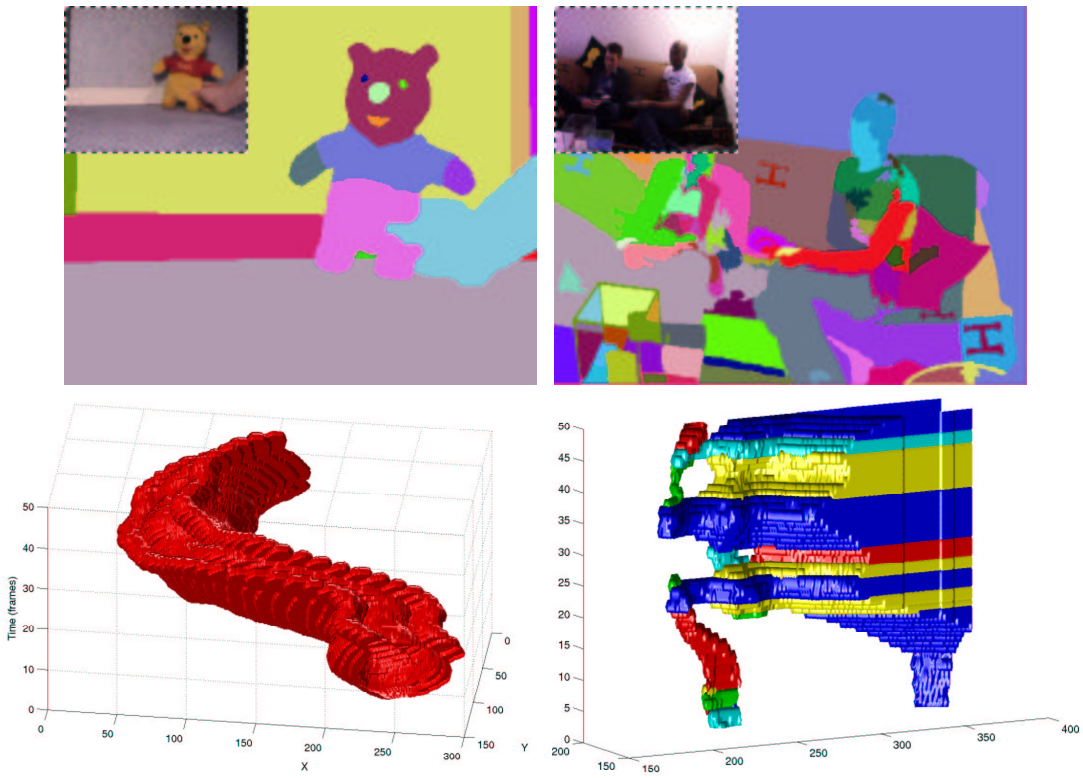

Figure 2: *Above: Sample segmented frames from the* BEAR *and* SOFA *sequences with original footage inset. Below: Two visualisations from the* BEAR *video volume, corresponding to the head and the right hand section of the skirting board. Observe that whilst the head produces a single video object, the skirting board is repeatedly occluded by the bear's hand during movement causing division of regions. This resulting volume consists of a connected structure of several temporally convex video objects (coloured individually for clarity).*

3.2. Region association algorithm

The problem of associating sets of regions, each within multiple frames, is combinatorial in nature and an optimal solution clearly can not be found through exhaustive search for any practical video. We propose a two stage heuristic solution to the association problem, which we have found to work sufficiently well in practise (that is, results in a locally optimal solution where associated objects exhibit an acceptable level of temporal coherence). First, for each frame we generate associations between regions in that frame and those in frames adjacent to it. These associations are made according to heuristics based on mutual colour, area, spatial overlap, and shape. Second, the resulting chains of associated regions are filtered using a graph based search. Association is complicated by the fact that objects may merge or divide in the scene. For example, a ball passing behind a post might appear to split into two regions, and then recombine. Note it is satisfactory to represent an occluded real object as multiple imaged regions as these regions become associated in a form of temporal graph, as a product of the association process.

We observe that in a robust segmentation: 1) properties of regions such as shape, colour, area, are subject only to minor change over short periods of time. The exceptions are the instances at which regions merge or divide; 2) the presence of a region in the segmentation should also be stable over short periods of time, the exceptions are the instances at which a region appears or disappears; 3) although regions may merge or divide over time, such events should be for the relative long-term (given a video frame rate of 50Hz) and not be subsequently reversed in the short-term. Observations 1 and 2 influence the choice of heuristics for the first stage of processing, while observation 3 governs the second stage.

3.2.1. Association heuristics

Given a single region (R_t) in frame t, we wish to find the set of regions in the previous frame ($\mathcal{R}_{t-1}$) which map to R_t, and the set of regions in the subsequent frame ($\mathcal{R}_{t+1}$) that R_t maps on to. Thus R_t could potentially map to zero or more regions in adjacent frames (Figure 3a). The suitability for two regions in adjacent frames $[R_t, r \in \{\mathcal{R}_{t-1} \| \mathcal{R}_{t+1}\}]$ to be associated may be written as a weighted energy score $E(.)$:

$$E(R_t,r) = \begin{cases} 0 \text{ if } \delta(R_t,r) > \Delta \\ w_1\sigma(R_t,r) + w_2\alpha(R_t,r) - \ldots \\ w_3\delta(R_t,r) - w_4\gamma(R_t,r) \text{ otherwise} \end{cases} \quad (1)$$

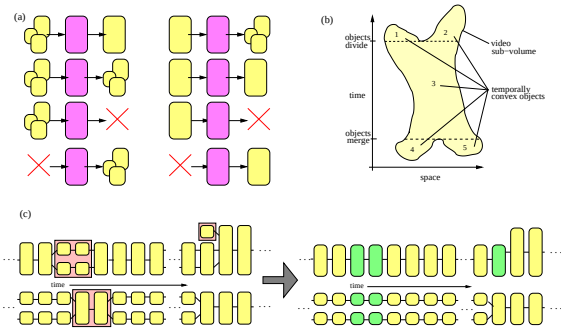

Figure 3: *(a) Eight cases for region association. Associations are created between a region in the current frame and potentially many regions in adjacent frames. (b) Example of a single video-sub-volume split into five temporally convex objects (c.f. Figure 2d). (c) A region association graph before and after graph filtering. Sporadic associations (red) are removed, and object boundaries interpolated (green) from neighbours.*

Constants $w_{1..4}$ are user defined weights which tune the influence of each of four bounded ($[0,1]$) heuristic functions. $\gamma(.)$ is the the Euclidean distance between the mean colours of the two regions in *CIELAB* space (normalised by division by $\sqrt{3}$). $\delta(.)$ is the spatial distance between the region centroids as a fraction of a maximum Δ. To speed up processing, we only examine candidates within a certain distance Δ; for our results $\Delta = 30$. $\alpha(.)$ is a ratio of the two regions' areas in pixels. $\sigma(.)$ is a linear conformal affine invariant shape similarity measure, computed between the two regions. Regions are first normalised to be of equal area (affecting uniform scale invariance). $\sigma(.)$ is then computed by taking Fourier descriptors of the angular description function [21] (translation invariant) of each region's boundary. Shape similarity is inversely to proportional to Euclidean distance between the magnitude vectors of the Fourier descriptors for both regions (disregarding phase for rotational invariance).

For a given frame t we examine mappings from $\mathcal{R}_{t-1}$ to R_t, and from $\mathcal{R}_{t+1}$ to R_t. Association is an iterative process. A count of pixels in R_t is initially established. At each iteration the 'best scoring' (1) association $b \in \mathcal{R}_{t\pm1}$ is associated with R_t, and the count of pixels for R_t decremented by the area of b. Iteration halts when the best association score falls below a threshold, or the count of pixels in R_t becomes close to or less than zero. It is therefore possible for no associations to be created; in such circumstances a feature appears or disappears in the video.

The process is repeated for each frame independently. The final set of associations for the sequence is taken to be the union of associations created for all frames.

Finally, associated regions are joined over time to create connected sub-volumes such as that in Figure 2c,d. These sub-volumes are broken into, possibly many, temporally convex *video objects*. Note we consider only the exterior boundary of these objects, disregarding 'holes' in a volume produced by other nested objects; these are represented by their own bounding surfaces. A property of the temporally convex representation is that two separate objects will merge to produce one novel object, and an object division will produce multiple novel objects (see Figures 2d,3b). This results in an object graph structure for each feature moving through the video, which simplifies processing in the filtering stage.

3.2.2. Filtering

Sporadic associations are often falsely created between regions due to noise. We have observed that associations maintained over short time intervals may be categorised as noise, and filter out these artefacts by examining the object graph structure.

Since new objects are created for every merge or divide encountered in a sub-volume, one can identify sporadic merges or divisions by searching the graph for short-lived objects. We specify a short-lived object as an object which exists for less than a quarter of a second (12 frames). This constant (as well as Δ from (1) may be adjusted according to the assumed maximum speed of objects in the video. Short-lived objects deemed to correspond to false associations and are removed by 'cutting' that object from the graph and filling the gap by extrapolating an intermediate object from one or both neighbours (see Figure 3c). A serendipitous effect of this process is that poorly segmented areas of the video exhibiting high incoherence, tend to merge to form one large coherent object. This is subsequently rendered as a single region, abstracting away detail which would otherwise flicker in the final animation.

Since each object is temporally convex, its boundary may be described by a continuous 2D surface (disregarding 'end caps'). By applying a low-pass filter in the temporal dimension to the fitted surface, we may smooth coarse temporal instabilities that may remain. However, rather than removing all instabilities we allow the user to define the scale of the low-pass filter. In this manner the user may chose to retain some of the noise present as a consequence of the animation process. We draw an analogy with the often desirable presence of film grain in a movie.

3.3. The Intermediate Representation

The result of the segmentation and association processes are a set of temporally convex objects, in a voxel representation. From this voxel volume we generate the intermediate representation (IR) to be passed to the back end, which consists of a series of 'stroke surface' patches and a supplementary database.

In our framework, the spatio-temporal locations of objects are represented in terms of their interfacing surfaces. This is preferable to storing each object in terms of its own bounding surface as surface information is not duplicated. This leads to a more compact, and more manipulable representation (which is useful later when we deform object boundaries, either for further fine smoothing, or to introduce temporal effects). When two objects abut in the video

volume, their interface may be represented in piecewise form by a number of surface patches (these may not necessarily be continuous). Each of these patches we term a 'stroke surface', and store the complete set of stroke surfaces for the video as one half of the IR. Each stroke surface holds an additional *winged edge structure* which contains two pointers corresponding to the two objects which it separates.

A supplementary database is maintained as the second half of the IR, containing one record per object in the video volume. This counterpart database is referenced by the pointers held in the stroke surfaces' winged edge structure. The database stores various attributes about each object, at each frame of its existence. At this stage we populate the database with the mean colour of the object at each instant, and a record of its temporal children and parents (thus encapsulating the object association graph).

3.4. Capturing internal detail

At this stage each feature with the video is stored in our representation only in terms of a series of linked, coherent spatio-temporal object boundaries and their interior colours. This can prove insufficient to render some artistic styles, since the interior detail that has been abstracted away often forms perceptual important visual cues (for example implying depth). We now describe how salient details such as these are reintroduced in a temporally coherent manner.

First, we fit a linear shading gradient to each object on a per frame basis. The gradient at time t over an object may be described as a triple $G_t = [g_0, g_1, \theta]$, where g_0 and g_1 are the start and end shading intensities respectively, and θ specifies the shading direction over the object. An optimal G_t is computed by a Nelder-Mead search [22] aiming to minimise the error $E[.]$:

$$E[G_t, F_t] = \frac{1}{|P|} \sum_{p \in P} \left(I(G_t) - F_t\right)^2 \qquad (2)$$

Where $I(G_t)$ is a image created from the gradient triple G_t, using the hue and saturation components of the object mean colour from the database and varying the luminance as defined by G_t. F_t is the video frame at time t, and P is the set of pixels inside the object region at time t. The application of this gradient alone, when rendering, can dramatically improve the sense of depth in an image.

Second, images of the original object in a small temporal window around t are differenced with the optimal $I(G)$ at those times; the result is a map containing the detail thus far abstracted away by our representation. We pass these maps through a salience filter (described in [2]) in which salience is deemed proportional to rarity. We form a local motion estimate for the object over the temporal window by assuming the object to be approximately planar, and so its motion relative to the camera to be well approximated by a homography. Object regions over the window are projected to the reference frame of the object at time t. A initial degenerate estimate of the homography is obtained by taking the 2nd order moments of the two regions. This estimate is then refined using a Levenburg-Marquadt iterative search to minimise mean square pixel error between the interiors of the regions. The computed salience maps are projected by homography to the reference frame at t, and averaged to form a final map. This results in the suppression of sporadic noise and reinforcement of persistent salient artefacts. We threshold the map, and apply morphological thinning. Stroke surface patches are then fitted around each disconnected artefact as before. The patches are then projected to their original reference frames via the inverse homographies used to generate the map.

The per frame shading gradient triple G and homographies are stored in the supplementary database. The homographies are of additional use later for advanced rotoscoping and stroke-based rendering. The new interior stroke surfaces are added to those of the existing representation, but with both sides of their winged edge structure set to point to the object in which they originated.

4. Back end: Rendering the Representation

We now describe the back end of the Video Paintbox responsible for rendering the IR. The rendering of objects is treated as two distinct tasks performed sequentially: 1) the rendering of an object's shaded interior region; 2) the rendering of the object's outline (often referred to as the *holding line* by animators) and any interior cue lines also present. This separation allows us to create many novel video effects, such allowing interior shading to spill outside of the holding lines.

Stroke surfaces are first duplicated to produce two sets; one for the interior shading stage and one for the line rendering stage. To render a particular frame at time t, the set of stroke surfaces embedded in the video volume $\Re^3 = [x, y, z]$ are intersected with the plane $z = t$. Any intersected surfaces are then rendered by their respective stages.

4.1. Rendering the interior

The intersection of plane and stroke surfaces produces a series of splines which are scan converted into a buffer. Bounded regions in this buffer are assigned an object pointer via the winged edge structure attached to each stroke surface. These regions correspond to the interiors of objects and may be rendered in a number of styles.

4.1.1. Cartoon style flat shading

Arguably the simplest style in which to render an object interior is to flat shade with the mean colour stored for the current frame; recall that this information was recorded in the supplementary database by the front end. However as objects divide or merge their mean colour can change significantly from frame to frame, causing unnatural rapid colour changes and flickering in the video (see the left hand skirting board in Figure 4). This is symptomatic of the general problem of assigning region attributes in a coherent way, and we draw upon our spatio-temporal representation

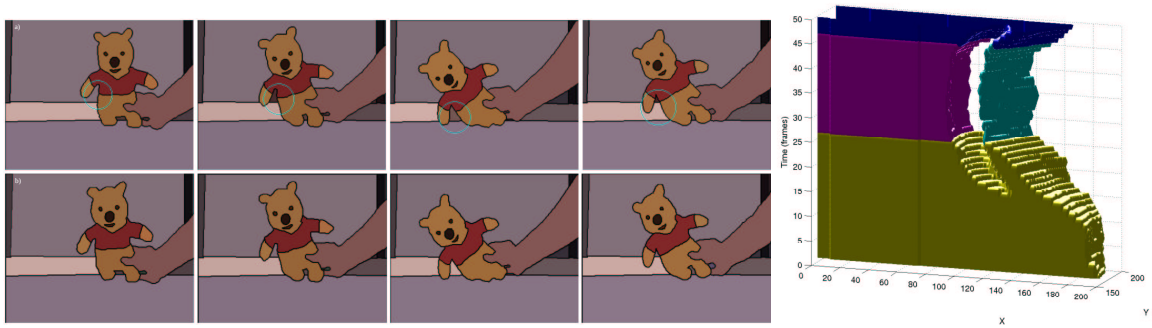

Figure 4: *Demonstrating the coherence of our cartoon style shading. Above: Temporal incoherence (highlighted in blue), in the form of flickering colour, is caused by shading on a per frame basis. Below: Our spatio-temporal representation allows us to mitigate against these incoherences by smoothing attributes, such as colour, over the video volume. Right: A volume visualisation of the left hand skirting board, comprised of a four associated video objects. Smoothing attributes with respect to this, and other volumes in the sequence, improves temporal coherence.*

to mitigate this incoherence. Recall that objects are associated via a graph structure. We may reconstruct the feature sub-volume in a small temporal window around the current frame using this graph. By averaging database attributes, such as colour, over the volume we can create a smooth transition of those attributes over time (even if objects appear disjoint in the current frame but connects at same other instant (in the past or future). Such coherence could not be obtained using the per frame sequential analysis performed by current NPR video methods.

We observed in Section 3.4 that the high abstraction level of a flat shaded video can be unappealing; artists often make use of shading and cue marks to add a sense of lighting and depth to a scene. We can augment the flat shaded regions by rendering the gradient shading attributes fitted earlier, smoothing the parameters in a similar manner to colour to ensure coherence (c.f. Figures 5d,e). Interior line cues may also be added by rendering the interior stroke surfaces of the object (Figure 5h) although strictly the rendering of such cues occurs later in the line rendering stage (Section 4.2).

4.1.2. Advanced Rotoscoping and Stroke based NPR

Rotoscoping is a technique pioneered in the 1970s, in which an animator manually traces over photographs or stills to given a stylised effect in a cartoon. In this sense, image-based NPR methods (for example stroke-based renderers [8, 9, 2] or adaptive textures[23]) may be considered a form of modern day rotoscoping. Recently 'Advanced Rotoscoping' tools have appeared in studios, allowing users to draw features at key-frames and interpolating between those key-frames automatically. This proves to be a valuable labour saving device.

We use the local motion estimate for video objects computed in Section 3.4 (stored in the supplementary database), to implement advanced rotoscoping through our framework. Users may draw a feature and attach it to a key-frame in the footage. The feature then moves with the local object reference frame to which it is attached. This facility is a valuable tool in adding personality to an animation, for example through the placement of expression on a face (Figure 5b). This is particularly important given that fine features such as facial expressions often do not survive the region segmentation and association process due to noise.

In a similar manner we are able to extend a variety of image-based NPR methods to video. NPR strokes (for example painterly strokes) or adaptive textures (for example Q-maps) are initially fixed upon objects, and subsequently move according to that object's local reference frame (Figure 5c). We have observed that, in the case of these NPR cues, the full homography is superfluous to needs and an affine approximation to the homography produces aesthetically superior results. To mitigate against stroke flicker, for example in colour due to video noise, individual strokes record persistent state of their various attributes. As each new frame is encountered, these values are updated based on a moving average (FIR filter) of their new and historic values; this effectively smooths their values over time.

4.1.3. Spatio-temporal effects

The representation of video features as a spatio-temporal volume allows us to produce novel temporal effects which would be otherwise tricky to produce on a per frame basis.

Fine scale temporal incoherences may remain present in object boundaries. Since adjacent objects are now represented in terms of their interface surfaces, temporally incoherent wobbles in those boundaries may be dampened by smoothing the surfaces in the z (temporal) plane. More interestingly though, we can also introduce controlled swimming and distortion effects by perturbing these surfaces; such an effect can lend a distinctive signature to an animation, see for example cartoons such as 'Roobarb and Custard'. Figure 5g demonstrates a sequence where we have imposed a periodic displacement function over stroke surface patches. A 2D parameterisation of each stroke surface patch is generated by projecting the patch orthogonally to an underlying 2D plane $\Re^2 = [s,t]$. Points on the patch are then translated in the direction of their surface normal $\hat{n}$

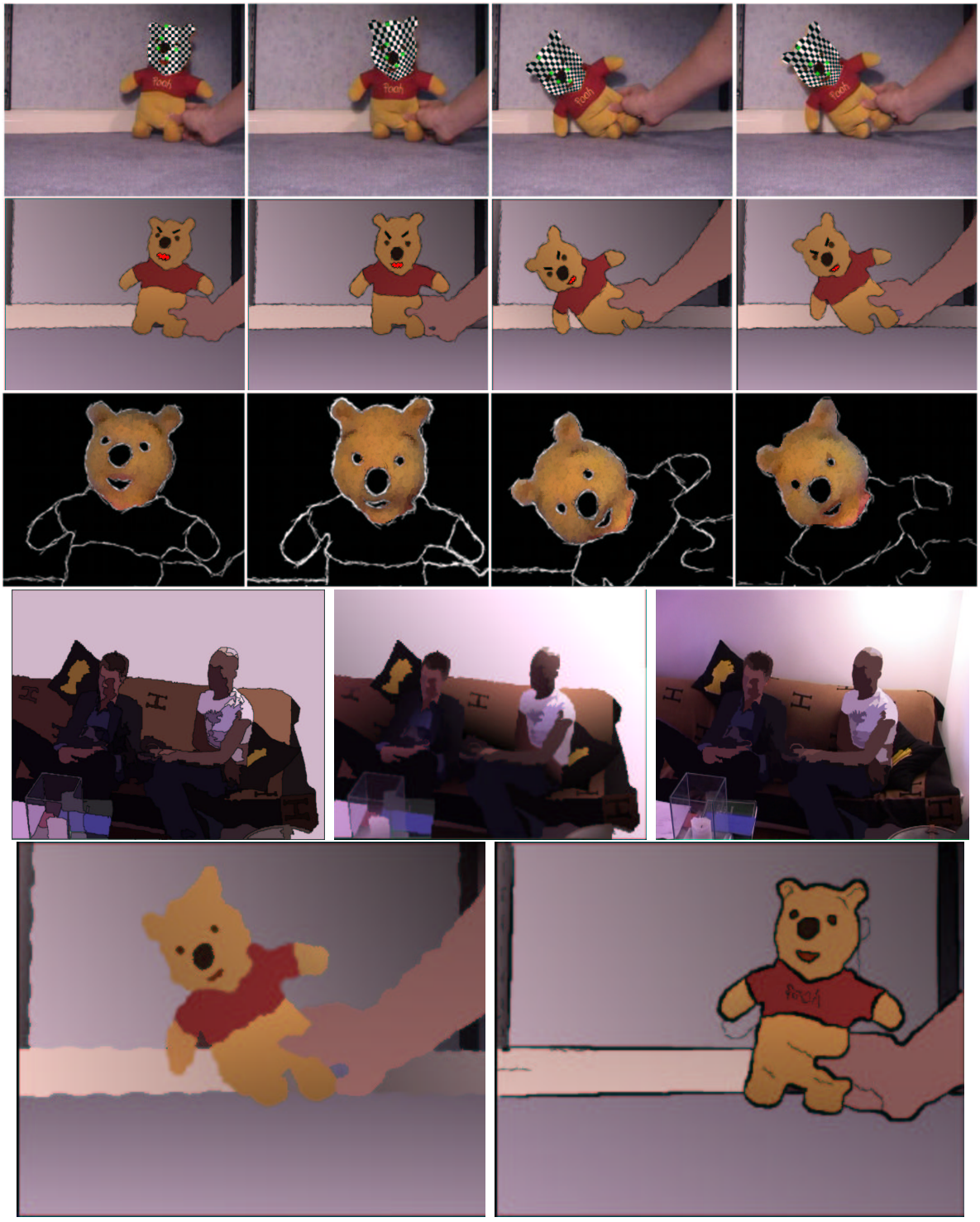

Figure 5: *(a) Demonstrating the local reference frame attached to each feature in the video, here the checker-board test image moves with the full homography. The green stars illustrate possible NPR stroke locations which can be seen to move coherently with each feature. (b) The framework has the facility for advanced rotoscoping operations, allowing the animator to easily add extra illustration to objects without having to re-sketch each frame. (c) An example of painterly (NPR stroke-based) rendering style; strokes move coherently with the tracked head from (a). Flat shaded (d) and gradient shaded (e) cartoon-style stills from the rendered* SOFA *sequence, (f) the animator may interactively define rendering styles for particular objects, in this case background objects are left photorealistic. (g) novel wobbling effects can be introduced by distorting the stroke surfaces (h) gradient shading and sketchy interior cue lines add a sense of depth and lighting to the final animation.*

according to some scalar functional $D(s,t)$, typically a sum of cosines:

$$D(s,t) = \hat{n}\frac{d(s,t)}{\int\int d(s,t)\,\delta s\delta t} \quad (3)$$

$$d(s,t) = k_{amp}\sum_{i=1}^{n} w_{i,1}\cos(w_{i,2}t + w_{i,3})\sum_{i=1}^{n} w_{i,4}\cos(w_{i,5}s + w_{i,6})$$

where k_{amp} controls the amplitude of the effect on the object borders. The $(6xn)$ matrix w is typically a constant initialised once per patch via a pseudo-random number generator. We can also produce interesting effects using morphological operations. For example, an erosion on interior regions has been used to allow a synthetic canvas to show through in Figure 6. This is complemented by a translucent watercolour wash texture that has been overlaid on objects and moves with the local reference frame as described in Section 4.1.2. More exotic deformations can be produced by passing the volume through various transfer functions; simple translations give an appealing 'shaded outside of the lines' visual effect, whilst a more general free form volume deformation can provide smooth variations in the shape of the interior shading over time.

4.2. Rendering the holding and interior lines

Recall the surface intersection operation in which we determine the stroke surfaces active in the current frame. When the holding and interior lines for the object are rendered, the splines resulting from the intersection are used to form long, flowing strokes which are stylised according to a chosen procedural NPR brush model (for example, sumi-e [24], graphite [25] or sticky paint [26]). This produces attractive long strokes which move coherently throughout the video. Some brush models require a stochastic element to simulate effects such as brush bristle texture; without due care this can cause swimming in the animation. We observe that in these cases only the illusion of randomness is required for aesthetics, and seeding the pseudo-random number generator with a hash of the unique ID of a stroke surface is a convenient way of preserving stroke behaviour for the entire stroke surface patch.

The surface patches that form these strokes may be manipulated prior to rendering to produce a range of effects. In Section 4.1.3 we describe a coherent wobble effect that may also be applied to the holding and interior line stroke surfaces. However, effects specific to line rendering may also be applied to produce novel styles as we now explain.

4.2.1. Sketchy stroke placement

Artists often produce sketchy effects by compositing several light, inaccurate strokes on canvas which merge to approximate the boundary of the form they wish to represent. We may apply a similar, coherent, sketchy effect to video using a similar technique applied to our stroke surface patches. Stroke surface patches are first shattered into smaller Catmull-Rom [27] cubic patches (Figure 6). The spatial and temporal intervals for this fragmentation are two of the many user parameters through which the appearance of the final animation may be influenced. The small spatial intervals create many small sketchy strokes, and very large temporal intervals can create interesting time lag effects. Each of these cubic patches becomes a stroke in its own right when later intersected and rendered.

The cubic patches are each subjected to a small random affine transformation M to introduce small inaccuracies in the positioning of patches; Figure 6b gives a visualisation of the perturbed patches.

$$M = T(\tau)T(-c)S(\sigma)R(\rho)T(c) \quad (4)$$

where $T(.)$, $S(.)$, and $R(.)$ are the standard 3D translation, scale and rotation matrices, c is the centroid of the patch, and τ,σ,ρ are small normal variates (the user is given control over these limits of these variates). The resulting cubic patches are intersected as described, and rendered as fine strokes to yield a sketchy effect. The coherence of a stroke is guaranteed over its temporal extent, since it arises from a smooth cubic patch embedded in the spatio-temporal volume. Finally, the density of sketching may be increased by further duplicating the stroke surfaces prior processing surfaces in the manner described.

It is a matter of artistic taste whether sketch lines should be re-sketched at a constant rate for the duration of the video, or whether they should only be re-sketched as the object moves. Currently our framework subscribes to the former point of view, but can be easily modified to produce the latter. By substituting the normal variates for values picked from a noise-box parameterised by location, sketchy strokes will only appear to be redrawn when the location of a stroke surface changes i.e. during object motion.

4.2.2. Other line styles

Rather than fragmenting the stroke surfaces for a sketchy effect, long flowing strokes may be painted along the edge of a stroke surface patch. We do not concern ourselves unduly with the appearance of the stroke; we defer the problem of artistic media emulation to the literature, and are concerned primarily with stroke placement rather than stroke rendering. However we have found that interesting results can be obtained by varying line weight according to some transfer function. Some good transfer functions are to vary a surface's line weight proportional to the maximum of the speed of the two objects it bounds; according to the intensity gradient between the two objects; according to the maximum of the area of the two objects; and according to the salience map local to the stroke surface. The latter suggestion helps to mitigate artefacts produced when a feature has been over-segmented, leading to, say, a face broken into two features divided by a thick black line. If there is little evidence for a salient edge in the image at that boundary, then the stroke may be omitted. In this case a modification would be required to store image salience information in the supplementary database. Note that interior 'cue' lines generated in Section 3.4 are naturally accommodated into this framework. One might introduce a precondition into the rendering process to selectively render only interior cues, or only exterior cues (these may be

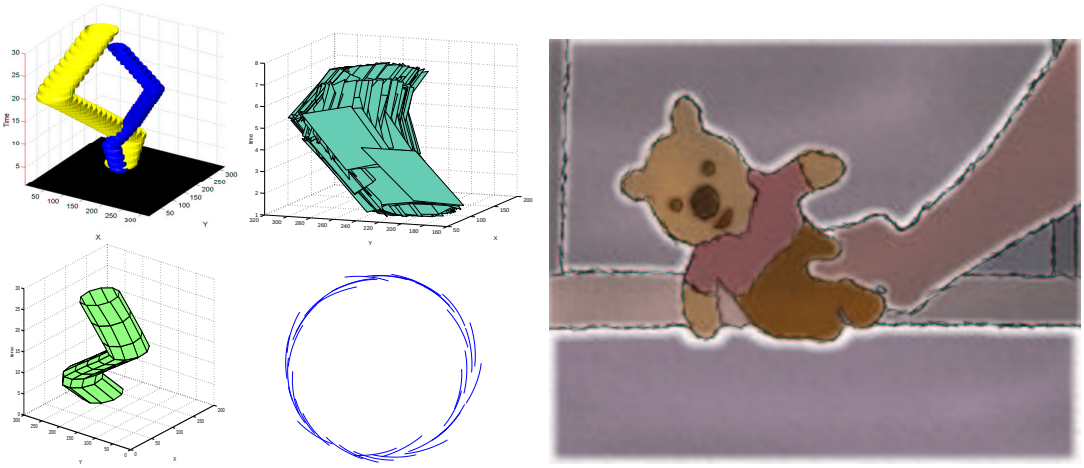

Figure 6: *Left: A test sequence of bouncing spheres is segmented (top left). We visualise the stroke surface about one sphere, prior to (bottom left), and following (top right) surface shattering; we approximate surfaces here with piecewise linear patches. When shattered patches are intersected they form a coherent sketchy effect (bottom right). Right: A still from a coherent animation produced from the* BEAR *footage. Here the sketch effect on the holding line has been combined with a watercolour wash on the interior.*

easily distinguished by examining the winged edge structure of the stroke surface).

4.3. Interactive control of the composition

Some features may be over-segmented in the video, producing two distinct graphs of video objects where one would suffice. This situation arises when the initial segmentation algorithm consistently over-segments a feature over several video frames, often due to local illumination variance. We provide an interactive facility for the user to merge such objects if required. Objects are linked by point-and-click mouse operations in a single frame, and those changes propagated naturally down to all other frames the object exists in (as objects are spatio-temporal in their extent). The user is given two options: 1) to semantically link the objects; 2) to physically link the objects. In the former case, objects are kept distinct in our representation, but the association graph is modified so that any colour smoothing etc. occurs over all linked objects (see Figure 7b). In the latter case, the linked objects are deleted from the representation and replaced by a single object which is the union of the linked objects (see Figure 7a). This interactive approach can be used to control focus in the composition, by coarsening the scale of chosen region. In the future we would like to drive this process automatically using a perceptual model.

Finally we allow the user to selectively chose rendering parameters for specific objects. Again, such interaction requires only a couple of mouse clicks to modify parameters for objects which may extend through the video for many frames. In Figure 5f we show an example where the animator desires people to appear photorealistic, but within a cartoon shaded environment; reminiscent of the A-Ha music video “Take On Me” [Barron, 1985].

5. Discussion and Concluding Remarks

We have proposed a novel framework for the rendering of non-photorealistic animations from video sequences, in the form of a “Video Paintbox”. Source video is initially segmented via standard Computer Vision techniques, and transformed into a spatio-temporal intermediate representation. This representation may then be manipulated and re-rendered in a wide gamut artistic styles. The user exerts control over this process at a high level by varying parameters on the Video paintbox. We have demonstrated the value of our high-level approach by overcoming problems of temporal incoherence using our spatio-temporal stroke surface representation (for example Figure 4), and through the production of several temporally coherent non-photorealistic animations.

We have already alluded to the possibility of future improvements, with regard to composition. Specifically we would like to extend the technique of scale-space rendering to video, driven by some automated salience measure. This will prove more complicated than simply extending 2D scale space concepts to 3D because large volumes do not necessarily correspond to important features (a large region persisting over a short time might be considered noise, while a small region persisting for a correspondingly long time might be salient).

An area worthy of further investigation is the extremely compact nature of the intermediate representation. Figure 8 summarises details of a brief comparative investigation, contrasting the storage requirements of the IR with those of leading video compression technologies. Approximately 150Kb were required to store 100 frames of video, which implies interesting applications in low band-width video transmission. The caveat is of course that the video must be abstracted and stylised in a cel animated fashion; how-

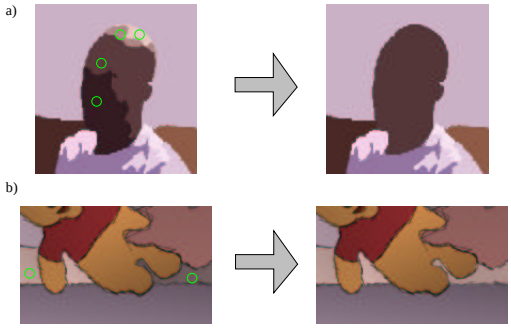

Figure 7: *Users may create manual associations between objects to tune the segmentation or composition. (a) User creates a physical link between over-segmented objects, a single object replaces four. (b) User creates a semantic link between two objects. The objects remain distinct, but associations are created in the object graphs; during rendering attributes are blended between regions to maintain temporal coherence.*

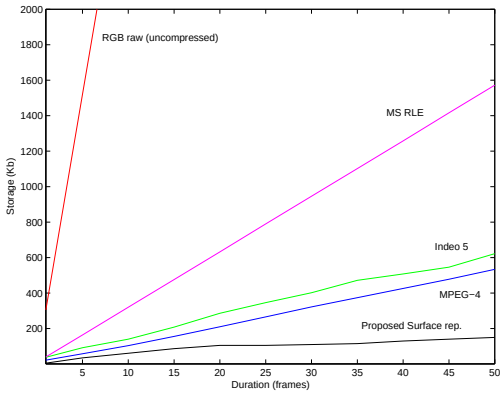

Figure 8: *Demonstrating the relatively low storage requirements of the surface representation. Comparison uses up to 50 frames of the gradient shaded cartoon* BEAR *sequence.*

ever the ability to select the artistic style of the output at the client-side is an interesting concept.

The ability to process video through our framework is predicated on the ability to coherently segment the original footage. Given that the general segmentation problem remains unsolved by the Vision community, there exist examples of video that we can not reliably process (examples include cluttered crowd scenes). However existing optical flow based NPR methods also do not work well in all circumstances e.g. during occlusion, and we believe the proposal of a novel solution framework to be a valuable contribution to NPR. We hope to perform objective comparative tests between our framework and these optical flow based methods to further clarify this claim. Finally, we hope to merge our Video Paintbox system with earlier work [1, 2] capable of visually rendering the presence of motion to fulfil our eventual goal – the automated generation of cartoons from video.

We observed in Section 1 that existing automated NPR video techniques operated at temporally low level, processing frames sequentially and considering only the previous frame in the sequence. These techniques also operate at a spatially low-level, driving the movements of individual strokes via optical flow. By contrast, our framework is centred around the paradigm of high-level spatio-temporal analysis. We believe the most productive avenues for future research will not be in incremental refinements to the current system, but rather will examine alternative uses for higher-level spatio-temporal analysis of video with applications to NPR.

A selection of rendered video sequences are available on-line at **http://www.cs.bath.ac.uk/∼vision/cartoon**.

References

1. J. P. Collomosse, D. Rowntree, and P. M. Hall, “Cartoon-style rendering of motion from video”, in *Vision, Video and Graphics*, pp. 117–124, (July 2003).
2. J. P. Collomosse, D. Rowntree, and P. M. Hall, “Video analysis for cartoon-like special effects.” BMVC 2003, in press.
3. S. Green, D. Salesin, S. Schofield, A. Hertzmann, and P. Litwinowicz, “Non-photorealistic rendering”, *SIGGRAPH ’99 Non-Photorealistic Rendering Course Notes*, (1999).
4. K. Koffka, *Principles of Gestalt Psychology*. New York: Harcourt Brace, (1935).
5. E. Wong and N. Weisstein, “Flicker induces depth: Spatial and temporal factors in the perceptual segregation of flickering and nonflickering regions in depth”, *Perception & Psycophysics*, **35**(3), pp. 229–236 (1984).
6. A. Hertzmann and K. Perlin, “Painterly rendering for video and interaction”, in *Proceedings NPAR Symposium*, pp. 7–12, (2000).
7. M. Haggerty, “Almost automatic computer painting”, *IEEE Computer Graphics and Applications*, pp. 11–12 (1991).
8. P. Litwinowicz, “Processing images and video for an impressionist effect”, in *Proceedings Computer Graphics (ACM SIGGRAPH)*, pp. 407–414, (1997).
9. A. Hertzmann, “Painterly rendering with curved brush strokes of multiple sizes”, in *Proceedings Computer Graphics (ACM SIGGRAPH)*, pp. 453–460, (1998).
10. J. P. Collomosse and P. M. Hall, “Painterly rendering using image salience”, in *Proceedings 20th Eurographics UK Conference*, pp. 122–128, (June 2002).
11. B. Galvin, B. McCane, K. Novins, D. Mason, and S. Mills, “Recovering motion fields: An evaluation of

eight optical flow algorithms", in *Proceedings BMVC*, vol. 1, pp. 195–204, (1998).

12. A. Santella and D. DeCarlo, "Abstracted painterly renderings using eye-tracking data", in *Proceedings NPAR Symposium*, (2002).

13. B. et al, "Scale-space trees and applications as filters...", in *BMVC*, (1999).

14. H. Zhang, A. Kankanhalli, and S. W. Smoliar, "Automatic partitioning of full-motion video", in *Mutlimedia Systems*, vol. 1, (1995).

15. M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*. P.W.S. Publishing, (1999).

16. V. Rehrmann, *Stabile, echtzeitfähige Farbbildauswertung*. PhD thesis, Dept. Computer Science, University of Koblenz, Germany, (1994).

17. C. Christoudias, B. Georgescu, and P. Meer, "Synergism in low level vision", in *Conference of Pattern Recognition*, (Quebec City, Canada), (August 2001).

18. D. Comanicu and P. Meer, "Mean shift: A robust approach toward feature space analysis", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2002).

19. Y. Deng and B. S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2001).

20. M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking", *Int. Journal of Computer Vision*, **29**(1), pp. 5–28 (1998).

21. R. L. Cosgriff, "Identification of shape", Tech. Rep. 820-11, ASTIA AD 254792, Ohio State Univ. Research Foundation, Columbus, Ohio USA, (1960).

22. J. Nelder and R. Mead, "A simplex method for function minimization", *Computer Journal*, **7**, pp. 308–313 (1965).

23. P. Hall, "Non-photorealistic rendering by Q–mapping", *Computer Graphics Forum*, **1**(18), pp. 27–39 (1999).

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

25. M. C. Sousa and J. W. Buchanan, "Observational models of graphite pencil materials", *Computer Graphics Forum*, **1**(19), pp. 27–49 (2000).

26. T. Cockshott, J. Patterson, and D. England, "Modelling the texture of paint", *Computer Graphics Forum*, **11**(3), pp. 217–226 (1992).

27. J. Foley, A. van Dam, S. Feiner, and J. Hughes, *Computer Graphics*. Addison Wesley, 2nd edition, reissued ed., (1995).