Available online at www.sciencedirect.com
SCIENCE @ DIRECT®

Computers & Graphics 29 (2005) 862–870

COMPUTERS & GRAPHICS

www.elsevier.com/locate/cag

Video Paintbox: The fine art of video painting

J.P. Collomosse*, P.M. Hall

Department of Computer Science, University of Bath, Claverton Down, Bath, England BA2 7AY, UK

Abstract

We present the Video Paintbox; a novel system capable of transforming video into stylised animations. Our system solves the problem of temporally coherent painting for a wide class of video, and is able to introduce cartoon-like motion emphasis cues. Coherent painting in video is a long-standing problem that is associated with the difficulties in placing, orienting, and sizing paint strokes over time. Our solution allows removal of stroke flicker, allowing animators control over any stylisation and incoherence they might care to re-introduce. Furthermore, the animator can choose to selectively paint certain objects, leaving photorealistic video elsewhere. Many common motion cues used by animators, such as streak-lines, squash-and-stretch, and anticipated movement may be integrated with either real or painted footage. The motion cues depend on a robust analysis of video that allows for effects such as camera motion and occlusion, and which is able to automatically build articulated dolls complete with kinematic properties. We describe our Video Paintbox, discuss animator controls, and illustrate with examples.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: Video stylization; Cartoon; Non-photorealistic rendering

1. Introduction

Processing real-world images into artwork is a significant and demanding area within non-photorealistic rendering (NPR). The number of high-quality papers addressing this issue is growing rapidly in the literature, and shrink-wrap software gives a public platform. The central demand made on all automated techniques is that of extracting information from images useful for artistic synthesis. To date the majority of such image-based NPR techniques have focused on still images; only a few address moving images. Furthermore, most perform image analysis using general purpose *early vision* methods such as edge detection and optical flow. A trend is beginning toward using *mid-level* and even *high-level* vision methods. This paper contributes to that trend.

In this paper we describe a "Video Paintbox" that accepts real-world video as input, and outputs either still or moving images that have been resynthesised into some non-photorealistic style. We offer the following contributions to video painting:

- We have solved the long-standing and important problem of frame-to-frame coherence for a wide class of input video.
- Uniquely, we introduce motion emphasis cues using all the effects employed by traditional animators: augmentation cues such as streak-lines, deformation cues such as squash-and-stretch, and timing cues such as anticipation (Fig. 1).

We have carefully balanced user interaction to provide a high degree of automation in the animation process, yet allowing the user to express control at a high-level over key stylistic parameters. Our system relies on modern,

*Corresponding author.
E-mail address: jpc@cs.bath.ac.uk (J.P. Collomosse).

doi:10.1016/j.cag.2005.09.003

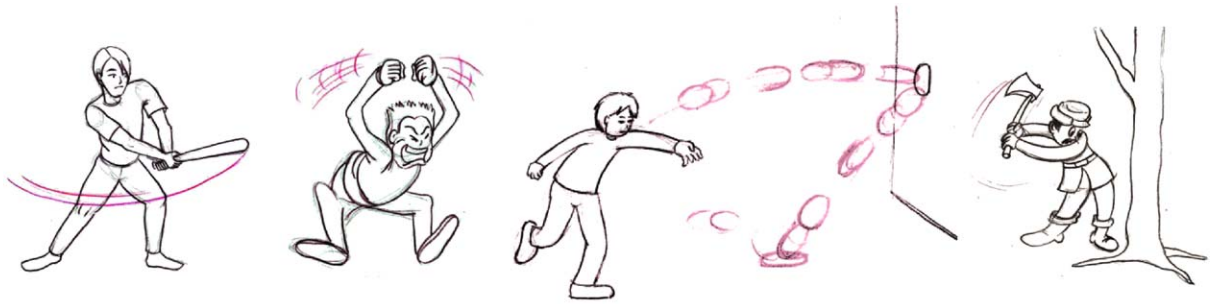

Fig. 1. Motion cues drawn by commercial animators include augmentation cues such as streak-lines, deformations such as squash-and-stretch, and dynamic cues such as anticipation.

mid-level computer vision methods, some of which we have developed specifically for this application and make a contribution to Computer Vision. Details of these contributions are reported elsewhere [1], the aim here is for the first time to describe the full architecture of our Video Paintbox.

We will explain the Video Paintbox and how it is designed so that:

- The animator has fine control over the output style, ranging from cartoon flat-shading, to painterly styles such as watercolour—all within a single framework. The quantity and quality of temporal incoherence is under animator control.
- We can subject different parts of our input video to different styles, for example we can choose to leave some areas of video as raw footage, while artistically rendering specific objects.
- Automated rotoscoping is possible, using the same sub-system that controls temporal coherence. In fact, rotoscoping and painting are unified in our approach.
- We can output still images and video exhibiting commonly used animation cues, for example deformation and streak-lines. Other systems can output high-quality painterly stills, but focus upon only visual stylisation of objects. Uniquely, we also address the equally important issue of motion emphasis.

We begin by describing the context within which this work rests, and describe our architecture in Section 2. We conclude with a brief discussion in Section 3.

1.1. Related work

Our work is aligned with image-based NPR, the branch of NPR which aims to synthesise artwork from images. Research in the area is strong and diverse. A number of techniques have been developed for transforming photographs into sketches [2] and paintings [3–5]. Unfortunately these static techniques do not generalise easily to video processing, resulting in a distracting temporal incoherence (flicker) if frames are painted independently.

Creating animation from real-world footage is by no means a new idea. Rotoscoping, drawing over real footage, helped Disney produce "Snow White" in the 1930s. Some labour can be saved using modern in-betweening methods; animators working on the film "Waking Life" (Fox Searchlight, 2001) drew around objects every 10 or 20 frames. Similar systems have been produced for line-art cartoons [6,7].

Litwinowicz [8] was the first to address the problem of video NPR in a fully automatic manner. Brush strokes painted upon the first frame are translated from frame to frame in accordance with optical flow motion vectors estimated between frames. A similar painterly technique using optical flow was later proposed by Kovacs and Sziranyi [9]. Hertzmann and Perlin [10] use differences between consecutive frames of video, re-painting only those areas which have changed above a global (user-defined) threshold. Whilst these methods can produce impressive painterly video, they do not fully control flicker. Errors present in the estimated motion field quickly accumulate and propagate to subsequent frames resulting in increasing temporal incoherence which then requires exhaustive manual correction [11]. As Litwinowicz points out [8], temporal coherence is possible by associating strokes with objects in the video. These observations motivate us to use Computer Vision techniques to parse objects from the video.

Addressing the issue of temporal coherence is just one aspect of our Video Paintbox. Motion emphasis plays a pivotal role in any animation. However, whilst existing NPR video methods seek to mitigate against the effects of motion for the purposes of coherence, the literature is relatively sparse concerning the emphasis and rendering of motion within video. In an early paper, Lasseter [12] articulates many of the motion emphasis techniques commonly used by animators, though proposes no algorithm solutions. Techniques such as streak-lines, object deformation and anticipation are all discussed. Streak-lines have been studied by Strothotte et al. [2] and Hsu and Lee [13]. In both studies streak-lines are generated via user-interactive processes. Recent work in 3D animation applies a squash-and-stretch effect to spheres and cylinders in object space prior to ray-tracing [14]. Also in object-space, Li et al. [15] allow users to adjust trajectories of objects to stylise their motion via an interactive process. To the best of our knowledge, the embellishment of real-world video with motion emphasis cues is a unique contribution of our Video Paintbox.

We did not set out to produce a fully automated system—the vision problem precludes that (there is no general method to segment all images into semantically meaningful parts). We adopt the view that the burden of correcting faulty output produced by Vision algorithms should be kept as small as possible, we should try to free the animator so they interact with the system to control style of output, to choose which objects are to be affected by motion emphasis cues, and so on.

2. System architecture

Our system architecture comprises three main components. The first is the *computer vision* component—responsible for measurement and analysis of the source video. The vision component produces an *intermediate representation* (IR); this is a database that models a variety of aspects of the video. Data stored within this representation are used by the *computer graphics* component to re-synthesise the video in a non-photorealistic style (Fig. 2).

The IR is in fact a suite of integrated representations, forming two main groups that reflect the two main problems we have addressed. To render coherently we partition the video into spatio-temporal volumes that are separated by spatio-temporal surfaces (Section 2.2). For motion emphasis cues we are interested in object trajectories, which are spatio-temporal curves (Section 2.1).

The animator influences the output by assisting the Vision component—drawing a round objects of interest, for example—and by specifying the types of artistic effect desired, such as watercolour or "squash-and-stretch". User parameters for each effect may also be specified; for example one "squash-and-stretch" parameter that may be varied is the apparent elasticity of an object. We will now elaborate on the modes of interaction as we expand upon the "motion emphasis" and "coherent rendering" aspects of the Video Paintbox in turn. We will conclude our description with an account of how output from each of the two halves are integrated into a final stylised animation.

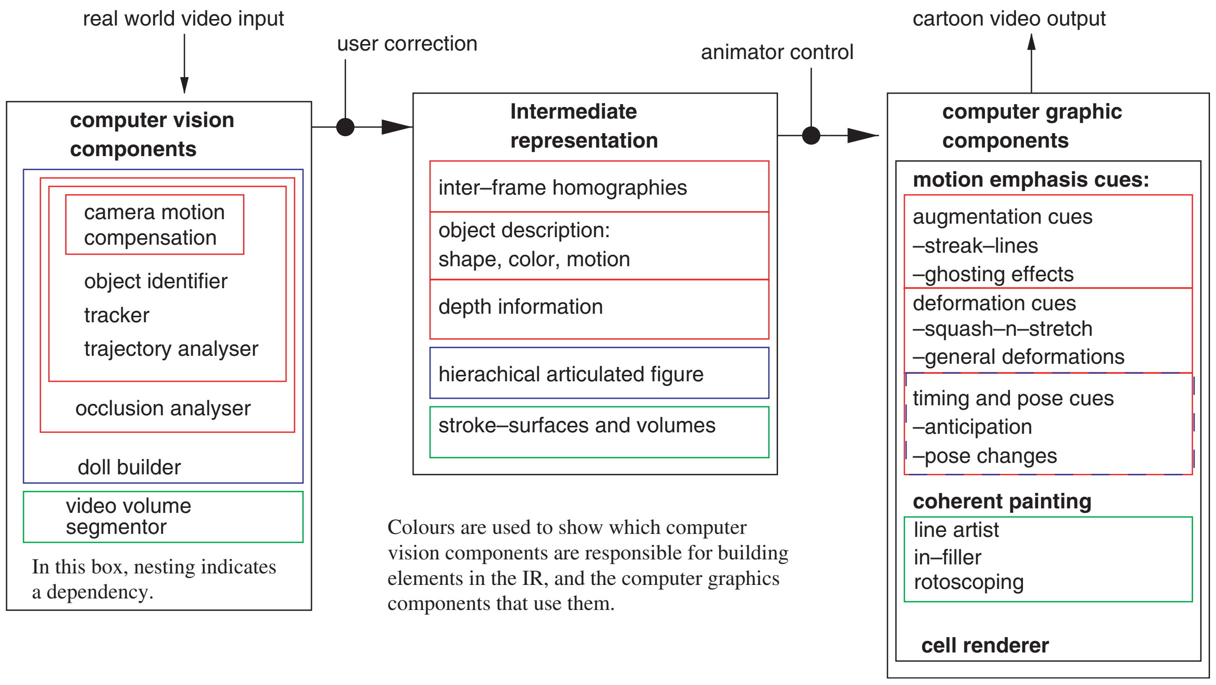

Fig. 2. An overview of our Video Paintbox, showing the three main components, their major contents, and the place animator control is exerted. The vision component processes real video input to make the intermediate representation, which is used by the graphics component for rendering.

2.1. Motion emphasis cues

We are able to reproduce many of the common motion emphasis cues described by Lasseter [12], and as used by animators [18] (Fig. 1). We recognise three main groups of motion emphasis cue: *augmentation cues* that add some painted artifact, such as streak and ghosting lines, to depict motion; *deformation cues* that alter the shape of objects, for example classical "squash-and-stretch"; and *dynamic cues* that alter pose and timing, for example motion anticipation and exaggeration.

The nature of the motions we wish to emphasise often demands a moving camera to capture them, so necessitating a camera (ego) motion compensation step as the first step in our system. Ego-motion determination allows the camera to move, to pan, and even compensates for camera wobble. We automatically compensate for such motion using similar methods to those used to stitch photographs into panoramas. The interested reader is referred elsewhere for details [18].

Motion emphasis cues require that we track objects—the motion of the camera is removed after tracking. There is no reason to assume the animator wants special effects applied to every object in the video. The animator identifies interesting objects by drawing loosely around them, the loose curve is automatically "shrink wrapped" to make a tightly fitting polygon around the object [17]. In this way we make a template of the object to be tracked.

We track objects using a Kalman filter, which enables us to predict their trajectory from one frame to the next. This prediction is affirmed and/or corrected by searching for the template in the future frame. If we find the object, but not in the predicted position, then we assume a collision has occurred and are able to estimate both the time of the event and the collision plane. If we find the object in the predicted position, but it is obscured in part or in whole, then we assume the object is passing behind some other body and use this to infer relative depth between them. Again we refer interested readers elsewhere for technical details [1].

The essential point here is that the Vision component is able to track objects of interest to the animator, measuring their *pose* which comprises position, velocity and acceleration relative to the world, in each frame of the video. The pose of objects changes in time, and the consequent *pose trajectories* are a key component of our IR. In addition to pose trajectories, we also deduce relative depth (via occlusions) and object collision events (detected when tracked motion poorly correlate with predicted motion). This is enough information to begin rendering the motion emphasis cues.

2.1.1. Augmentation cues: streak-lines and ghosting

Streak-lines are drawn by animators as long, flowing lines that are intended to represent the motion of objects over an extended period of time. Streak-lines trace object motion, and an intuitive first step to producing such lines might be to visualise optical flow. However, this is infeasible for two main reasons: (1) In practice, optical flow algorithms operate over periods of time that are much shorter than covered by typical streak-lines, they are susceptible to noise, and they fail at occlusions. (2) In principle, streak-lines do not trace the motion of a single point, but depict the overall "sense-of-motion" of an object. This is especially obvious in the case of rotating objects where the trailing edge often changes shape (Fig. 3).

We paint streak-lines by tracing what we call *correspondence trails*. These are formed by relating points on trailing edges from one frame to the next. A *trailing edge* comprises points on an object's boundary whose outward normal points in more or less the opposite direction to its motion. Since the shape, and even the topology, of a trailing edge varies as a function of motion, we create correspondences between edge points using a simple approach that works well in practice—a linear conformal affine mapping is sought to align edges in successive frames, and points along both edges are corresponded on a nearest neighbour basis.

Correspondence trails are fragmented where they are discontinuous and smoothed where they are continuous. Typically we will over produce correspondence trails and must discard many of them to avoid a confusing abundance of streak lines. We use a greedy algorithm to select the correspondence trails used to form streak-lines

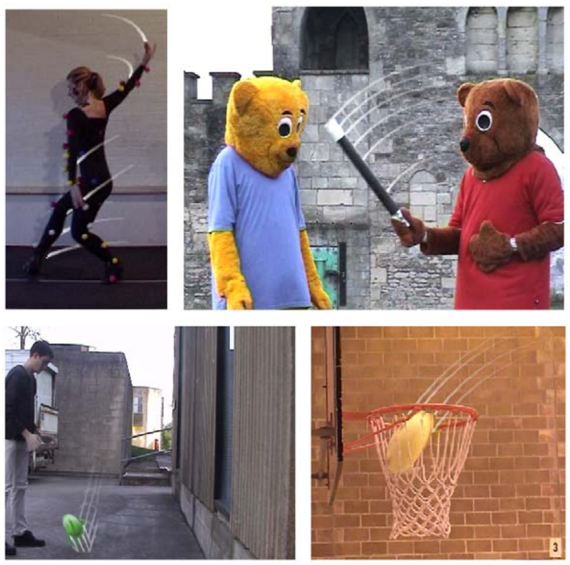

Fig. 3. Streak-lines depict the path of the objects they trail, while ghosting effects highlight the object's trailing edge. These cues can "wrap around corners" at collisions, and honour their object's occlusions.

for the animation. This algorithm is effectively a search to maximise a heuristic measure for optimal streak-line placement, distilled after experimentation and consultation with animators. We prefer to fix streak-lines to convex parts of the object that move rapidly and space them apart, so they do not interfere with one another. A preference for long, flowing streak-lines is also encoded into the heuristic measure (see [18]).

A streak-line is rendered starting in some frame and moving backwards in time. We could potentially render streak-lines using any icon we choose, but typically render to a simple disc whose radius and opacity both decay in time, under animator control. Ghosting effects can be synthesised by rendering the trailing edge, typically at uniformly spaced instants in time so that faster moving objects have more widely spaced ghost lines.

2.1.2. *Deformation cues: squash-and-stretch and general deformations*

Animators use deformation cues both as a general motion emphasis technique, and often also to depict inertia or drag. We rely on analysis of the trajectory of object centroid, in the camera-motion compensated sequence. The trajectory is broken into smooth sections just as for correspondence trails. However, we must now distinguish between *collisions* and predictable discontinuities. For example, the changes in motion due to the harmonic motion of a metronome are predictable discontinuities but the bounce of a ball on a wall is not.

We erect a curvilinear basis along the centroid trajectory. A point in this basis is located by two parameters, one the distance along the trajectory and a second which is the perpendicular distance to it. All deformation cues are generated by warping the object's image within this curvilinear basis.

Squash-and-stretch effects are generated using an area-preserving differential scale within curvilinear space. This produces smoother, more aesthetically appealing deformations than simply stretching the object along a linear axis tangential to the motion (Fig. 4, bottom). More general deformations are possible. By performing a non-linear warp in which each pixel is shifted along the abscissa in curvilinear space (i.e. the direction of motion) as a function of its motion coefficients, we can ascribe a "rubbery" quality to objects so that they appear to bend. By shifting pixels in proportion to acceleration magnitude we can cue on inertia, and create a simulated "drag" effect by cuing on speed (Fig. 4, top).

In cases where a collision event has been recorded, we can exchange the standard curvilinear basis for an "impact" basis. To generate the latter we estimate the exact point and time of the collision by extrapolating and intersecting centroid trajectories before and after the collision. By assuming equal angles of incidence and deflection we can compute the collision plane, and hence set up an "impact" basis in which the object deforms. This adds realism to the squash-and-stretch effect, allowing objects to create cartoon-like impact cues via *squashing*. A smooth transition between basis sets is effected by interpolating the vector fields induced by the deformations.

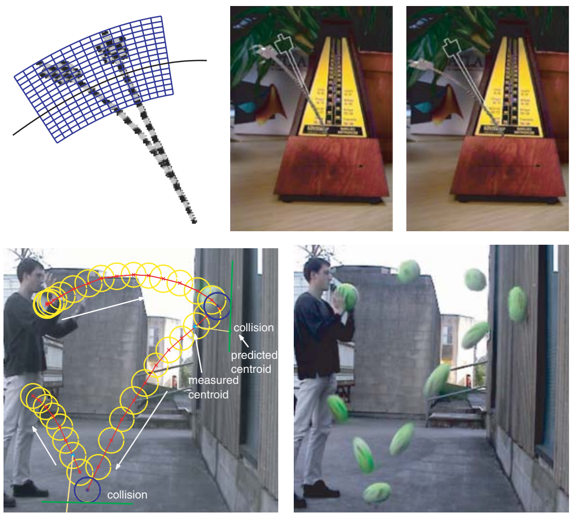

Fig. 4. Top: All deformations occur within a local curvilinear basis frame (blue). A non-linear warping in this space using a functional based on velocity (top left) and acceleration (top right) shown at the same point in time; original beater position in white. Bottom: A time-lapsed image demonstrating "squash-and-stretch". We infer the collision plane (green) and location of ball at impact (blue), and well as partitioning the trajectory into smooth sections.

2.1.3. *Dynamic cues: anticipation and motion exaggeration*

Dynamic cues involve a change in the way an object behaves; in general this involves directly affecting the objects pose trajectory. This is to be contrasted with all motion effects previously discussed, which are driven by the pose trajectories of objects but do not modify those trajectories. This behaviour—using the original pose trajectory to generate a new pose trajectory—makes dynamic cues the most complex of the motion cues to generate. Here we briefly discuss how we produce two examples of dynamic cue: anticipation and motion exaggeration, illustrated in Fig. 5.

Our needs to manipulate the pose of object demands that a motion model be inferred—the parameter space for this model is the space in which the pose trajectory lies. We are able to automatically infer a hierarchical articulated model of a tracked object, say a walking man (Fig. 5, right) by analysing the motion of tracked component features (for example, limbs) in the camera

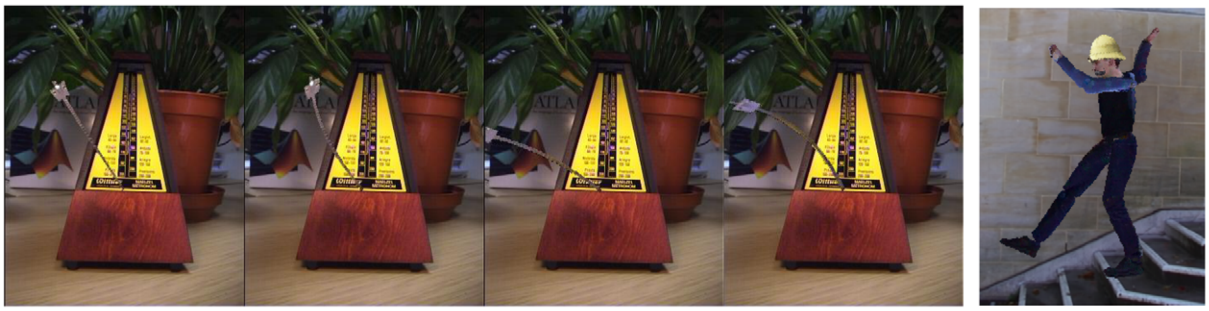

Fig. 5. Left: An animation of a metronome exhibiting anticipation, or “snap”, as it changes direction. The “drag” deformation of Section 2.1.2 has also been incorporated. Right: Motion exaggeration applied to human gait.

plane. The pose trajectory comprises a vector of local inter-joint angles and the relationship of the object to the camera reference frame.

Animators often cue the onset of motion using *anticipation* (sometimes referred to as “snap”). Consider a character about to run—the animator may draw that character briefly recoiling in the opposite direction, implying the storing of energy to be unleashed into a sprint. This clearly affects the pose trajectory, and we implement such cues by “cutting” a piece of the original pose trajectory and “pasting” in a modified piece. The system can select local points in the pose trajectory where anticipation might be appropriate, typically these points are discontinuities or extremities in acceleration. Four user parameters are available to the animator such as delay, magnitude of emphasis, and other timing values, to modify the shape of the novel pose trajectory to be inserted (Fig. 6 illustrates).

Motion exaggeration involves altering the pose trajectory more globally. This is well illustrated by the motion of a walking person, which creates a cyclic pose trajectory. Scaling the poses in this cycle away from their mean exaggerates the characteristic motion of the walker (Fig. 5, right). Our system allows additional constraints to be imposed on features over specified temporal windows. The animator may use these constraints to ensure that feet, for example, do not appear to slide over the floor.

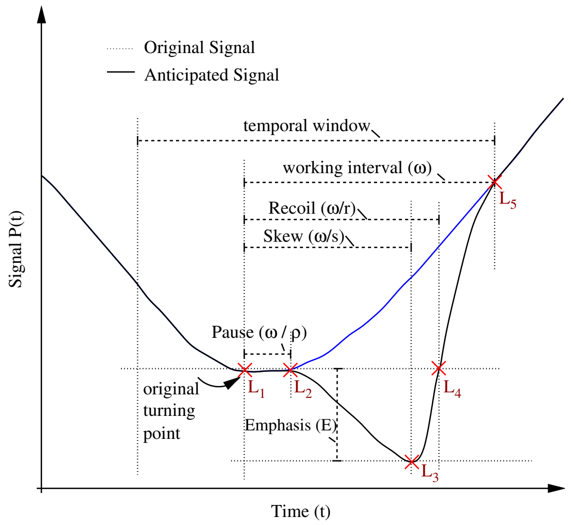

Fig. 6. Plotting one element of the metronome pose vector $P(t)$ varying over time t (corresponding to Fig. 5, left). A portion of the original pose trajectory (blue) is automatically identified local to the turning point (here, with duration ω), cut, and then replaced with a new trajectory fragment. The user can specify four parameters to control the shape of this fragment: delay (p), emphasis (E), recoil (r) and skew (s), as indicated.

2.2. Temporally coherent NPR painting

We now describe how our paintbox addresses the long-standing problem of temporally coherent video painting—full descriptions are given in [16,19]. We begin by regarding the video input as a spatio-temporal volume in line with recent work by Wang et al. [20]. We assume the video clip is free of cross-fades, cuts, and such like; techniques exist to reliable segment video sequences into suitable clips, see [21].

We have designed a data structure specifically with coherent and flexible rendering in mind. To form our data structure we begin by independently segmenting frames into connected homogeneous regions using standard Computer Vision techniques [22]. The criterion for homogeneity we have chosen is colour (after [23]). Associations are created between the regions in a frame and regions in adjacent frames, based upon similarities in colour, location, and shape. We have found this 2D + time associative approach preferable to a 3D segmentation of the video volume (used by [20]). Attributes used for association, for example colour, are permitted to vary with time, which improves robustness to changes in properties such as luminance. Small, rapidly moving objects may not overlap spatially from one frame to the next, a situation that causes problems using 3D segmentation, but which our approach handles. The result of our process is a set of sub-volumes that partition the spatio-temporal video volume, in which time is regarded as the third dimension, Fig. 7 illustrates with an example.

In the IR, the sub-volumes are represented by their bounding surfaces. We partition the bounding surfaces into pieces, that we dub *stroke surfaces*, each of which divides exactly two sub-volumes. Each stroke surface holds pointers to the volume that it partly bounds. The video volume is, therefore, stored in the IR using a *winged edge structure* [24]. Supplementary information about sub-volumes, such as mean colour, can also be stored and smoothed over time. Importantly, we maintain a graph that records connectivity between sub-volumes. This allows us to associate areas in a given frame that appear to be disconnected—separated by a lamp-post, perhaps—provided the sub-volumes connect at some instant in time. This does not always occur, and we allow the animator to intervene in such cases, but only as they deem necessary. Often user correction is also required to correct the segmentation, as segmented regions do not always correspond with semantic objects in the video. In practice, we bias our process towards spatial over-segmentation, on the observation that it is easier for a user to merge two video objects via point and click operations, than to sub-divide objects. Such interventions need only be made once to associate two objects for the entirety of the video sequence. Internal edge detail within the regions may also be encapsulated in the stroke surface representation; in such cases both pointers in the winged edge structure reference the same video object. Furthermore, the surface representation we employ prevents ''holes'' from appearing in rendered frames as a consequence of any manipulations we may make (such as smoothing the surfaces to enhance temporal coherence); we have found this to be a problematic issue with alternative representations such as closed bounded or solid voxel volumes.

Rendering begins by intersecting the surfaces and volumes with a plane of constant time. We can render just the surfaces to give a line drawing, just the volumes to paint coloured interiors, or render both together. We can also choose to render a subset of volumes, leaving the remainder to appear photorealistic, or apply different effects to different regions. We can even choose to paint some areas by keying in pixels from an alternative image or video source. The colour information stored for each video volume may be augmented with other attributes, inferred from the source video by the Computer Vision component. For example, we can attach rigid body reference frames to sub-volumes selected by the animator as important. These reference frames provide a stable, moving coordinate system that we use to place paint strokes. In this way we not only solve the coherence problem, for strokes are naturally coherent, but unify automated rotoscoping, matting, and painting within a single framework.

Temporal effects are possible. For example, we can apply a sinusoid to the stroke surfaces that makes the line drawing appear to ''wobble'' rhythmically; we can ''shatter'' the surfaces into pseudo-random shards, each of which exhibits temporal coherence over a short time.

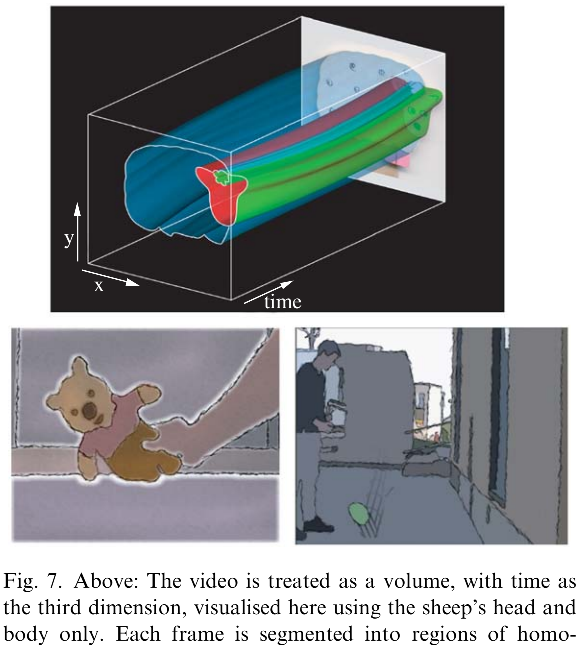

Fig. 7. Above: The video is treated as a volume, with time as the third dimension, visualised here using the sheep's head and body only. Each frame is segmented into regions of homogeneous colour, region boundaries are connected over time to form surfaces, which define sub-volumes. Temporal coherence can be improved by smoothing the surfaces over time, and controlled incoherence may be re-introduced by manipulation of the smoothed surfaces. Below: Sample frames from coherently shaded animations.

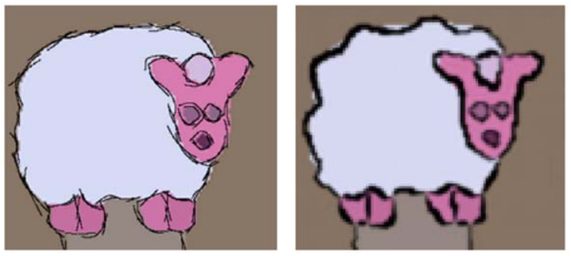

Fig. 8. Examples of control temporal incoherences that may be introduced. Left: Stroke surfaces are shattered and jittered to produce a sketchy effect. Right: High frequencies induced in the stroke surfaces produce a coherent ''wobbling'' effect.

The latter produces an aesthetically pleasing "sketchy" out-line to objects (Fig. 8). In this way we can re-introduce incoherence, but crucially this is now under animator control.

2.3. System integration

For final rendering we adopt a cell based view of the video; it is here the coherent painting and motion emphasis cues are integrated into one system. One consequence of our processing is that we are able to assign a relative depth ordering to objects being rendered, and these are rendered on *layers*. At any time instant *background layer* is made from pixels that are neither in an object identified by the animator, nor which occlude any object. Occasionally, and typically because of the deformation and anticipation cues, "holes" can appear in a background layer; we fill these holes with pixels taken from nearby frames. The *foreground* layer comprises all pixels that occlude an identified object. These pixels are taken from an *occlusion buffer* which forms part of our IR; explained elsewhere [1]. The *object layers* lay between the back-ground and foreground layers. In general, each object layer is in fact a compound of several layers. The back-most of these contains the object itself, near layers contain painting marks, including streak-lines and ghosting marks. If the object suffers any deformation, then all these layers suffer the same deformation. Similarly, if there is any change in the object's motion due to anticipatory cues, then all layers undergo the same change. Subject to these changes, final rendering is simply from back to front (Fig. 9).

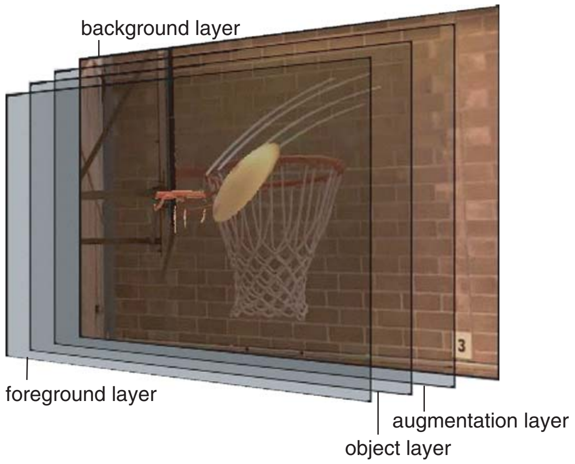

Fig. 9. Frames are rendered as layers, in back-to-front order. In this case a single object (a ball) is tracked to produce augmentation cues (streak-lines) and deformations (squash-and-stretch). Occluding pixels are overlaid in the foreground to produce the illusion of the ball deforming behind the hoop.

3. Discussion and conclusion

We have described a comprehensive and versatile Video Paintbox. It solves the problem of coherent painting; the same framework allows many novel painterly effects, and even rotoscoping. It introduces all major motion emphasis cues to video-based NPR. The animator can choose to paint, to emphasise motion, or to do both; to paint all or part of a video.

We have not attempted to build a fully automatic system. Instead, we have made use of mid-level vision techniques to make a robust system that animators can control. Some of the vision techniques described (correspondence trails and their filtering, collision analysis, pivot point computation, the stroke-surface framework) have been developed specifically for this application. The remainder we have taken from the standard vision literature.

The system would benefit from additional computer vision analysis. One example is segmentation; currently based on colour coherence it would benefit from use of texture or even motion; the vision literature holds suitable techniques. No change need be made to the way the video volume is then subsequently analysed—the stroke-surfaces can still be produced—but we might appeal to high-level vision method to enhance the automatic construction of the connectivity graph. Feature tracking is an area that might benefit from harnessing contemporary techniques such as CONDEN-SATION [25]. However, CONDENSATION is a high-level technique in that it requires a much more specific model of motion than Kalman filtering requires, which is why we opted for the latter. We can imaging a system that uses CONDENSATION tracker for common objects, such as people walking, but defaults to a Kalman filter in more general cases.

The single most restrictive assumption we make is that object motion is planar. Removing this assumption would add considerable complexity to the vision components, and there could be ramifications for the graphics components too. Whether the effort is cost-effective is debatable: many motions of interest are planar (not necessarily in the image plane, but we do not require that) and our approach does allow a wide range of video to be processed. Similarly, coherent painting currently operates through attachment of rigid frames to objects, and there might be advantages in using a deformable basis for soft objects. Such extensions would widen the classes of video footage that could be processed. This said, our system remains very flexible; this paper has not been able to detail all the options we have tried; such as individual control of colour channels, a variety of deformation functionals, and many pain-terly effects. Indeed, we feel that we have not exhausted plausible options. By far the easiest, and possibly the most productive, way forward for this work is to develop new deformations to better control anticipation (through inverse kinematics perhaps) and to adapt further static artistic styles from the NPR literature using our coherent rotoscoping framework.

References

[1] Collomosse JP. Higher level techniques for the artistic rendering of images and video. PhD thesis, University of Bath, UK; May 2004.

[2] Strothotte T, Priem B, Raab A, Schumann J, Forsey D. How to render frames and influence people. In: Proceedings of the Eurographics, vol. 13, Eurographics; 1994. p. C455–66.

[3] Hertzmann A. Painterly rendering with curved brush strokes of multiple sizes. In: Proceedings of the ACM SIGGRAPH; 1998. p. 453–60.

[4] Shiraishi M, Yamaguchi Y. An algorithm for automatic painterly rendering based on local source approximation. In: Proceedings of the ACM NPAR; 2000. p. 53–8.

[5] Gooch B, Coombe G, Shirley P. Artistic vision: painterly rendering using computer vision techniques. In: Proceedings of the ACM NPAR; 2002.

[6] Fekete J, Bizouarn T, Galas T, Taillefer F. Tic-tac-toon: a paperless system for professional 2D animations. In: Proceedings of the ACM SIGGRAPH; 1995. p. 79–90.

[7] Agarwala A. Snaketoonz: a semi-automatic approach to creating cel animation from video. In: Proceedings of the ACM NPAR; 2001.

[8] Litwinowicz P. Processing images and video for an impressionist effect. In: Proceedings of the ACM SIGGRAPH; 1997. p. 407–14.

[9] Kovacs L, Sziranyi T. Creating video animations combining stochastic paintbrush transformation and motion detection. In: Proceedings of the international conference on pattern recognition, vol. II; 2002. p. 1090–3.

[10] Hertzmann A, Perlin K. Painterly rendering for video and interaction. In: Proceedings of the ACM NPAR; 2000. p. 7–12.

[11] Green S, Salesin D, Schofield S, Hertzmann A, Litwinowicz P. Non-photorealistic rendering, ACM SIGGRAPH '99 non-photorealistic rendering course notes.

[12] Lasseter J. Principles of traditional animation applied to 3D computer animation. In: Proceedings of the ACM SIGGRAPH, vol. 21; 1987. p. 35–44.

[13] Hsu S, Lee I. Drawing and animation using skeletal strokes. In: Proceedings of the ACM SIGGRAPH; 1994. p. 109–18.

[14] Chenney S, Pingel M, Iverson R, Szymanski M. Simulating cartoon style animation. In: Proceedings of the ACM NPAR; 2002.

[15] Li Y, Gleicher M, Xu Y, Shum H. Stylizing motion with drawings. In: Proceedings of the symposium on computer animation; 2003.

[16] Collomosse JP, Rowntree D, Hall PM. Stroke surfaces: a spatio-temporal framework for temporally coherent non-photorealistic animations. Technical Report 2003–01, University of Bath, UK; June 2003.

[17] Williams D, Shah M. A fast algorithm for active contours and curvature estimation. CVGIP: Image Understanding 1992;55:14–26.

[18] Collomosse JP, Rowntree D, Hall PM. Video analysis for cartoon-like special effects. In: Proceedings of the British machine vision conference, vol. 2; 2003. p. 749–58.

[19] Collomosse JP, Hall PM. Stroke surfaces: temporally coherent artistic animations from video. IEEE Transactions on Visualization and Computer Graphics 2005; 11(5).

[20] Wang J, Xu Y, Shum H-Y, Cohen M. Video tooning. In: Proceedings of the ACM SIGGRAPH; 2004. p. 574–83.

[21] Zhang H, Kankanhalli A, Smoliar SW. Automatic partitioning of full-motion video. In: Mutlimedia systems, vol. 1; 1995.

[22] Comaniciu D, Meer P. Mean shift: a robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 2002;24:603–19.

[23] DeCarlo D, Santella A. Abstracted painterly renderings using eye-tracking data. In: Proceedings of the ACM SIGGRAPH; 2002. p. 769–76.

[24] Baumgart B. A polyhedral representation for computer vision. In: National computer conference; 1975. p. 589–96.

[25] Isard M, Blake A. Contour tracking by stochastic propagation of conditional density. In: Buxton B, Cipolla R, editors., Proceedings of the European conference on computer vision; 1996. p. 343–56.