Computer Animation and Human Animators

Philip J Willis

Computing Group, School of Mathematical Sciences.
University Of Bath, Claverton Down, Bath BA2 7AY, UK
Email: P.J.Willis@bath.ac.uk

Abstract

There is often a gap between what technology can deliver and what users want. Sometimes this is because of fundamental technological constraints but often it is because technologists do not address the realities of user needs and expectations before attempting to solve the problem.
The talk will describe our approach to computer animation and some of the pitfalls of trying to address a community which is visually more literate than typical computer graphics people, while being far less computer oriented.
Keywords: computer animation, animation

1 Introduction

The computer graphics community has paid much attention to improving ways of modelling and rendering a wide variety of objects, including natural phenomena. There is almost a feeling of competition in some areas, trying to outdo the effects produced last year. The results can be visually stunning but the real value of these techniques is only realised when they are put in the hands of those who are trying to create pictures as a means to an end.

Such people are visual designers, in the broadest sense of the term, who may have no computing skills at all but will typically be much more visually-aware than the mathematicians or computer programmers who win the conference prizes. To reach this 'industrial strength' software product, the core software will have been extended to offer a proper user-interface. Sometimes this is a straightforward process, though it seems to be surprisingly easy to get it wrong! The simple solution most often arises when the users do not need to get their hands on the picture being manipulated. For example, certain fractal-generation programs let the user set parameters but the program then runs non-interactively to build up the required picture.

Computer animation is an area which almost requires that the user manipulates the objects directly. The movements and distortions involved very much start in the mind of the animator and then have to be transferred to the animation. This is by no means an easy process and it is noteworthy that the 3D animation community is pushing towards higher-level control of largely autonomous actors, which 'know' how to move and how to avoid obstacles, rather than expecting the animator to build-up detailed movements.

Our own work is in 2D animation, which mimics the traditional process of layering together drawings to produce each frame. Here the drawing does not match reality, not just in the sense that such work tends to use exaggerated cartoon characters but also in the sense that the animator may deliberately produce drawings which cannot have come from a 'real' 3D character, in order to produce a particular effect. For example, both eyes may be visible in a side view, when one eye would be hidden in reality. This stretches to the limit the need for the animator to have direct control of the pictures while at the same time pushing the computer system beyond what would be needed for more conventional applications.

2 Traditional Animation

Traditional animation is an art which happens to make use of photographic technology as the means of recording moving pictures. As the pictures first have to be drawn, the movement originally exists only in the mind of the animator. The translation of this idea to paper requires skills beyond those of a good visual imagination and the ability to draw, because it turns out that the best results are obtained only when the limitations of the medium are overcome to expressive advantage by the animator: animation can be an art, rather than a craft.

As a result of nearly a century of practice, the animation community has developed a corpus of knowledge about how to represent movement and emotion while at the same time providing incidental clues to emphasise the effect required[1, 2]. For example, a rapid exit to the left is often preceded with a 'winding-up' movement to the *right*, effective when seen but with no counterpart in reality, so not predictable from everyday experience.

The move to a new recording technology, such as video, or to a new drawing technology, such as the computer, should not result in the throwing away of the previous 100 years' knowledge, yet this has happened in the computer graphics community. As late as 1987 John Lassiter found it necessary to write a paper recommending the application of traditional techniques in 3D computer animation[3]. Even so, much 3D work has concentrated on making the puppets more life-like (very useful for some applications but most traditional animation does not mimic life), or on making them more autonomous (to gear the amount of output from a given amount of animator's input) and hence less-finely controlled[5, 6, 7].

As for the apparently easier 2D computer animation, as early as 1978 Ed Catmull was arguing that there were grave difficulties with even bringing the new technology to the old approach[4]. It seems we have not made very much progress since, certainly nothing to compare with the extremely rapid progress in rendering or indeed in computer performance. Recently however, there has been a renewal of interest in 2D computer animation, some lessons have been learned and some of the real problems are being addressed. It is these ideas which we wish to explore briefly in this paper.

3 What is 2D Computer Animation?

First let us describe a model of traditional animation which has a ready analogy with computer graphics. We will focus on the central, graphical component of the process. Readers wishing for a full account, from initial concept to final product, should refer to details we have given elsewhere[8].

In essence, each frame of an animation is constructed from layers of transparent "cels", on which various components of the action are drawn and painted. Typically the rearmost layer is a background painted on paper, while the cels hold individual characters, or even parts of a character. The entire stack is brought together, illuminated and a single frame is recorded on film. Some components are then changed or moved, the next frame is exposed and so on. This is necessary because films cannot otherwise take advantage of what the computer graphics community calls "coherence", similarities which carry over from frame to frame. The use of cels allows re-use of material. For the cheaper forms of animation, this is taken to the extreme of re-drawing only those parts of a character which *must* move.

4 A Computer Approach

This sounds a close match to computer graphics. We can certainly build pictures in multiple frame buffers and composite them together, retaining any needed for the next frame. Of course this begs the question of how we represent the drawings, and indeed of how we interact with them to revise them. In a sense it can also lead us astray: we should not start by thinking of pixels at all. Some years ago work started in the author's laboratory on paint programs which, by using an underlying vector-based recording, were able to achieve high resolution independent of the display[9, 10]. This lead to a vector-based paint program, UltraPaint[11, 12, 13], with an independent scan-converter, UltraScan, capable of rendering at any definition. This software had a number of advantages over the pixel-pushing normally used for this kind of application. Specifically, a vector representation is inherently scale-free. It is important to understand that by "vector-based" we do not mean that only line drawings are possible. What we did was present the user with a normal paint program interface, including a choice of brush sizes and shapes and a colour palette. The user created drawings by direct painting: pixel-pushing was used to give the screen feedback. However, at the same time the program stored in vector form the path of the mouse, thus recording the drawing actions and indeed all of the menu selections made. This recording could be to arbitrary scale, within the generous limits of the 32 bit wordlength of the host computer, and indeed it was possible to add fine detail by zooming and continuing to paint. The full-quality final picture was generated off-line, essentially by replaying the actions over a virtual framestore of arbitrarily high definition. To achieve this we wrote a scanline renderer for these vector files, which compressed each scanline as it was produced. Thus it was possible to attain very high definition without the need for a full framestore. In fact we demonstrated pictures of 32k by 32k[13] and the performance of the renderer was linear in the resolution, avoiding the square-law problems of frame buffers.

During the UltraPaint project, we realised that the approach had the same advantages for moving pictures as it had for stills. Now the output-independence became an additional virtue. Most of the animation market is in television but television formats are in a state of flux as the merits of high-definition versus enhanced definition, and digital versus analogue are debated. The same is true in the domestic video market where various disk-based systems have come and gone. Previously, recording to 35mm film (the highest available quality) was the only way of future-proofing, but this is a very expensive solution. With the computer approach, it is possible to retain a form which can be scan-converted on demand to meet any future standard.

In the mid 1980s we discussed this approach at length with people known to us in the industry, who were also looking to escape from the tyranny of the pixel. These people subsequently founded Cambridge Animation Systems and their core product, Animo, has recently established itself as a world-leader. It differs in many detailed ways appropriate to a commercial product but it is a vector system at heart.

We had also moved our own research in the same direction, investigating new ways

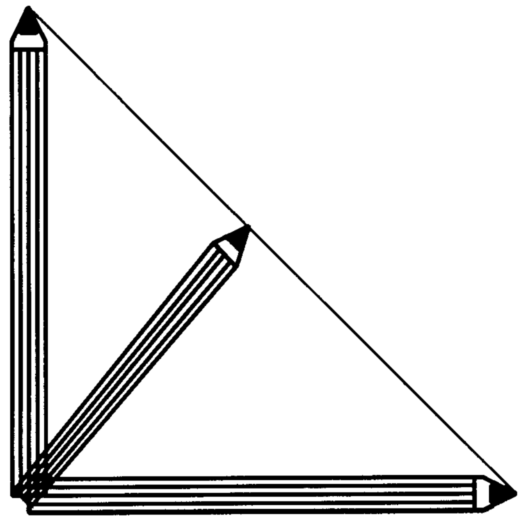


Figure 1: *Interpolating a falling pencil*

of representing layers of colour, with separate control of each cel's illumination[14] and new formulations of curves (in fact NURBS-based) appropriate to scanline rendering[15]. We thereby escape from the limitations of poor polygonisation resulting from the unforseen effects of the transformation pipeline, while ensuring that we retain the ultra-high definition option of our earlier paint program. Importantly, we have also examined the way the system can be presented to users[16].

5 User Interaction

Animators are used to working with key-frames, which are the most important poses of a sequence. The process of generating the other frames required is one of in-betweening, usually performed by teams of less-skilled technicians. A goal of computer animation has been to produce the in-between frames automatically, thereby both improving productivity. Typically this has proved more difficult than it might seem, so that additional key-frames have had to be inserted to guide the software more closely. This can degenerate to the point where the main animator is in fact drawing all the in-betweens as well!

What this tells us is that in-betweening is not a simple matter of interpolating between two pictures. The classic example, rapidly becoming the Utah teapot of 2D animation, is the falling pencil (Figure 1). Here the key-frames are the vertical and horizontal pencils. Linear interpolation produces in-betweens which shrink and grow larger again as the pencil falls. In-betweening is more of a circular interpolation than it is a linear one, as John Patterson and colleagues at the University of Glasgow realise[17]. This is an interesting area of research. The basic problem is that the transformation of one drawing into another can be done in many ways. In the case of the apparently-similar morphing, the in-betweens are only required to blend smoothly, there is no requirement that the various components of a picture behave in a correctly animated way. In-betweening is at the heart of the potential productivity gains from using a computer so it is important to have software which behaves as expected most of the time.

Another area which matters to the user is shape manipulation. Characters can be built from areas bounded by closed curves. These curves must be maleable in such a way that the user can perform both localised changes, to produce the right shape, and global changes, to produce squash and stretch effects. Some effects, such as the indentation of a character as a result of an object hitting it, really require a curve formulation with an appropriate (and variable) degree of "stiffness". Curve segments must not be so loosely coupled that the joins become apparent under these circumstances, but neither should they be so tightly coupled that all of the shape changes. It is easy for the software designer to overlook this problem and assume that the user will overcome it.

This is very much an application in which users wish to have their hands on the shape, working first with a rough freehand drawing and then refining it until the shape is right. It is perhaps not widely appreciated how much a drawing is a thinking tool to an animator. A rough sketch may be overdrawn many times, trying the effects of different emphasis, pushing a curve this way and that. It is not the case that the animator starts with a well-formed mental model and only uses the drawing to make the idea visible. Typical solutions to this problem are to bring the computer into use only after the thinking has been done by using traditional materials. Often this goes a step further: the user must first produce a clean drawing on paper which is then scanned to produce a pixel image. If the software leaves the drawing in pixel form, many of the advantages of using the computer are lost. Pixel-to-vector conversion therefore is the correct thing to do first.

We would argue that control of lighting is something which computers are good at, which is of enormous benefit to animation, and is poorly addressed by current 2D animation systems. Back-lights, soft spotlights and colouration can all be used to great dramatic effect. To our knowledge, our own proposals[14] are the only ones to address this properly. Our approach allows fine control of the front and back lighting of each cell, coloured filters to produce graded and localised effects, and full animation of the resulting effects.

Special effects are commonplace. These include: rain and other water effects such as waves on the sea and ripples on a pond; lightening flashes; dust swirls from rapidly departing characters; and many others. The vector-based approach wins here because these can all be created in a library, held until needed. They are then coloured appropriately and transformed to fit the scene in question.

Similar arguments apply to the main characters of a series. The important poses can be stored in a library and extracted as needed. This is an example of re-use and it follows that the visual quality of the animation can be improved cost-effectively: the effort which goes into drawing the character will be used over and over again, so it makes sense to take extra trouble to get a fully-detailed original.

Indeed a computer-based library of information is one of the parts of the production process which computers ought to be used for. They are good at dealing with large volumes of data, cross-referenced and accessed by multiple users. A typical medium-size animation studio has a number of people working simultaneously on a given production, and may have more than one project underway. Images are space-consuming, but vector-based drawings can be dramatically less so. Coupled with a fast renderer, this is not a disadvantage to users who wish to see the finished frames.

Finally, there are techniques from the computer graphics community which can be imported. High-speed motion is usually indicated with speed-streaks and dust eddies. Where appropriate, the rather different visual language of motion blur can be used. Similarly, texture map techniques, glossy surfaces and all the rest can be imported as needed, even in 2D. The important point is that this should be used to extend the animator's language, not to supplant it.

6 Concluding Remarks

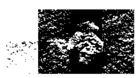
We have considered some aspects of animators, animation and the ways that computers might be used to go between the two. Certain general approaches have been identified and a few potential pitfalls mentioned. It seems from recent developments that the age of computer-mediated 2D animation production is just starting in earnest.

References

[1] A White, "The animator's workbook", Phaidon, (Oxford, 1986).

[2] F Thomas and O Johnson, "Disney animation - the illusion of life", Abbeville Press (New York 1981).

[3] J Lassiter, "Principles of traditional animation applied to 3D computer animation", *Computer Graphics* (SIGGRAPH'87) 21, 4 July 1987, pp. 35-44.



[4] E Catmull, "The problems of computer-assisted animation", *Computer Graphics* (SIGGRAPH'78) 12 (3) August 1978, pp. 348-353.

[5] N Magnenat-Thalmann and D Thalmann, "Computer animation: theory and Practice", Springer-Verlag, (Berlin 1985).

[6] N Magnenat-Thalmann and D Thalmann, "An indexed bibliography on computer animation", *IEEE CG&A* (July 1985).

[7] X Pueyo and D Tost, "Survey of computer animation", *Computer Graphics Forum* **7**(4), December 1988, pp. 281-300.

[8] J W Patterson and P J Willis, "Computer assisted animation: 2D or not 2D?", *Computer Journal*, to be published (1995).

[9] J B Hanson and P J Willis, "A graphic arts display system", *Proc. of Electronic Displays 82*, pp. 13-22, (Network 1982).

[10] P J Willis, "High fidelity pictures from digitizing tablets", *Displays*, pp. 147-151, (July 1983).

[11] P J Willis, "A paint program for the graphic arts in printing", *Proc Eurographics 84*, Copenhagen (North Holland 1984), pp. 109–120 (1984).

[12] P J Willis and G W Watters, "UltraPaint: a new approach to a painting system", *Computer Graphics Forum* **6**(2), pp. 125–132, (1987).

[13] G W Watters and P J Willis, "Scan converting extruded lines at ultra high definition", *Computer Graphics Forum* **6**(2), pp. 133–140, (1987).

[14] R J Oddy and P J Willis, "A physically-based colour model", *Computer Graphics Forum* **10**(2), pp. 121–127, (1991). Also: Eurographics UK Conference, April 10-12th, pp. 87-103, (1991).

[15] R J Oddy and P J Willis, "Rendering NURB regions for 2D animation", *Computer Graphics Forum*, **11**(3), pp. C-35 – C44 and p. C-465, (Proceedings issue: Eurographics 92 Conference, Cambridge, UK, Sept 1992).

[16] Mark Owen and Philip Willis, "Modelling and interpolating cartoon characters", *Proceedings of Computer Animation '94*, (Geneva, May 25-28th), IEEE, (1994), Text pp. 148-155; colour plates p. 203.

[17] J H Yu, "Inbetweening for computer animation using polar coordinates Linear Interpolation", *Research Report 90/R23*, (Sept 1990), Dept of Computing Science, University of Glasgow, UK.