CATEGORIZING ANIMATION SYSTEMS BY THEIR VOCABULARY OF MOTION

William Buxton & Alain Fournier
Dynamic Graphics Project
Computer Systems Research Institute
University of Toronto
Toronto, Ontario
Canada M5S 1A4

(416)-978-6320

ABSTRACT

A model for characterizing animation systems by the types of motion that they support is presented. The point of departure for the model differs from the more common practice of describing motion in terms of how it is effected (e.g., "keyframe" animation or "rotoscoping"). Rather, the model is based on the vocabulary of what one actually sees in the way of motion. A key aspect of the model is that it considers separately the notions of dimensionality, rendering style, and type of motion. The "vocabulary of motion" axis of the model identifies a number of increasingly complex types of motion. The importance of the model is that it helps better understand animation systems in terms of what they must support at the user's end. In addition, it is shown how there are important architectural considerations associated with each of the categories.

1. INTRODUCTION

We all seem to know what we mean by animation: Mickey Mouse, Yogi Bear and Bugs Bunny. So far, we have no problem. However, if we are asked to describe the visual vocabulary used in an animation, or to characterize different approaches, we start to have difficulty. In nearly every case, the characterization will be in terms of how the animation was made rather than what the animation does. Professional animators will talk about cel animation, rotoscoping, and key framing. And computer scientists, with their own peculiar arsenal of techniques, will talk about colour table and frame-buffer animation, and automatic inbetweening.

In many contexts, such categorizations are useful and informative. What they do not do directly, however, is inform us as to what is happening in terms of the vocabulary of motion used in the imagery. "What moved?", "How?", and "In relation to what?" are all questions which are most likely not directly addressed by this type of characterization.

Why is this important to us as computer scientists who are trying to build animation systems? In the simplest terms, we think that we can build better animation systems if we can identify different categories of animation with respect to visual vocabulary. The reason is that animation is hard. We cannot hope to develop a system architecture that can effectively accommodate and support all types of animation. But on the other hand, we don't need to. Animation is not just one thing. There are different vocabularies, and each is suited for conveying different concepts. The strength of many existing systems has suffered from attempts to over generalize their applicability. And yet if the converse approach is to build a powerful specialized system, then how do we arrive at an optimal architecture? Clearly we have to know for what the system is intended, and what the demands on that architecture will be.

Our intention in this paper is to develop a model of animation based on successive stages in an increasingly complex vocabulary to describe moving imagery. One objective is to establish a framework for future discussion. We want to define our terms of reference to minimize "apples-and-bananas" type comparisons. Each of the stages along the primary axis of our model involve what we consider a significant increase in power.

What is important to the engineer in all of this is our belief that there are identifiable architectural considerations that correspond to each stage in this model. What this means is that if one knows what one wants to say visually, and can describe the required vocabulary in terms of our model, there is some hope that the minimal (read most cost effective) architecture adequate for the job can be identified.

The model represents an attempt to bridge the gap between intent and content on the one hand, and appropriate technology on the other. As presented, the model has problems and is incomplete. But like Booth, Kochanek and Wein (1983), on which some of the ideas are based, it serves as the basis for discussion aimed at clarifying some important issues.

2. THE BASIS OF THE MODEL

The model which we present attempts to characterize animation within a three dimensional space. Each of the three axes represents an aspect of animation which we feel is independent of the other two (within the intended function of the model). The three axes are:

- number of dimensions
- style of rendering
- vocabulary of motion

The meaning of these axes can be understood by thinking about a bouncing ball. The first axis concerns issues like, "Is the ball represented as a bouncing disk in 2-D or a sphere in 3-D space?". The second axis concerns questions like, "Is it coloured?", "Does it have texture?", or "Is it just a line drawing?". The final axis is the one that we will spend the bulk of the paper discussing. It has to do with questions like "Is the ball spinning?", "Does it deform when it hits the ground?", and "How does the viewing position move throughout the scene?".

3. VOCABULARY OF MOTION

Our objective is to characterize motion in animation. So let us begin by defining this in the broad sense: motion in the object, in the scene, in the camera, in the eye, and in the mind's eye. We are, therefore, expanding the third axis of our model. Our expansion more-or-less follows a path of increasing complexity. By this, however, we do not intend to imply that this is some linear axis, or that things of higher complexity necessarily presume that all of the lower order elements are included. In fact, they generally are not.

3.1 Motion in the Mind's Eye: the Cartoon Strip

The lowly cartoon strip in the newspaper can serve as the initial entry into our vocabulary. In one sense there is no motion: the image is static. However, notions of motion can be conveyed with this class of imagery. The motion is simply in the mind's eye. What is important to note, however, is that this representation is often the most effective in conveying what it is we want to say about motion. An example would be a case where it is important to make comparisons of an object's position at critical stages through time. In this case, a few critical static frames may tell us more than a fully animated rendering.

3.2 Moving Point of View - Static Image: Panning and Zooming

In the next level of complexity, the image itself is still static. What does move is our view. In two dimensions this might mean zooming in and out and panning over static artwork. In three dimensions it might mean moving through a static city, as in Robert Able and Skidmore, Owings and Merril's Chicago film. The primary characteristic of animation at this level is that it permits the camera to explore, or the eye to be directed over, a static visual terrain.

3.3 Progressive Disclosure

There is a fuzzy line that delimits this next level. Nevertheless, we feel that it is a distinction that is worth making. The issue is, when does the drawing of a single image, or frame, become an animation in itself? The point is that our power of expression is different, for example, if we immediately put up an image with its caption, than if we first put up the image, and then added the caption word-by-word. In

one sense we are putting up the "same" image, but at the same time, the second can be considered animation.

The significance of this class of animation became clear in examining how good videotex artists learned to maintain interest throughout the interval that their images were being rendered. In fact, they were making a virtue of the failure of engineering to turn videotex into an video version of a simple slide-projector.

To the extent that we can consider such drawn-over-time images as a single image, then they deserve a distinct place in our taxonomy. Beyond that point, they belong in a later category.

3.4 Frame-to-Frame Transition Effects

All of the levels to this point have, in one sense or another, involved animation derived from a single visual source. In its simplest sense, this next level introduces into our vocabulary methods of making transitions from one source to another. The two sources may be static or dynamic. What we are concerned with here are methods of transition. These are effects which we are all familiar with, such as cuts, fades, dissolves, double exposures, wipes, flips, and spins (Madsen, 1969).

As in the previous stage, things break down if we push this level too far. For example, there is a frame-to-frame transition between *all* frames in an animation. Our intention here is to isolate transitions that are taking place at a rate significantly below the flicker rate, and where there is a significant difference between the images on either side of the transition.

3.5 Relative Changes of Position Among Objects: Translation

In one sense, this is the first stage where the objects in the visual space actually move. Here we want to introduce the case where an object's position can change with respect to its surroundings. If you think of a video game as animation, then the motion of the paddles and ball in Atari's first popular game Pong are a good illustration of what we mean. We want to continue to rule out, however, any changes other than position (such as size, shape, or orientation). The reason is that we want to isolate and emphasize the point made by Baecker in his Genesis system (Baecker, 1969): that there is significant power in even the restricted case of motion in space with no further transformations in orientation, shape or scale.

What Baecker did was demonstrate that an object, and its motion path and dynamics represented three separate concepts: object, trajectory and dynamics. The simplest example of this (although not great animation) is the motion of the tracking cross (or "cursor") of a graphics terminal. The cross is the object being "animated". The motion (path and dynamics) of the hand is what drives the animation. What is interesting is how rich and varied are the concepts that can be animated using even the simple vocabulary described thus far.

3.6 Non-Shape Distorting Transformations

In the previous stage, there was global motion of an object (translation) with respect to its environment. In this next stage, we augment our vocabulary to include transformations on the object itself, but only transformations that do not "mutate" its basic shape or definition. What we mean by this are transformations such as rotation or scaling. An example would be an animation of a clock's pendulum, or of a plane's propeller going around.

3.7 Changes in the Object Itself: Mutations

Finally, we introduce the type of motion which defines the nature and dynamics of changes and mutations in the object itself.[1] An example would be the motion of the change in shape of a tire as it goes flat. This is motion of a significantly different type than, for example, that that which defines the rotation of the tire before it hit the nail. This rotation would be another example of the type of motion introduced in the previous stage. And even that motion is different than that introduced earlier which would, for example, describe the motion of the tire relative to the nail. Of course we could backtrack even further

---

1. This class of motion has been termed "metamorphic" by Booth, Kochanek and Wein (1983). They distinguished between metamorphic and isometric motion. The latter corresponds to our "Relative Changes of Position" plus object rotation. We have chosen to distinguish things to a finer degree.

in this imaginary scene, and include the motion of the camera as it zooms in on the nail just before the puncture.

In visualizing this scene, we should not forget the other two axes of our model. It is worthwhile to remind ourselves that *with respect to the motion* all distinctions in the above example are quite independent of whether the car, tire, and nail are in 2 or 3 dimensions, or rendered as line drawings, solids, or in colour or black and white. The issue is, we now have a taxonomy which categorizes various classes of motion *across* animations of differing numbers of dimensions, and *across* different rendering styles. And in so doing, maintains a consistency of description. This can help us.

4. IMPLICATIONS ON DISPLAY PROCESSOR ARCHITECTURE

Having developed the above categorizations, it is interesting to examine each stage with respect to its hardware implications. Our ultimate aim is to sufficiently understand the demands of different styles of animation so that we can design appropriate display processor architectures. In the discussion that follows, we will concentrate on raster-scan technologies (Baecker, 1979), since they currently constitute the dominant technology in computer graphics.

4.1 Motion in the Mind's Eye: the Cartoon Strip

In Cartoon Strip animation, we really only have two main concerns. The first one is image resolution. If we are to break the screen down into several sub-frames, the display must be capable of sufficient resolution to give appropriate detail. The major tradeoff here is spatial resolution (horizontal/vertical) *vs* depth (to support anti aliasing)[2]. The second major issue is bandwidth *vs* display processor intelligence. Thus, if the image is modeled as a simple raster, then the display processor can be quite simple, but the bandwidth to transmit the image must be quite high. On the other hand, the image could be encoded (as geometric primitives, such as lines, polygons, etc., for example), thereby reducing the bandwidth. The cost for this, however, will be the higher level of intelligence required in the display processor.

4.2 Moving Point of View - Static Image: Panning and Zooming

Panning and zooming introduces some interesting problems. First, the processor needs to support the zooming and panning process. In this case, the resolution issue becomes more complicated. The simplest way to support these effects is through pixel replication, which means that as we zoom in, we lose resolution. However, there is also the option of storing a higher-level representation of the image in the display processor which can be rendered at maximum resolution regardless of how closely one has zoomed in. This is reasonably common in calligraphic displays. However, in colour raster systems, it is generally only seen in high-end systems, such as flight simulators.

In panning and zooming in 3-D, we have the problem of visible surface determination. When this is the only class of motion in the animation, there is a display processor based technique that can be used to significantly speed up the display process. This is the Binary Space Partitioning Tree (BSPT) algorithm introduced by Fuchs, Kedem Naylor (1981). Fuchs, Abram and Grant (1983) have demonstrated how a high end processor (such as an Adage Ikonas 3000) can be programmed to display scenes of this class made up of several hundred polygons at a rate of several frames a second. Furthermore, if the display were to be used for specialized applications within this class of 3-D imagery, then additional hardware assist could be provided that would speed things up even further.

4.3 Progressive Disclosure

The case of progressive disclosure is the opposite of panning and zooming. Here it is in inexpensive systems, such as videotex systems, that we see the effect most used. The reason is largely one of making a virtue out of a necessity. Videotex systems, such as Telidon (Godfrey & Chang, 1981), have shown that this form of presentation is very powerful in many instances. This being the case, note that when we move to higher-end systems it is quite difficult to script this type of animation.

---

2. It is important to note that there is a limit on the extent to which we can make up for lack of spatial resolution through anti aliasing. No amount of anti aliasing will obviate the limits on image density (such as line density) dictated by the spatial resolution.

4.4 Frame-to-Frame Transition Effects

In commercial video studios, Frame-to-Frame Transition Effects are done almost exclusively by special-purpose digital processors. Such processors, or software emulations of them, have not generally penetrated computer-graphics labs that are not directly involved in commercial film making. However, most high-end display processors can be microprogrammed to do these effects. As it stands, they are a very neglected part of our visual vocabulary.

4.5 Relative Changes of Position Among Objects: Translation

Not everyone working in interactive computer graphics think of their manipulating the display's cursor, or tracking symbol, as creating an animation. It is, however, in that it involves moving a graphical object, the tracking symbol, through space over time. It is also one of the few instances where we can do animation in real time. This is afforded by the fact that modern raster systems provide special purpose hardware to support tracking. In the Adage Ikonas 3000, for example, one can manipulate an arbitrary tracking symbol defined within a 32 by 32 raster. This example shows how the motion of the object defined is totally independent of its shape. Furthermore, it is easy to see how the hardware which supports the movement of the tracking symbol could be controlled by some software script just as easily as by a tablet. Two points emerge from this. First, we see that the tracking symbol handling mechanism can be naturally extended into a more general animation device. Second, we see that by restricting our animation to this class of motion, we can build relatively simple special purpose hardware which will support the animation in real time.

In fact, this tracking symbol mechanism has already been extended for more general animation. This is most commonly seen in video games. The general name given to this type of moving raster is a *sprite*. Sprite based display processors have expanded upon tracking symbol mechanisms in at least two ways. First, they generally support more than one simultaneously moving sprite. Eight is typical. Secondly, they often provide a larger raster in which the moving object can be defined.

In 2-D systems, if simple translation is sufficient with respect to motion, then the animation can be supported by a simple but powerful architecture. In this, note that the use of sprites has received very little attention in main-stream computer graphics, and they are not even mentioned in either of the two main computer graphics texts (Newman & Sproull, 1979; Foley & Van Dam, 1982).

Thinking of sprites in the context of our model suggests investigating analogous techniques for 3-D animation. One possibility could be realized by using a modified version of the BSPT algorithm. It could be applied when the number of polygons of the moving objects is low in comparison to the overall number in the scene. The polygons of each moving object could be treated by the user in a manner similar to a sprite (although they would be modelled at the geometric, rather than display level). Special algorithms could then be developed to accelerate the local transformation of these polygons, and their being added (and subsequently pruned) from the tree each frame.

4.6 Non-Shape Distorting Transformations

When we move on to consider Non-Shape Distorting Motion, we have the potential to build upon the concepts developed in the immediately preceding discussion. In 2-D, for example, one architectural feature that seems potentially useful is to consider supporting scaling and rotational transformations independently for each sprite. Similarly, in 3-D we could consider performing local transformations on each moving object before merging their polygons into the BSPT structure.

4.7 Changes in the Object Itself: Mutations

Shape distortions, or Mutations, place very different requirements on the display system than those discussed so far. In some cases, where the number of frames in a sequence is very low, they can all be preloaded into the display, whose processor then sequences through them. This is discussed in Booth and MacKay (1982), for example. This approach does nothing, however, to reduce the host's overhead in actually computing the frames. The technique most commonly used (in 2-D) for this class of motion is key frame animation with automatic inbetweening (Burtnyk & Wein, 1971; Kochanek, Bartels, & Booth, 1982; Reeves, 1981). More recent work on controlling tension and bias in parametric surfaces (e.g., Barskey & Beatty, 1983) is suited to this class of transformation. It is important to mention that shape distortion is an important cue in indicating motion in animation, as can be clearly seen in the Lucasfilm

film *Andre and Wally B.*

Fournier (1981) addresses the question of the extent to which linear interpolation (as in inbetweening) can be supported in hardware. One area that deserves attention, and which we have been investigating, is the extent to which the notions of Reeves (1981) can be transferred into 3-D. Our approach has been to keyframe between 3-D shapes defined as parametric surfaces, where the moving point constraints go between the control points. One question currently being investigated concerns the extent to which this computation can be carried out in the display processor itself.

Finally, there are many cases where the mutational motion is cyclical. Yogi bear's feet while walking is one example. Pac Man's munching mouth is another. In such cases where the motion can be generated by cycling through a short sequence of frames, we can provide some hardware assist. One method is through frame-buffer animation (Shoup, 1979; Booth & MacKay, 1982). A more general approach is to permit a sequence of rasters to be associated with each sprite, and provide a mechanism for controlling the timing of cycling through them. This is a technique commonly used in video games. Its main benefit over simple frame-buffer animation is that it allows local mutational change of an object as well as unlimited global change of position.

In summary, we have seen how each step along our Vocabulary of Motion axis has had particular implications on the display processor architecture. One consequence of this is that we have an aid that helps us find the appropriate hardware given a specified vocabulary. In addition, we have seen how in some cases, techniques to improve existing architectures have been suggested.

5. PROBLEMS WITH THE MODEL

The model described has proven useful in discussing architectures. However, some problems remain to be resolved. Most of them result from our attempt to categorize things by what was actually happening visually, rather than by describing the technical effect that generated them. First, there are some visual gestures that are difficult to place in the three-space defined by the model. One class of these, for example, are keying effects, such as chroma keying. They seem to belong in the neighbourhood of the Frame-to-Frame Transition Effects. Among other problems, however, keying effects do not fit well with the axis of dimensionality. They are essentially 2-D effects that, when applied to 3-D scenes, are applied only after the 3-D image has been projected onto a 2-D surface. We have similar problems when we try to integrate colour changes, as in colour table manipulation. This is generally a global effect which works on the final 2-D projection of the image, and again, is not well handled by the current formulation of the model.

The dimensionality axis seems to be the source of another problem. Take the example of an animation of a flipping coin. In two dimensions, this could only be achieved through mutations on the shape of the coin (in order to simulate the effect of perspective transformations). In a 3-D animation, however, there would be no mutation of the coin. The perspective transformation would occur by simple virtue of our move in the dimensionality axis. Thus, according to our model, the 2-D animation of this scene would require more steps in complexity along the Vocabulary of Motion axis. This is, in fact, a meaningful (and perhaps useful) distinction in what is required to generate the sequence. However, for an audience watching the two scenes, there is no distinction. The vocabulary of motion, in terms of what is happening visually, is identical.

6. CONCLUSIONS

The model that we have presented has problems, and is incomplete. Nevertheless, we believe that the concepts are important even if only as an example of trying to analyse animation from what is happening visually. But even further, we believe that we have demonstrated that it is of some practical value in designing and understanding animation systems. That the model is of use in spite of its acknowledged flaws simply encourages us to pursue it further.

7. ACKNOWLEDGEMENTS

The work described in this paper was undertaken under contract OST83-00128 from the Communications Research Centre of the Federal Department of Communications. We would like to acknowledge

the support of R. Fujaros of the CRC for his support in this project. We are also deeply indebted to Ms. Catherine Richards of the Canadian Broadcasting Corporation, who made a significant contribution to the ideas presented in this paper. Finally, we would like to acknowledge the contribution of our students in the Dynamic Graphics Project who did much of the work underlying this project.

8. REFERENCES

Baecker, R. (1969). Picture Driven Animation. *Proceedings of the AFIPS Spring Joint Computer Conference.* **34,** 273 - 288.

Baecker, R. (1979). Digital Video Display Systems and Dynamic Graphics. *Computer Graphics* **13(2),** 48 - 56.

Barsky, B. & Beatty, J. C. (1983). Local Control of Bias and Tension in Beta-splines. *ACM Tansactions on Graphics* **2(2),** April, 1983

Booth, K., Kochanek, D. & Wein, M. (1983). Computers Animate Films and Video. *IEEE Spectrum* **20(2),** 44 - 51.

Booth, K. & MacKay, S. (1982). Techniques for Frame Buffer Animation. *Proceedings of Graphics Interface '82,* Toronto, Ontario, 213 - 220.

Burtnyk, N. & Wein, M. (1971). Computer Generated Keyframe Animation. *Journal of the SMPTE* **80,** 149 - 153.

Foley, J. & Van Dam, A. (1982). *Fundamentals of Interactive Computer Graphics.* Reading, MA: Addison-Wesley.

Fournier, A. (1981). A Proposal for a Four-Dimensional Graphics System. *Proceedings of the 7th Conference of the Canadian Man-Computer Communications Society,* 371 - 375.

Fuchs, H., Abram, G. & Grant, E. (1983). Near Real-Time Shaded Display of Rigid Objects. *Computer Graphics* **17(3),** 65 - 69.

Fuchs, H, Kedem, Z. & Naylor, B. (1980). On Visible Surface Generation by A Priori Tree Structures. *Computer Graphics* **14(3),** 124 - 133.

Godfrey, D. & Chang, E. (Eds.) (1981). *The Telidon Book.* Reston, VA: Reston Publishing.

Kochanek, D., Bartels, R. & Booth, K. (1982). A Computer System for Smooth Keyframe Animation. *Technical Report CS-82-42,* Computer Graphics Laboratory, University of Waterloo.

Madsen, R. (1969). *Animated Film: Concepts, Methods, Uses.* New York: Interland Publishing.

Newman, W. & Sproull, R. (1979). *Principles of Interactive Computer Graphics* (2nd Edition). New York: McGraw-Hill.

Reeves, W. (1981). Inbetweening for Computer Animation Utilizing Moving Point Constraints, *Computer Graphics,* **15(3),** 263 - 269.

Shoup, R. (1979). Colour Table Animation. *Computer Graphics* **13(2),** 8 - 13.