Integrating Computer Animation and Multimedia

Martin Preston, Terry Hewitt
Computer Graphics Unit, Manchester Computing,
University of Manchester, Manchester, M13 9PL, United Kingdom.
Email: *{preston | w.t.hewitt}@mcc.ac.uk*
WWW: *http://info.mcc.ac.uk/CGU/staff/{preston | hewitt}*

Abstract

Multimedia provides an immensely powerful tool for the dissemination of both information and entertainment. Current multimedia presentations consist of synchronised excerpts of media (such as sound, video & text) which are coordinated by an author to ensure a clear narrative is presented to the audience. However, each of the segments of the presentation consist of previously recorded footage, only the timing and synchronisation are dynamically constructed. The next logical advance for such systems is therefore to include the capability of generating material 'on-the-fly' in response to the actions of the audience. This paper describes a mechanism for using computer animation to generate this interactive material. Unlike previous animation techniques the approach presented here is suitable for use in constructing a storyline which the author can control, but the user can influence. In order to allow such techniques to be used we also present a multimedia authoring & playback system which incorporates interactive animation with existing media.

Keywords Multimedia, Computer Animation, Keyframing.

1. Introduction

The integration of existing media types into a coherent, and coordinated whole has presented information and entertainment providers with an enormous opportunity to significantly improve the manner in which they disseminate information. Contemporary multimedia systems allow authors to collate material stored in a variety of formats (principally text, video & audio), and combine them in the preparation of a single 'multimedia document', which can be read (or perhaps 'played') as though it were a more traditional monomedia experience. However, advanced though these systems are, they only allow the author to integrate *existing* material in a dynamic manner, none of the component media items are generated in response to the reader.

The logical next-step for multimedia systems is therefore to include the capacity to generate media 'on-the-fly', in response to the actions of the user, or audience. Whilst it is possible to imagine sound being generated interactively in such a fashion, it is harder to envisage video or text being produced dynamically (except in extremely limited circumstances). In order to produce interactive visual material it is therefore necessary to consider how more synthetic 'media', such as computer animation, might be incorporated.

Unfortunately the bulk of computer animation techniques are not suitable for inclusion in an interactive system, as they were developed to assist in the production of passive films, and so require the attention of an animator to make *any* change to the motion. The family of animation tools which most closely match our requirements are simulation techniques, where movement is generated either through simple dynamic simulation [16], or as the result of autonomous controllers [12][15][9]. Unfortunately, because the simulation nature of these tools means that the exact result is undetermined, they do not provide all the functionality required in multimedia presentations, where the author often needs to retain control over the narrative.

This paper describes a new form of 3D animation synthesis which aims to combine the controllability of traditional computer animation with the interactive nature of these newer techniques, in producing motion synthesis tools which may be used by a multimedia author in the preparation of a truly interactive document. We achieve this in two ways, firstly by developing a multimedia authoring mechanism which accommodates interactive 3D graphics, and secondly by adapting motion synthesis techniques so they may be used to construct a storyline which is capable of responding to the audience. Therefore these techniques add to existing multimedia formats, rather than replacing them.

The inclusion of interactive animation is beneficial to multimedia for two principal reasons. Firstly it allows authors to enliven presentations which would otherwise consist of purely pre-recorded media, and secondly it allows the production of documents which could not be created by any other means, such as presentations which require 3D information.

The principal contributions of this paper are:

- We outline a presentation authoring model which enables authors to integrate interactive, or 'user-driven' animation, into

Published in *Computer Graphics Forum* (Proceedings of Eurographics '96) 15(3).

a more conventional multimedia document.

- We describe how existing animation techniques may be adapted to allow them to be used as part of such systems (using a state machine which defines what properties the motion must exhibit).
- We describe a framework for interactive animation algorithms to assist in further work, and finally,
- We present two animation techniques which fit into this framework.

The rest of the paper is structured as follows. Section 2 highlights related attempts to produce interactive animation, as well as discussing the general structure of a multimedia presentation. Section 3 presents our mechanism for enabling a 3D narrative to be imposed on a presentation and highlights the differences we need to make to include animation. Section 4 discusses the states that the author constructs the narrative with, and section 5 outlines how animation is controlled by the author using the arcs between states. Following this Section 6 discusses the sample implementation as well as proposing a new motion synthesis technique which enables animators to exploit keyframe control in this new environment. Section 7 presents a case study of interactive animation in multimedia (including a description of the second motion generation tool), and section 8 concludes by highlighting both the advances proposed, and further work suggested by the results.

2. Related Work

There are two broad categories of multimedia systems, those environments which enable the author to combine existing media in constructing a narrative [2][1], and those that add multimedia capabilities to existing simulation or virtual reality programs [13]. As we are interested in adding to the functionality of both these systems it is important to highlight the mechanism by which authors construct presentation in the two styles.

Four primary presentation authoring metaphors have been developed for the first group; scripting languages [5], time line specification, organisation by hierarchical composition and finally a combination of the last two into a time-hierarchy style [1], where both the organisation and synchronisation of media playbacks are easily viewed. In contrast to this categorisation the second style of multimedia systems tend to employ far more *ad-hoc* approaches, where media playback is instead initiated by events in the virtual world [13], without any clear metaphor for authoring of such systems. However, they often provide the ability to add 3D graphics & simulation, so it is necessary to combine their non-predictive nature with the more strongly scripted styles of the first classification in the production of our multimedia & animation environment.

Having identified some of the functionality of the multimedia system which we wish to supplement, it is now necessary to identify areas of computer animation research which most closely match our requirements. In such literature ‘interactive animation’ is normally used to refer to systems which allow the animators to construct some model of the world, which executes to assist the artist in constructing the keyframes which control the animation [18]. Such systems interactive nature does not extend to the animation *playback*, and therefore they do not provide all the functionality we require in a multimedia presentation. In order to achieve this we must have some mechanism for modelling non-linear animations. Kalra & Barr [7] go some way to providing a basis for this in their description of a mechanism for coordinating simulation and kinematic animation tools together in an ‘event system’. Using a collection of states, each of which corresponds to a motion synthesis technique, they trigger changes of state by the satisfaction of conditions.

This is sufficient for systems where we only need state changes to occur in response to simulation, but in most multimedia presentations we must control state change by a storyline, and so we must modify the animation authoring representation to allow this.

3. The Authoring Model

In this section we describe a narrative authoring & storage model which is capable of accommodating interactive animation as part of the multimedia document. We need a new model because; while the most powerful scheme (the time hierarchy model [1]) is general enough to accommodate conventional media, it becomes cumbersome when used with interactive animation (the control of a sequence of 3D objects, lights and cameras). There are two principal reasons for this:

- In generating an interactive environment we need to encapsulate some possibility of choice, i.e., we need some mechanism for deciding which course the narrative takes dependent on the current state of the world model.
- We need some mechanism for describing laxness in state satisfaction. Whilst simple media (such as audio or video) are either playing or not, in representing 3D animation the author will often need to specify bounds of satisfaction, e.g., the author may not care exactly how a mannequin sits in a chair at some point in the presentation, so long as the viewer would recognise the pose as sitting, hence there is some bounded goal for the mannequin to reach.

As neither of these two goals can be represented in the time-hierarchy model we must construct a variant which suits our requirements. The system we describe achieves this by combining a variant of Kalra & Barrs [7] state machine model and the time-hierarchy schema. The presentation is represented as a hierarchy of controls; at the micro level the author controls the narrative that portions of the environment will follow as well as the ability of the audience to influence it, and at the

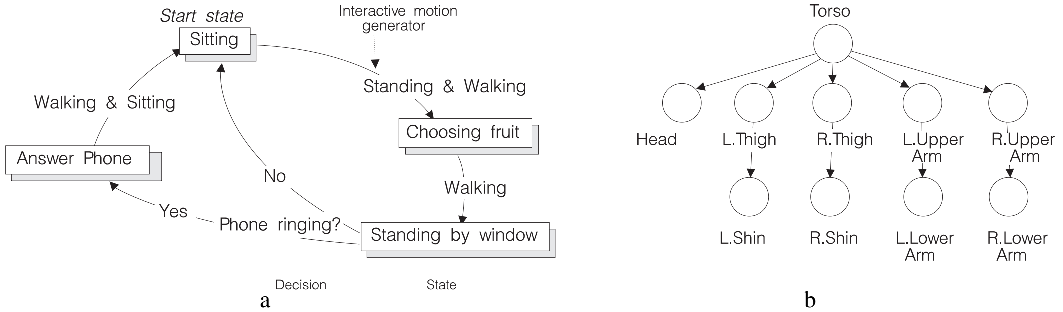

Figure 1 A simple state machine, which forms a micro level component of the narrative (a), controls a number of set of connected objects (in this case a single set) (b).

macro level the author combines these, and other non-interactive media, into the complete presentation.

This stratification of control is presented in the following 3 subsections. Section 3.1 introduces the model we employ at the micro level: a state machine (which will be described in greater detail in Sections 4 & 5). In 3.2 we present the macro level model of the presentation. Finally in section 3.3 we highlight the advantages of this approach.

3.1 Micro Level: The State Machine

Each state machine is a collection of tools which when played cause movement in some portion of the virtual world being maintained by the viewing system. The virtual world consists of a camera, some lights and a collection of bodies (represented by NURBS or polyhedra). These bodies are grouped into a collection of (possibly intersecting) sets, whose movement is only *directly* related to other bodies in that set. For example, an animator may choose to construct a mannequin from 10 individual bodies (2 bodies per leg, two per arm, a torso and a head). The movement of each individual body has an important bearing on the movement of the others, and so they form a set.

Each state-machine that the author uses to construct the narrative controls a number of these sets, and normally a state machine will only control a single set (the mannequin in our example). However, as we allow it to control multiple sets it is also possible to model more complicated interactions (perhaps the authors wants the two mannequins to shake hands).

Having now defined what a state machine controls, it is necessary to identify what form it takes. The description of this is most easily made by contrasting it to a previous approach. In Kalra & Barrs' system [7] the FSM defines, for each state, a motion synthesis device and the conditions in the world model for this synthesizer to be enabled. This allowed complicated simulated environments to be easily constructed. A simple FSM for such an environment might be textually represented as: do a dynamic simulation until some time limit has expired, then do some keyframing until you hit point A, then do more simulation.

This isn't sufficient for our purposes, as we want to model more elaborate storylines than pure simulations, so instead the states are used to represent *goals which must be attained.* Therefore in our model the arcs represent the motion synthesis algorithms, which move the collection of bodies between the states and may be arbitrarily complicated. A very simple state model which conforms to this style is shown in Figure 1, where we are controlling a mannequin as it moves around the room, the walking arcs here using an elaborate algorithm [3]. The state-machine is circular, so there is no defined completion point, but we do mark one state as the beginning position (the 'sitting' state in Figure 1).

This simple example also shows the ability to model decision making at the micro-level, as two possible arcs exit the standing state, with the decision being made based upon some metric which is stored in the state (which we discuss further in section 4). Each of the states may trigger the playback of other pre-recorded media upon arrival, so in Figure 1 we might wish to playback a recording of footsteps while walking.

3.2 Macro Level: The Time Line

We have now introduced the basic building block of an interactive presentation which includes non-predictive 3D animation: the state machine. However, such a primitive approach becomes cumbersome once we wish to either control more complicated portions of the world, or need to use cross-set synchronisation. Therefore the multimedia author will also use a higher level, or macro, construction system to describe his or her presentation.

This is a variant on the time-hierarchy model proposed by Ackermann [1], which uses the state machine as one of its building components, rather than purely the playback of existing footage. This means that the author constructs his or her narrative from collections of state machines, which he or she synchronises with playback of more conventional media, using a timeline based display (as shown in Figure 2). Therefore this macro level description of the presentation imposes the relationship between the state machines & other media with time. A graphical depiction of the presentation is helpful in

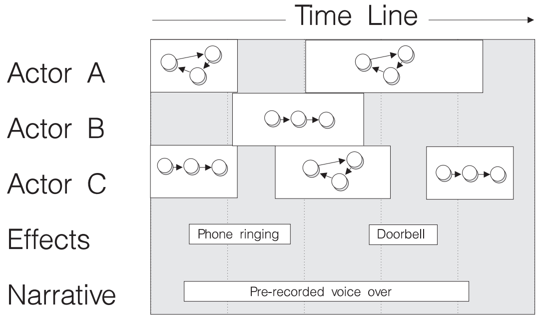

Figure 2 The macro level view of the narrative, with state machines and media playbacks arranged along a time line.

describing the strengths of this process, and is used to assist the author in controlling the presentation.

In the previous sub-section we described how the virtual world in which the presentation occurs is decomposed into a sequence of sets of bodies, with the grouping being designed to assist in the animation. These sets, which we refer to as 'actors', are arranged graphically in a script format relative to time (as shown in Figure 2). Likewise more traditional media (such as audio, video & text) are also arranged alongside the actors. Across the top of the display the time line is shown. Where gaps exist in the hierarchy no effect is being imposed on either the scene or other media devices.

The author constructs the narrative by placing state machines on the timeline for each actor. By aligning the temporal position of these state glyphs with those controlling other actors, cross actor synchrony can be enforced. In order to accommodate open-ended narratives the author may choose not to stipulate an end-time for each state machine, and this will cause that machine to be active for the lifetime of the presentation (we will see an example of this in Section 8). The author can place media initiation events on the script [1] which signal the playback of prerecorded media. Looping in the presentation at the macro level is depicted by graphics placed in the time line.

3.3 Combining the hierarchy

The combination of micro and macro views of the presentation have the advantage of

- allowing the author to combine 3D animation control in a multimedia authoring environment in which he or she is likely to be familiar with.
- assisting the animator in producing a description of an animation which conforms to a narrative, but whose temporal synchronisation with other media types is readily apparent, and finally
- having a clear description of the entire presentation where the portions which the audience can effect (the micro level) is kept separate from the fixed portion of the presentation (which is at the macro level).

4. States

Having outlined a framework for the description of a multimedia presentation which includes animation, we now describe in greater detail the micro level view of the narrative. In this section we will describe the contents of the states in the machine, and highlight how they are used both as an authors primary mechanism of controlling the narrative, and as a way of adding other media.

The micro level view controls motion in a 3D model of the world, which includes lights, a camera and a sequence of objects which may be connected in a hierarchy. However, as we are simulating the behaviour of the world, rather than simply representing its appearance, we also maintain a list of *state variables*, which are visible to all state machines in the narrative. These are simply variables which the user can define for his or her own use, and should be used to maintain that portion of the state which is difficult to represent purely by the state machine. For example, in the graph shown in Figure 1 we need some mechanism for setting the phone ringing. This may be caused by the satisfaction of another state elsewhere in either a micro or macro view of the narrative, and so is represented by a state variable.

A summary of the data which is stored in each state is shown in Figure 3. As discussed in section 3.1, a state in our model represents the goals that the narrative must satisfy. So, from an animators point of view, they can be viewed as poses that the actor must traverse for the storyline to be maintained. Therefore the state representation must at least contain the time at which the state must be met, and the position of the body set (i.e., the x, y, z as well as rotation of each body).

However, as described in section 3, we would also like to incorporate some degree of laxness in storyline description in the micro-level. Therefore our state-model actually consists of either

- specific values for each of these pieces of data,
- ranges of values, or finally

Basic Information:	
Time:	Specific Value (e.g. 0.1s) | Range (e.g., 1-2) | Unspecified
Geometry:	For each geometric dimension (including (x, y, z) positions and joint angles) Specific value | Range of acceptable values | Unspecified.
Media Events:	List of media events which are executed on arrival at state, e.g., "Play fanfare.au" or "Skip to 10sec into narrative.mpg and begin playing"
Advanced Information:	
List of exiting arcs:	An ordered list of arcs which exit this state. For each arc there exists a condition function, which must be true for this arc to be selected, e.g., `table.x > 10.0 & !phone_ringing`
Changes to variables:	A collection of changes to the state variables, e.g., `phone_ringing = 1;`

Figure 3 The information stored in each state in the state machine.

- unspecified.

This final case assists the animator/author in constructing some special cases, and provides an extreme case of indecision. For example, an author may wish to include portions of simulation within a narrative, and so may wish to describe a state which represents a particular time, but which may be met by any pose of actor. The state also needs to include initiations of other forms of media playback. This will enable authors to key events such as sound effects to particular states, and so tag pre-recorded media to unpredictable times.

The remainder of the information within the state is used by the author to express how the user may affect the playback. This is achieved in two ways: firstly by allowing the state to be defined relative to a portion of the scene that the user can manipulate, and secondly by including conditional selection of which arc to choose when exiting a state. The first requirement is achieved by a simple hierarchical description of the scene, if we wish to allow the user to modify where the mannequin in the example shown in Figure 1 sits down, then we define the geometry in the 'Sitting' state to be relative to the chair (which the user can move).

The ability to model conditional execution is achieved through use of the state variables. Each state contains an ordered list of exiting arcs, and for each a condition which must be true for that arc to be selected. When the playback program needs to choose which arc to use it loops through them, and the first arc whose condition it finds true is selected. If no conditions are suitable then the simulator continues to loop (at the next time step) until one becomes true. More elaborate decision strategies are possible, for example, some degree of randomness may be useful in a game-style presentation.

The final piece of information stored in the state is a sequence of variables which are changed when the state is *entered*. This allows us to achieve some measure of synchrony at the micro level, as we can allow the occurrence of events in other state machines to trigger activity here, e.g., another mannequin may lift another phone, which causes the variable `phone_ringing` to be altered, which causes a different arc to be selected in this machine.

5. Arcs

The states describe the important properties of the presentation, but it is the arcs which control how the world moves, i.e., how animation is incorporated into this document. The arcs represent the role which the motion synthesis techniques have within the multimedia presentation, and is therefore the most important contribution to the new system.

These arcs are designed to connect different states, and in playback they are responsible for ensuring that the objects they control reach the end-state within the constraints imposed. Because the states they connect may be loosely defined (perhaps their geometry is a range, or their time may be unspecified) no single animation technique is sufficient for all cases. For authors to be able to construct presentations easily it is important that a range of arcs are available, which are suitable for different situations. So we first present a classification scheme for such arcs, before presenting two such techniques in sections 6.1 & 7.

When an author is constructing a narrative he or she first defines the collection of states which must be met for the narrative to be followed, and then selects the relevant motion synthesizer to cause the states to be followed. The freedom of his or her decision is based on two factors: the nature of the destination state (specified, range of unspecified), and the manner in which they operate during playback. These two classification schemes, which together allow a multimedia author to easily select a motion generator which meets his or her requirements, are the subject of the next two subsections.

Table 1 Arcs may be classified by the degree of uncertainty of their destination.

		Time		
		Specified	Range	Unspecified
Geometry	Specified	Any existing animation technique.	Example shown in Section 6.1	Might employ optimisation
	Range	Could employ modification of constraints	Might employ minimisation	Might employ minimisation
	Unspecified	Autonomous Controllers	Autonomous Controllers	Forward Dynamic Simulation

5.1 Classification by Connected States

Table 1 shows 9 different classes of motion generator which may be incorporated in this system. The two principal factors which decide which class of generator may be used are; the specification of the time at which the goal must be met, and how rigidly the geometry (position as well as shape & colour) are described. Some of these combinations correspond to existing animation techniques (shown as shaded blocks in Table 1):

- If the destination state and time are rigidly described (and so the audience cannot effect the animation) then we may let the author define the animation in advance, using any form of motion synthesis technique he or she finds useful (such as keyframing [4], spacetime [17] or procedural control [14]).
- If both the destination time & geometry are unspecified, then the author may use any form of pure forward dynamic simulation [16](or even use a null arc which causes no movement).
- If the geometry is unspecified, but the time range is either specific or a range, then we may employ autonomous controllers [9], with the time limit causing termination of the solution.

The remaining 5 types of motion generator have not been developed by previous researchers, largely as the facilities they would provide are of little help in the production of conventional computer animation. They are:

- Motion generators which drive objects to a specified geometry target, within a range of time specified by the animator (we describe such an arc in section 6.1).
- Techniques which move the objects to a particular orientation in an unknown amount of time, presumably using some extra information provided by the authors (e.g., if the FSM is controlling a mannequin the system may be provided with an accurate model of human capabilities to guide the synthesis of a good path).
- Algorithms which must reach a range of geometric positions by a specified time. With some adaptation, some constraint maintenance techniques could be used as motion generators in this class [6].
- Those techniques which take the objects to within a range of positions, within some prescribed time range.
- Tools to drive objects to a range of positions in an unknown amount of time.

5.2 Classification by Implementation

Arc motion generators can also be classified by the manner of their operation. During the system playback these algorithms are called at each time step to bring their set of objects closer to their destination. Fundamentally these techniques will all need to take into account the configuration of the virtual world (such as position and shape of the scenery), the time, and finally the activity of the audience in achieving their goals. We can therefore classify such algorithms on *how often* they examine this configuration during their operation.

This allows us to split the motion generators into two further sub-groups: firstly those which examine the world state only at a limited number of times (in extreme cases only once), and secondly those which examine the world at *every* time step. This categorisation is important to the author, rather than purely to the simulator, because it has a large effect on how the relevant arc may be used within a narrative.

Consider an example of an arc which causes a hopping motion in the collection of objects that it controls. The arc maintains a list of pre-determined keyframe positions of objects during the hop, and during playback generates the motion by determining the end-state, and rotating these pre-defined values so that the objects hop towards the destination (such an arc is described in section 7). If this arc were implemented in a manner which allowed it to query the end-state at every time step there is a possibility (if the user moves the end position during the hop) that the object would change direction in mid-air. Whilst an author *may* want this, it is important that he or she is aware that such an unrealistic movement could be caused.

Additionally arcs which examine the end-goal only once may not drive the objects towards the destination all the time (as the destination may move without their knowledge), and so it is important that the author is careful to only use them in cases where it would not cause problems in the narrative.

This classification, and the one in the previous subsection, allows authors to easily identify collection of motion generator arcs which are suitable for use in a particular context. By producing a collection of states which define the important parts of the storyline, and connecting them using arcs which are categorised into the correct groups, it is possible to ease the process of constructing an interactive animation.

6. Implementation

Previous sections have presented a description of the changes we must make to a multimedia presentation to permit the inclusion of animation. Here we turn to the implementation of such a system, with Section 6.1 presenting a motion generator which fits into the scheme we have outlined, and Sections 6.2 & 6.3 outlining how the authoring and playback of these interactive presentations is performed.

6.1 Keyframe Control for Multimedia

In describing the role of the geometry in the states (in section 4) we noted that they are similar to keyframes in conventional computer animation. In the classification scheme in 5.1 we included keyframe control [4], but showed that it could only be used in cases where the end time & geometry were completely fixed. In this section we present an adaptation of keyframing which drives the collection of bodies to a fixed geometry in a *range of time*, and also allows the end-state to be interactively moved by the users. In doing this the technique queries the virtual world configuration at each time step, and so falls into the second category of groups highlighted in 5.2.

The arc first selects a ‘target completion time’, which falls mid-way in the range the author has provided. The timing curve initially describes the distance travelled from the start position at t=0.0 to the end position at this target time. At each time step the arc examines the end state. If it hasn’t changed since the previous time step then we update the objects along the spline using the timing curve. If it *has* changed then we try to change the path along which the object passes in two ways: firstly by rotating and scaling the remaining portion of the spline (if the destination has changed), and secondly by scaling the timing curve to meet the destination within the time range (while minimising the change in velocity).

We first calculate the scaling for the timing curve we would like to perform if we had an unlimited time range:

$$\textit{time remaining} = \frac{\textit{new distance}}{\textit{old distance}} \times \textit{time remaining} \qquad (1)$$

If this new end time is within the range, then we scale the timing curve and carry on as before, but if it isn’t then we attempt to perform the relevant scaling which falls within the range (as shown in Figure 4). This simple modification to basic keyframing allows it to be used in a variety of situations in interactive multimedia presentations, and has the advantage of presenting a pleasant and familiar interface to animators trying to adapt their work for interactive use.

6.2 Authoring Tool

As described in section 3 the narrative representation employed is a two tier scheme, and this split is reflected in the interface that the author employs. The prototype system employs the Tcl/Tk [8] language to allow a pleasant user interface to be presented to the author for both macro and micro level editing. The macro level editor is essentially similar to the timeline editing system employed by many existing commercial and research multimedia packages, but the micro level (state machine) authoring system is unusual, and so deserving of discussion.

The micro level editor operates by allowing the user to create and place states on a scrollable canvas. Once these are in place the author edits the poses that these states represent (using either dialog boxes or direct manipulation of the graphics

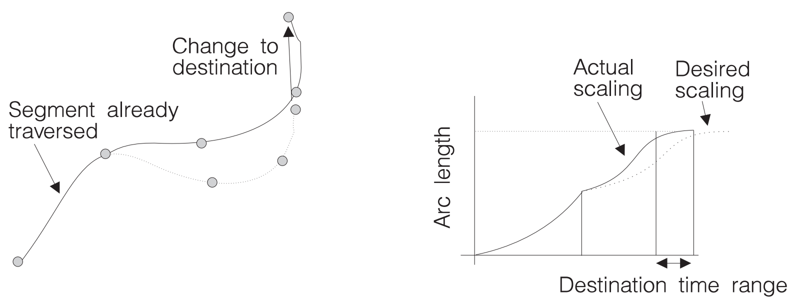

Figure 4 Here the distance to be travelled has increased considerably, but in order to meet our goal within the destination time we cannot use the desired scaling, and must only scale so we reach the end of the arc at the end of the time limit (shown as actual scaling).

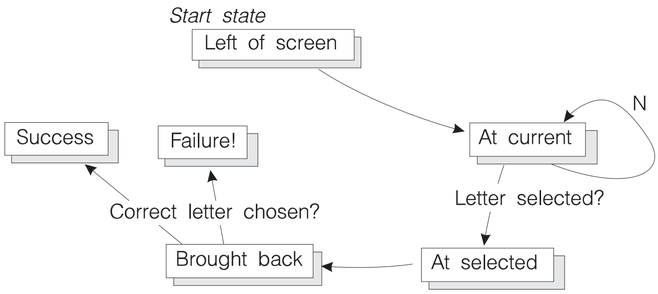

Figure 5 The micro level narrative for the hopping cup.

display where possible), thereby constructing the important ‘key-points’ of the narrative.

Once this is complete the author must connect the states to cause animation to be produced during playback. The user first selects the ‘current’ state, and then chooses the destination state for the new arc. A dialog box appears which allows the user to select the conditions under which this arc will be chosen (as discussed in section 4), and the particular motion generator. As the authoring program knows the destination it uses the classification scheme presented in section 5.1 to determine which (out of the selection of arcs) can be used in this context, and which the author may select. Once the arc is created (which will involve adding extra, arc-specific, information) the authoring program will use this as a constraint to prevent the destination state being edited in a way which would invalidate this arc. For example, if an arc with a fixed destination time is chosen, then the destination states’ time must remain fixed (though its value may change).

Whilst the authoring program may use the first classification scheme to only allow useful arcs to be selected, it is up to the author to select which of the categories identified in 5.2 is suitable. The authoring tool highlights which (of the available arcs) falls into which category, but the choice is made by the author.

6.3 The Playback

A full playback program, which was capable of allowing animation to be integrated with all existing media, would require the ability to use real-time video playback from multiple sources and high quality audio. As these were not available on the implementation platform (a HP 9000/735 workstation) a more basic, yet still useful, playback tool was produced which is capable of supporting animation (using the Starbase graphics API) and pre-recorded sound. The playback program operates by reading in a file which describes the narrative (which was produced using the authoring tool described in the previous subsection) and a scene [11], and then proceeds to play the presentation. At each time step the system examines the users actions to guide the animation. In the prototype implementation there are three mechanisms the system uses to achieve this.

Firstly the user may move the objects in the scene using a direct manipulation metaphor (the user selects an object using the mouse button, and drags it through the scene). Secondly the playback program associates an id with each object, and maintains a ‘currently_selected’ scene variable. When the user moves the mouse over an object the currently selected variable is changed to the relative id. Finally the user may initiate changes in state variables by right clicking on objects. When a user right clicks on an object the variable ‘currently_chosen’ is changed to the id of that object.

7. Example Interactive Presentation

As mentioned previously animation may be combined into multimedia presentations for a variety of reasons, and in this section we describe an example document which employs it for the purposes of enlivening an otherwise very straightforward ‘game’. This game forms part of a childs’ educational multimedia document, and is intended to teach the alphabet. In essence the computer selects a letter, and the child must choose which letter follows it in the alphabet sequence. To engage the childs’ interest the scene contains a 3D animated character, in this case a hopping coffee cup. When the child moves the mouse pointer over the collection of candidate letters the cup hops across the scene towards them. Once the child chooses a particular letter (using the right clicking mechanism discussed in the previous section) the cup carries the letter back to the left of the screen. If the selection was correct a fanfare is played, otherwise an ‘incorrect’ sound is made. To amuse the child sound effects are attached to the selection and hopping process. The number of possible routes that the cup might take through the scene preclude pre-recorded animations, and so make use of the ‘on-the-fly’ nature of this systems’ animation properties.

The macro level presentation is relatively straightforward: an FSM without any defined end time controls the hopping cup, a looping audio track with the background music is attached, and at the beginning of the game a pre-recorded voice-over explains the rules.

The micro level narrative, as shown in Figure 5, employs the variables described in section 6.3 to guide where the cup goes in the states ‘At Current’ and ‘At Selected’. The states are relatively straightforward, except for the ‘At Current’ state whose geometric goal is defined relative to the currently_selected variables’ object. The hopping motion is formed by a motion generator which replays a previously created sequence of movements of the NURBS model of the cup. Upon initialisation the arc examines its destination, splits this into a number of hops, and then performs these by playing back a NURBS hyper-surface [10] (with the relevant rotations) which contains the hopping movement. It only takes into account the destination state at the *beginning* of each hop (and so falls into the first category shown in section 5.2). This means that it is possible for the cup to hop towards a letter which the child has moved away from during the hop (hence the addition of the loop around the ‘At Current’ state).

Figure 6 shows a series of frames captured during the playback of the presentation, with the number at the top right of each panel indicating the frame number. The letters which the user can select are arranged on the right hand side of the screen, and are displayed as 3D text. When the user moves his or her mouse pointer over the letter it becomes the ‘current’ object, and the square on which it rests it highlighted. At the beginning of the presentation (Frame 1) the letter F is highlighted, so the cup starts to hop towards it. Shortly after frame 46 (which is during the cups’ second hop) the user moves the pointer to letter D. The cup waits until it lands, and then turns round and starts hopping towards the D (as shown in Frame 92). As it moves towards D the user selects the letter. Consequently, upon arrival at D, the state machine detects it has reached the ‘At Selected’ state, and so picks up the letter, and begins to hop back towards the left of the screen (as shown in Frames 276 & 322). When the cup reaches C (which corresponds to the ‘Brought Back’ state in Figure 5), the state machine determines that the correct letter has been fetched, which causes a move to the ‘Success’ state, the fanfare is played, and the micro level narrative terminates. If the author wanted the presentation to be repeated he or she could connect the success and failure states to the ‘Left of Screen’ state.

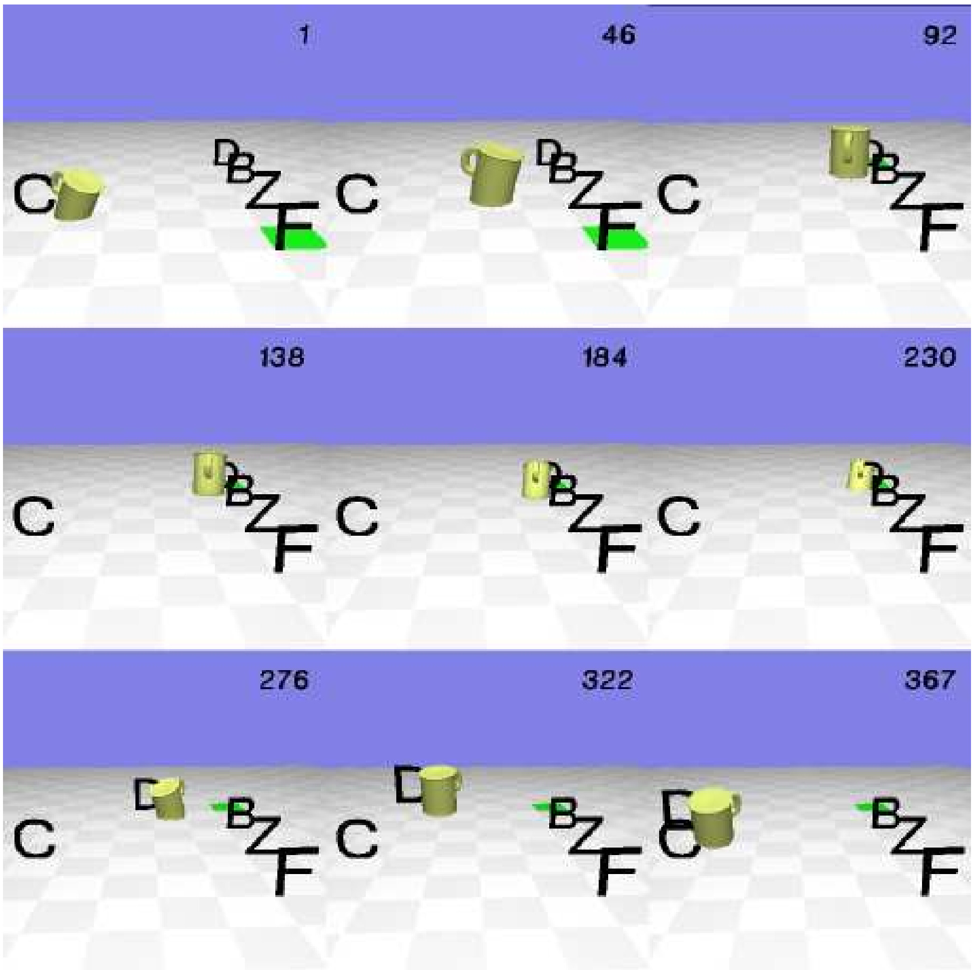

Figure 6 Frames captured from playing back the narrative.

8. Discussion and Conclusions

This paper has described a mechanism for incorporating interactive animation into multimedia presentations. This has been achieved in two ways. Firstly we have presented an authoring model which is capable of accommodating the control of a 3D virtual world, while still allowing existing media types to be included. Secondly we identified the role of motion generators in this system, i.e., the achievement of goals within narrative and author imposed constraints. To assist an author in construction of such a system we have described how such motion generators may be categorised, and have proposed a two stage classification scheme. We have detailed two motion generators designed for such systems (the keyframer in section 6.1, and the script replayer in section 7), and demonstrated how such generators may be used in a finished presentation.

The system provides advantages in three distinct ways. Firstly it allows existing multimedia presentations to be made more appealing by the addition of the ability to include 'on-the-fly' animation. Secondly it allows a multimedia author to produce presentations which would otherwise be impossible, or costly (requiring large quantities of pre-recorded motion) to generate. Finally, and perhaps ultimately more importantly, it provides the animation and multimedia communities with a mechanism for combining their skills in the generation of more interactive media.

This work has suggested several new lines of research which we are investigating. These include:

- The application of distributed computation, thereby reducing the resources required to support presentations.
- Development of more intuitive motion generators which are designed primarily for use by multimedia authors.
- The production of better mechanisms for encapsulating how the audience may interact with the presentation.
- Investigation of the use of a macro level which can change in response to the audience.
- The development of 'narrative debugging tools'. One of the side effects of development of complicated interactive storylines is the ability to accidentally miss certain possibilities. To assist artists in the production of larger narratives it is therefore important to produce development aids.

Acknowledgments

We would like to thank all the members of Manchester Computing's Computer Graphics Unit for their support and interest, and Martin wishes to thank the New Technologies Initiative of the Joint Information Services Committee (UK Higher Education Funding Councils) for its support. We are particularly grateful to Paul Lever for numerous discussions of this work.

References

[1] P. Ackermann. Direct Manipulation of Temporal Structures in a Multimedia Application Framework. In *Proceedings of ACM Multimedia '94*, San Francisco, 1994, pp. 51-58.

[2] F. Arbab, I. Herman and G.J. Reynolds. An Object Model for Multimedia Programming. *Computer Graphics Forum* (Proceedings of Eurographics '93) 12(3):101-113.

[3] N.I. Badler, C.B. Phillips and B.L. Webber. Simulating Humans : Computer Graphics, Animation and Control. Oxford University Press, 1993.

[4] N. Burtnyk and M. Wein. Computer-Generated Key-Frame Animation. *Journal of the Society of Motion Picture and Television Engineers*, 80:149-153, 1971.

[5] T. Little and A. Ghafoor. Synchronisation and Storage Models for Multimedia Objects. *IEEE Journal on Selected Areas in Communications*. 8(3):413-427, April 1990.

[6] J-D. Gascuel and M-P. Gascuel. Displacement Constraints : A New Method for Interactive Dynamic Animation of Articulated Solids. In *3rd Eurographics Workshop on Animation & Simulation*, Cambridge, 1992.

[7] D. Kalra and A.H. Barr. Modeling with Time and Events in Computer Animation. *Computer Graphics Forum* (Proceedings of Eurographics '92) 11(3):45-58.

[8] J. K. Ousterhout. Tcl and the Tk Toolkit. Addison-Wesley, 1994.

[9] M. van de Panne and E. Fiume. Sensor Actuator Networks. In *Proceedings of SIGGRAPH '93*, pages 335-342. In *Computer Graphics* Proceedings, Annual Conference Series.

[10] M. Preston and W.T. Hewitt. Animation Using NURBS. *Computer Graphics Forum* 13(4):229-241. 1994.

[11] M. Preston, N. Gatenby, W.T. Hewitt. The Manchester Scene Description Language (MSDL) V1.1, Technical Report, University of Manchester, CGU88.

[12] C. Reynolds. Flocks, Herds and Schools: A Distributed Behavioural Model. *Computer Graphics* 21(4):25-34. 1987.

[13] C. Rich *et.al.* Demonstration of an Interactive Multimedia Environment. *IEEE Computer*, 27(12):15-22. December 1994.

[14] N. Magnenat-Thalmann and D. Thalmann. The Use of High Level Graphical Types in the MIRA Animation System. *IEEE Computer Graphics & Applications* 3(9):9-16. November 1983.

[15] J. Wilhelms and R. Skinner. A Notion for Interactive Behavioural Control. *IEEE Computer Graphics & Applications*, pp. 14-22, May 1990.

[16] J. Wilhelms, M. Moore and R. Skinner. Dynamic Animation : Interaction and Control. *The Visual Computer*, 1988(4):283-295.

[17] A. Witkin and M. Kass. Spacetime Constraints. *Computer Graphics* 22(4):159-168. August 1988.

[18] R. Zeleznik *et.al.* An Object-Oriented Framework for the Integration of Interactive Animation Techniques. *Computer Graphics* 25(4):105-112. 1991.