DEVELOPMENT OF COMPUTER-ACTORS USING THE INTERVAL SCRIPT PARADIGM

T. A. Meyer & C. H. Messom

Institute of Information and Mathematical Sciences, Massey University

T.A.Meyer@massey.ac.nz, C.H.Messom@massey.ac.nz

Abstract: Computer-theatre is a relatively new concept involving the use of computers in the artistic process of a stage performance. One particular computer-theatre tool for writers and directors is the computer-actor – a computer program that assumes the role of a character (however basic) in the performance, similar to digital characters in film. Given the dynamic nature of stage performances, such computer-actors must be reactive and have a considerable amount of temporal flexibility within the script that they are performing. Dr. Claudio Pinhanez, while at M.I.T., developed the interval script paradigm to provide this flexibility. Interval scripts are based on James Allen's thirteen basic temporal constraints, and an interval engine processes a network of these constraints using *PNF propagation*. In the future, computer-actors will not only be able to follow their script, but take full part in the creative rehearsal process, improvising, and growing with their counterpart human actors.

Keywords: Computer-theatre, computer-actors, interval scripts

1 COMPUTER-THEATRE

Computer-theatre refers to live (real-time) theatrical performances that involve the active use of computers in the artistic process. It is "about providing means to enhance the artistic possibilities and experiences of professional and amateur actors, or of audiences clearly engaged in a representational role in a performance" [1]. The aim of all forms of computer-theatre is to provide an additional tool to writers and directors – to complement, not replace, the human elements. In [2] computer-theatre is broken down into four main categories – *computerized stages*, *hyper-actors*, *electronic puppets*, and *computer-actors*. This last category, where a computer program assumes the role of one (or more) of the characters in a play, is the subject of this research.

1.1 COMPUTER-ACTORS

Obviously a computer-actor will never completely take the place of a human actor, just as film will never take the place of live theatre. However, while computer-actors lack a lifetime's experience to draw upon, in certain situations they have significant advantages over their human counterparts.

Most importantly, computer-actors are not necessarily humanoid. Talking mirrors, aging portraits and the proverbial flying pigs are all character forms that would be difficult for a human actor to take on, but simple for a computer-actor.

Another advantage – one that particularly applies to amateur groups – is that computer-actors have no 'other life'. Family or work emergencies will not call them away from a rehearsal or performance; they'll never be late, and their performance will not be affected by their 'outside life'.

Traditional theatre is full of danger; computer-actors are able to remove some of these. As the laws of physics need not apply to a computer-actor (on a projected screen), such an actor may fly, explode, or melt away; while some of these effects may be possible using traditional theatrical 'tricks', using a computer-actor may be much simpler and safer.

Computer-actors may also be cast in small and uninteresting roles that would not be of much interest to a human actor, or in performances that call for a hundred six year old children to be on stage for five minutes (for example).

1.2 PREVIOUS USE OF COMPUTERS IN THEATRE

Very little use has been made of computers in the creative aspects of theatre, other than in lighting and sound systems. There have been a few productions, notably those at the Brooklyn Academy of Music and the University of Kansas [3]; however, these have generally only involved the use of virtual reality or other such displays, without involving any sort of computer-actor. The notable exception is the work at M.I.T.'s Media Lab by Dr. Claudio Pinhanez, Dr. Aaron Bobick and Dr. Kenji Mase. Pinhanez staged two main performances – *SingSong* (a short skit where a human conductor performs a song with four virtual singers), and *It/I* (a longer play about the interactions between man and machine).

While not focusing on live stage performances, the work of Dr Barbara Hayes-Roth on directed improvisation is relevant to development of computer-actors. Hayes-Roth developed a "Virtual Theater for Children" [4], with which children were able to direct characters within a virtual world. The characters were able to improvise the actions that they were directed to perform, as well as improvise action when without direction.

1.3 COMPUTER-ACTORS IN FILM

Characters resembling computer-actors, are, however, commonly used within the film industry. Over the last fifteen years, the integration of digital characters into live-action films has rapidly increased. Characters have progressed from the obvious cartoons in 1988's *Who Framed Roger Rabbit* to the stage where New Line Cinema can almost convince the public that the star of *Simone* [5] is a completely digital creation (possessing full intelligence) [6]. Typically, digital characters are not center-stage; for the three-film *Lord of the Rings* project, Weta Digital created MASSIVE (now a spin-off company [7]), which creates and manages thousands of digital extras that do not need to be scripted or animated because they have (reactionary) intelligence.

The computer-actors at the focus of this research differ from those in film in two main areas. Firstly, as these actors are required for live theatrical performances, they are required to operate completely in real-time. Additionally, digital characters in films are generally scripted or manually animated; even the software program MASSIVE, described above, provides only enough intelligence to control individual extras in scenes where the character is surrounded by thousands of others. This research aims to develop computer-actors that are much closer to human actors in their ability to act and improvise. These differences are fairly minor, however, and research and developments within computer-theatre are likely to be relevant to digital characters in film, and vice-versa.

2 INTERVAL SCRIPTS

The objective of Pinhanez's research was "the design and implementation of an interaction manager system … able to handle complex patterns of interaction evolving through time" [8]. As a result, he developed the interval script paradigm. An interval script outlines any number of intervals of time (during which something happens), and describes the temporal constraints that connect these intervals. The interval script language includes a wide variety of structures, and, using James Allen's thirteen basic temporal relationships (see Figure 1), any possible temporal constraint may be described [9]. However, an interval script is not simply limited to the structures within its grammar – programming code (such as C++) may be embedded into the script, effectively allowing an interval script to duplicate any action that a computer can perform [10].

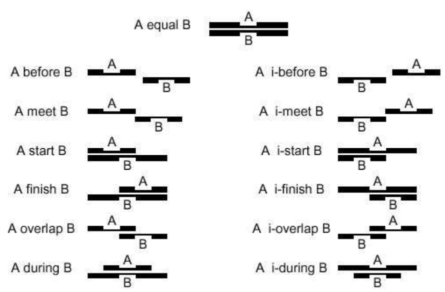

Figure 1: Allen's Thirteen Temporal Constraints

2.1 PNF NETWORKS

The basis of Pinhanez's interval script paradigm is the *PNF network*. Rather than using the temporal network that is created from Allen's constraints (algorithms for which are NP-hard), a network is created that restricts knowledge about an interval to whether it has happened (past), is happening (now), or is yet to happen (future), or some combination of these states. Pinhanez terms these Past-Now-

Future (or PNF) networks, and algorithms for these networks are solvable in polynomial time, allowing them to be used in a real-time computer-theatre system.

2.2 PNF PROPAGATION AND RESTRICTION

An interval engine performs four main steps during each loop of operation. First, the current PNF state of the network is time-expanded (past => past, now => past | now, future => now | future). The PNF Restriction process then is applied to the resulting PNF state; this process uses an arc-consistency algorithm adapted by Pinhanez (essentially, a set of possible states is determined, based on the time-expanded state, and the temporal constraints between the intervals). This set of states is then thinned down into one selected state, and appropriate action is taken – depending on the differences between the next state and the previous one, intervals are started or stopped.

2.3 ALTERATIONS FROM PINHANEZ INTERVAL SCRIPTS

A number of changes have been proposed to Pinhanez's final implementation of the interval script language [11]. Some of these result from inaccurate or unclear sections of his grammar, while others are made in an attempt to improve the language. The most significant difference is the conversion of the interval script language to XML. Formatting interval scripts in XML offers many advantages over Pinhanez's punctuation-heavy method; in particular scripts become much more modular, more easily expandable, and easier to develop from 'traditional' scripts (particularly if the original scripts are also in XML). The main drawback to XML is the verbosity of the script (although in some circumstances this is an advantage).

3 COMPUTER-THEATRE AT MASSEY UNIVERSITY

In 2001, Massey University began to experiment with computer-theatre. While there are few tangible results at present, it is hoped that within the next three years Massey will have a fully functional computer-theatre system.

3.1 CURRENT STATE

As part of an honour's degree project, the author developed a basic computer-theatre system, based on Pinhanez's interval script paradigm. Given the time and financial constraints, an input system (such as a vision or audio recognition system) was not developed; input was simulated via mouse clicks. Graphical output was created using simple models developed with Curious Lab's Poser, and managed by the 3d development program WorldUp (from Sense8). At the heart, an interval engine processed interval scripts; based on the simulated input, and the temporal constraints in the scripts, appropriate commands were sent to WorldUp, resulting in changes in the graphical output.

Three small scripts were developed as part of this project. *SingSong*, based on one of Pinhanez's first scripts, *How Maui Fished Up New Zealand*, a retelling of the traditional Maori myth, and (as an experiment in design simplicity), a scene from the stage musical *Footrot Flats*. While all three were successfully scripted, and both *SingSong* and *Maui* had graphical output designed, much more work remains to be done before a stage performance could be presented to an audience at Massey University.

3.2 FUTURE RESEARCH

Over the next year, a basic computer-theatre system will be constructed at Massey University's Albany campus. The system will include a graphical output sub-system (most likely utilising the Quake or Unreal Tournament 3d graphics engines [12]) and one or more sensory input sub-systems (most likely a form of vision recognition and possibly some form of audio recognition). At the heart of the initial system will be a basic interval script engine, much like that used by Pinhanez.

The computer-theatre system will be highly modular, allowing use of different sensor systems and actuators. The aim is to develop a central control system that is reusable by any theatre organization wishing to experiment with computer-actors.

Hopefully, before the end of 2004, a computer-theatre performance, including at least one computer-actor, will be performed at Massey University, providing not only entertainment, but also a look into the future of theatre.

3.3 FUTURE COMPUTER-ACTORS

While computer-actors based on Pinhanez's interval scripts have many advantages, particularly in temporal flexibility, the intelligence of the computer-actors is reactive only. A truly useful computer-actor needs to interact with the other members of the cast, and with the director. Like Hayes-Roth's *Improv Puppets* [13], they need to accept directed improvisation, and fill in the gaps between directives.

The 'users' of computer-actors will not be technically savvy programmers or engineers, but more traditionally creative people – scriptwriters, directors and other thespians. As a result the inclusion of a computer-actor in a performance must be both simple and intuitive.

A large part of theatrical stage performances is the rehearsal process. While the script for the performance is generally known before rehearsals begin, much of the performance is developed through the interaction of the actors and director in rehearsals. Future computer-actors must also have the capacity to rehearse, including offering new ways of acting and responding to small changes in the actions of other cast members.

4 CONCLUSION

Computer-theatre is a young technique, but promises to be an exciting new tool for creative directors and writers (not only in live stage performances, but for interactive exhibits, film, and other areas). The interval script paradigm provides an excellent base on which to develop computer-actors, as great temporal flexibility is provided, in polynomial time, without sacrificing simplicity. Much work remains to be done, however, particularly developing computer-actors so that they are able to be useful members of a rehearsal process.

5 REFERENCES

1 Pinhanez, C. (1996). *Computer Theatre* [electronic version]. M.I.T. Media Lab Perceptual Computing Section Technical Report No. 378.

2 Pinhanez, C. & Bobick, A. (1998). *Fast Constraint Propagation on Specialized Allen Networks and its Application to Action Recognition and Control* [electronic version]. M.I.T. Media Lab Perceptual Computing Section Technical Report No. 456.

3 Reaney, M. (1995). *The Adding Machine: A Virtual Reality Project*. Retrieved 12th May 2002 from http://www.ukans.edu/~mreaney/machine/

4 Hayes-Roth, B., Brownston, L. & Sincoff, E. (1995). *Directed Improvisation by Computer Characters*. Stanford University: Knowledge Systems Laboratory Technical Report KSL-95-04.

5 *Simone* (2002). Retrieved 2nd September 2002 from http://www.s1m0ne.com

6 Turner, M (2002, August 20). *A Star Is Made*. New York Post. Available http://www.nypost.com/entertainment/55114.htm

7 *Massive* (2002). Retrieved 2nd September 2002 from http://www.massivesoftware.com/massive.html

8 Pinhanez, C., Mase, K. & Bobick, A. (1996). *Interval Scripts: a Design Paradigm for Story-Based Interactive Systems* [electronic version]. M.I.T. Media Lab Perceptual Computing Section Technical Report No. 391.

9 Allen, J. (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the A.C.M.*, 26 (11), 832-843.

10 Pinhanez, C. (1999). *Representation and Recognition of Action in Interactive Spaces* [electronic version]. Unpublished Doctoral Dissertation. M.I.T. Media Lab, Massachusetts.

11 Meyer, T. (2002). *Development of Computer-Actors within the Interval Script Paradigm*. Unpublished Honours Dissertation. Massey University. Available http://www.massey.ac.nz/~tameyer/research

12 Lewis, M & Jacobson, L. (2002). Game Engines in Scientific Research. In *Communications of the A.C.M.*, 45:1.

13 Hayes-Roth, B. & van Gent, R. (1995). *Story-Making with Improvisational Puppets and Actors*. Stanford University: Computer Science Department.