A CCNVERSATIONAL EXTENSIBLE SYSTEM FOR THE ANIMATION OF SHADED IMAGES(*)

by

Ronald M. Baecker
Dynamic Graphics Project, Computer Systems Research Group
University of Torcnto, Toronto, Ontario, Canada

Abstract

The terms "conversational" and "extensible" are defined and shown to be useful properties of computer animation systems. A conversational extensible system for the animation of shaded images is then described. With this system, implemented on a minicomputer, the animator can sketch images and movements freehand, or can define them algorithmically via the Smalltalk language. The system is itself implemented in Smalltalk, and hence can be easily extended or mcdified to suit the animator's personal style.

(*) This work was supported primarily by the Xerox Palo Alto Research Center.

I. Introduction

Computer animation consists of a variety cf techniques and processes in which the computer is used as an aid in the production of animated sequences [Halas 74, Wein 74]. Computer animation systems can generally be classified as either algorihmic or demonstrative [Tilson 75].

1. Programming-language based systems (Algorithmic systems)

EEFLIX [Knowlton 64], EXFLOR [Knowlton 7C], and ZAPP (Guerin 73, Eaecker 76], reuire the animator to describe a movie in a written programming language. These systems generally are not interactive; the arimator is provided no direct visual feedback.

2. Interactive systems based on freehand sketching (Demcnstrative systems)

GENESYS [Eaecker 69a,69b,70a], ARTA [Mezei 71], and the Eurtnyk-Wein system [Eurtnyk 71a,71b] allow the animator to sketch images and movements free-hand. These systems generally provide immediate real time playback of the resulting movie.

Permission to make digital or hard copies of part or all of this work or personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee.
Siggraph '76, July 14-16 Philadelphia, Pennsylvania

We shall now focus upon the second category cf systems. The strengths of GENESYS lay in the spontaneous, real time interaction it facilitated, and in the ccnceptualization of the computer animated film and the filmmaking process that it embodied. The heart of this conceptualization was a duality between image and movement, and a rich set of representations for movement and tools for the construction of movement. The ARTA and Eurtnyk-Wein systems demonstrated the utility of various picture construction and transformation tools, including the ability to interpolate between images (key frame animation).

By 1969, one could identify several major weaknesses in these systems:

1. They were rigid, difficult to modify and extend.

2. The animator could draw images and movements, but could not compute them directly; he had a fixed set of drawing commands, not an open-ended programming and drawing system at his fingertips.

3. Image quality was poor, consisting of black-and-white dot and line drawings, lacking levels of tone, texture, and color.

In the pericd 1969-1974, there were several useful developments addressing these problems. Burtnyk and Wein focused upcn the problem of image quality [Burtnyk 73], succeeding to the point that Peter Fcldes was able to make his Cannes award-winning film HUNGER [Foldes 74]. Richard Shcup and Robert Flegal at the Xerox Palo Alto Research Center (PARC) developed a rich color video system, and produced numerous striking examples of directly sketched or ccmputed color video [Shoup 74]. In Alan Kay's Learning Research Group (IRG) at Xerox PARC, the Smalltalk language was developed and shown to be a viable and congenial host for the development of user-responsive programs [Kay 72,74; LRG 76]. Experiments with freehand painting programs demonstrated the viability of black and white TV generated by a minicomputer as a medium for shaded drawings. Finally, the now legendary Pegasus-Cookie Monster movie produced using Steve Purcell's playback process (described below) demonstrated that this same hardware could be used for the real time animation of these drawings.

Thus, encouraged by these prior developments, and guided by a previous vision [Baecker 69a], until now unfulfilled, of what it would like to build arimation systems in an appropriate language, we developed a new animation system at the Xerox Palo Alto Research Center in the summer of 1974. This system successfully incorporates Purcell's real time shaded image animation capability into LRG's Smalltalk programming and drawing environment. SHAZAM (Smalltalk's sHaded image Zippy Animated Moviemaker) is a demonstrative picture-driven animation system modeled directly upon GENESYS [Eaecker 69a,69b,70a,74]. Hence the animator need know no Smalltalk. However, the system was designed for children ages 8 to 15 who are learning Smalltalk [Goldberg 74], and they can compute images and movements by executing Smalltalk commands or by writing Smalltalk programs as easily as they can sketch them freehand. Furthermore, a skilled Smalltalk programmer can extend or mcdify the system himself.

We shall next define the terms "conversational" and "extensible", and discuss the relevance of these language features to computer animation. These arguments constitute a concise and sharper presentation of points originally made in [Eaecker 69a]. Key aspects of the design and implementation of SHAZAM will then be presented; the relevance of ccnversaticnality and extensibility will be stressed, and some directions for future research proposed.

II. Cn Conversaticnality and Extensibility in Animation Systems

In concluding its discussion of picture-driven animation, and in motivating the design of the Animation and Picture Processing Language (APPL), Eaecker [69a] noted:

"We have seen that there are advantages and disadvantages to each of several approaches to the definition of dynamic pictures--the construction of individual frames, the algorithmic generation of sequences of frames, and picture-driven animation. This suggests that a flexible animation system would allow the harmonious blending of all these techniques. Here GENESYS fails a priori, fcr it makes inaccessible the full ccmputaticnal power of the computer. within the language of GENESYS cne cannot implement algorithms by writing programs. We have also seen that GENESYS is inadequate because it presents the animator with a fixed set of commands and tools, with fixed mechanisms cf control, and with fixed models cf pictures and of processes of picture construction. A suitably skilled animator may himself determine these aspects cf his animation system, his arimation-machine, only if the system is not a fixed set of commands but an extensible, truly open-ended programming language."

The design of APPI, and the construction of SHAZAM in Smalltalk, were intended to demonstrate the feasibility and attractiveness of unifying algorithmic and demonstrative computer animation capabilities in one system. (A similar goal is pursued by [de Fanti 73].) For this tc be possible, APPL had to be (and Smalltalk is) conversational and extensible. These terms are much used and misused in the literature, so we shall now define our use of them quite precisely.

A. Ccnversationality

The terms "conversational", "on-line", and "interactive" are often used interchangeably when applied to computer systems. our usage shall be more restrictive. A conversational language is one in which:

1. "Response time is proportional to the demand for computation and, in particular, there is rapid response tc trivial requests" [Standish 70].

2. The language is conveniently extendable, that is, program's may be written in the same language with which cne expresses direct commands to the system, and they may be immediately tested.

3. The animation environment or data base is directly accessible at all times.

The response time criterion is important for computer animation because of the frequency with which animators make and wish to preview the effect of small changes tc their movies. Such changes typically include speeding up or slowing down a motion, introducing or removing an object a split second earlier or later, and making slight pcsitional changes to an object. Compiled languages operating in card-oriented slcw-turnaround batch systems (the antithesis of a conversational system) impede the animator's ability and desire to carry out such refinements to a movie.

The extendibility criterion is important for computer animation because demonstrative systems rarely have the perfect command set and the perfect interaction style for every animator. Although a system can be changed even if the implementation language is very remote from the user command language, the dialogue between animator and animation system programmer can be enhanced if the twc languages are the same. Furthermore, the animator can ultimately learn to make simple additions and modifications by himself.

The environment accessability criterion is important for computer animation because animation computations often yield interesting pictures or movements as by-products. It is of course possible to anticipate these and explicitly modify any animation program to write them onto a file. But such results are often unanticipated, even the result of accidents, and one wants to be able to access them at any time using the same naming conventions as are used by the animation program.

Conversational systems satisfying these criteria include most LISP, LOGO, API, and BASIC implementations, and Smalltalk. These systems generally are interpretive, but they need not be; the same results could be achieved by incremental compilation.

B. Extensibility

One recent definition of extensibility has been provided by Cheatham [71]:

"By extensible programming language we mean a base (or core) language plus a programming system the totality of which has facilities for extending or modifying:(1) the syntax of programs; (2) the operators available; (3) the data structures and internal representations of data structures which can be defined, given values, manipulated, output, and so on: and, (4) the regimes of control which can be employed".

Extensibility of operators, data structures, and regimes of control are relevant to computer graphics in general and to computer animation in particular [Eaecker 70b, Standish 70].

A data definition facility in an extensible language enables the specification of a composite data type (data structure) built out of primitive data types and/or other composite data structures. The extended language must contain mechanisms for constructing members of the new class from suitable components (constructors), mechanism for selecting distinguished components (selectors), and mechanisms for testing if an arbitrary datum belongs to the new class (predicates). Standish [67] presented a formalism and a methodology for augmenting a language with a data definition facility so that, given a new data structure definition, the system can automatically provide the associated constructors, selectors, and predicates. Furthermore, these constructors, selectors, and predicates are used in formulating extensions of the meaning of old operators so that they apply to new data types. (Adding new operators can be done via function definitions in most procedural languages, although these language enhancements may be made more gracefully if coupled with the syntax extension capabilities of an extensible language.)

In computer animation we often encounter classes of pictures or data which possess unique characteristic features, and specialized techniques for their construction, decomposition and manipulation. P-curves, waveforms, selection description, and rhythm descriptions [Baecker 69a,69b] are some examples. An extensible animation language allows us to define these new concepts gracefully while remaining within the same language.

The same kind of argument applies to extensibilityof control structure. One good example is quasi-parallel processing. Simulation is the mathematical dynamic modelling of a hypothetical or real system; animation is the pictural dynamic modelling of a hypothetical or real system [Baecker 69a]. Hence an animation language must allow several strands of activity to proceed concurrently, synchronously or asynchronously. Similar cases can be made fcr other control regimes such as semaphores [Horsley 74, Duff 76] and continuously evaluating expressions. As with data structures, greater flexibility can be achieved if new control structures can be defined by extending a set of control primitives with an appropriate definition facility [Fisher 70].

III. A Conversational Extensible Animation System

A. The Essential Features of Smalltalk

[Goldberg 75] describes Smalltalk as a programming language that:

"...permits detailed examinations of the processes involved in carrying out a task. It is a communication medium based on sending and receiving messages among objects. No exceptions exist to this notion. Everything in Smalltalk is an object; actions occur as a result of sending a message to an object; an object understands a message according to the way it receives the message; and objects are grouped in classes because of the similarity of their description and the sameness of the actions they can take. Each member of a class remembers information that distinguishes it from the other members of the class, but each member receives and sends messages according to the class definition".

Any class definition in Smalltalk follows the same skeletal pattern:

```
to class_name tempcrary_vars
object_vars 1 class_vars
(acticns of_any_member_of_the_class)
```

Class variables are those whose meaning (binding) applies to all objects of the class, object variables are those whose meaning is unique to a particular object (instance of the class), and temporary variables are those that are created and destroyed anew in any activation of any object of any class. When an object is activated (fired up, called), it carries out the actions contained in the body of the class definition.

The concepts of class and object embodied in Smalltalk are similar to (in fact, inspired by) those contained in SIMULA [rahl 72]. But the concept of sending and receiving messages is significantly different from that of passing arguments and binding them to procedure parameters.

Each class definition contains specifications which determine when it chooses to receive a message and how it will interpret that message. In particular, it may receive a message literally (unevaluated), it may receive and evaluate a message in its own environment, or it may receive and evaluate a message in the environment of the message transmitter (retrieve a binding from that environment). These mechanisms fcr controlling evaluation are similar to but more general than those provided in LISP.

Like LISP and LOGO, Smalltalk is ccnversational. Smalltalk furthermore shares with LISP the property that Smalltalk program segments (code vectors) are data objects within the language. Thus Smalltalk programs can easily construct or modify other Smalltalk programs, which is a tool of great power (although perhaps one easily misused). Smalltalk also shares with ICGO some properties that facilitate its use by children, who are the prime intended users of both languages. These features include a straightforward syntax requiring no declarations, and the simple graphics sublanguage turtle geometry. Experiences with both languages demonstrate that children begin composing procedure definitions (class definitions) in their first hour on the machine.

E. SHAZAM's Basic Capabilities and Interaction Style

SHAZAM contains only the most essential animation capabilities. This is in order to facilitate its learning by children, who use it for making small movies. Another reason is that more scphisticated features can easily be added by extension.

Static images in SHAZAM are called cels. They resemble the cels cf. the animation industry, in that they behave like clear sheets of celluloid which can be "painted" and overlaid. Commands exist to paint cels, to erase them totally, to copy a cel, and to cycle through all available cels for review and possible insertion into a movie. Cels are painted using "brushes" which can have different shapes and which can be "dipped" into different "paints". Commands exist to select one of a number of brushes (such as a pin or a blob), and one of a number of paints (such as black, grey, white, or transparent). The effect of these brushes and paints may be seen in some typical SHAZAM drawings shown in Figures 1 and 2.

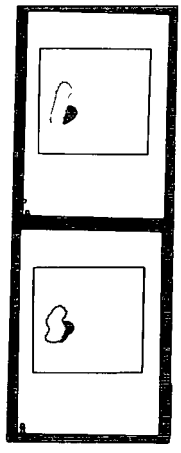


Figure 1

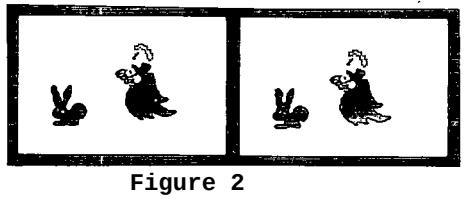


Figure 2

Dynamic images in SHAZAM are called movies. A movie consists of a sequence of cel selections (selection description), which determines which cel is visible in which frame, and a sequence of positions (F-curve), which determines where the selected cel is located in each frame. Commands exist to sketch a p-curve, to select a cel fcr a particular frame, to single step through the movie (advance one frame at a time), and to play back the movie (cycle it continuously). As was true using GENESYS, the SHAZAN animator can create a movie such as the dripping faucet of Figure 3 in 5 minutes of work with the p-curve, selection description, single step, and real-time playback capabilities.



Figure 3

Every cel and every movie is affiliated with its own unique window. Commands exist to reposition both the cel (movie) and window relative to the screen, tc move the window relative to the cel (mcvie), and to expand or shrink the window. Cels and movies are viewed through their windows, which are drawn as dark black outlines and which occlude material falling outside their boundaries. Composite movies such as those in Figures 4 and 5 may be formed by activating individual movies, positioning them on the screen, and controlling their appearance by changing the location and size of their surrounding windows.

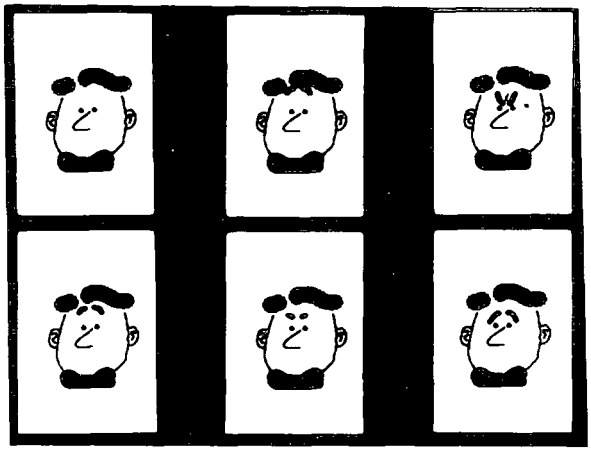


Figure 4



Figure 5

SHAZAM continues to execute in parallel with the playback process as it displays one or more movies. Thus the sketching of a cel can occur concurrently with a playback of a movie in which that cel is used. Defining a blinking pair (two frame movie) using this feature provides a vivid demonstration of the Phi Phenomenon on which the illusion of animation is based. One constructs, for example, a two frame movie consisting of a bent arrow and an undefined cel, fires up the movie playback process, and then begins to sketch a straight arrow as the second cel. At first the movie is only a mechanical alternation between two static images, but, as the second cel begins to take shape, the movie suddenly comes to life as an energetic flexible arrow. (Figure 6)

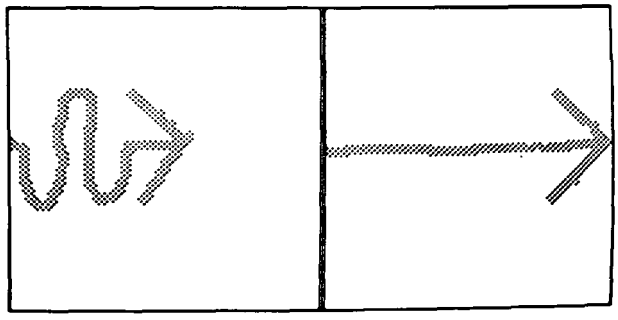


Figure 6

Aside from typing in movie names so that they may be saved on a disk, all SHAZAM input is given from a hardware pointing device such as a stylus. Cel and movie windows each contain a menu of command icons (light buttons). Pointing at a particular icon causes the associated ccmmand to be activated. For example, pointing at a tiny palette activates painting; pointing at a tiny staircase causes the movie to single step. Each window's command icons appear (disappear) when the stylus enters (leaves) that window. Positioning or scaling of windows is done by "dragging" them with the stylus. All windows also have a "vertical location"; this layering causes some pcrticns of pictures to be occluded by others. A window can be "brought to the top" by depressing the tip of the stylus while it is within that window but not pointing at any command icon.

C. The SHAZAM Playback Processor

Use of SHAZAM results in the ccnstructicn of a tree of Smalltalk objects. These objects are instances of some Smalltalk classes all of which respond tc the message 'display'. This tree structure may be regarded as a display file. There is also a special program which accepts a pointer to the root of the tree and passes the message 'display' down it; this program may be considered a display processor.

Primitive pictures in SHAZAM are defined as 256x256 rasters of points, each black, white, or transparent. They are stored in a highly compact form, using a hierarchic area encoding scheme, so that a number of pictures can fit into memory together. The display processor traces the tree structure, scan converts, translates clips, and does transparency-opacity calculations (to determine which portions of which layers are visible) in real time, approximately 3 to 10 frames per second. The results of these calculations are double buffered to the video display to enhance the perception of movement.

D. Conversationality in SHAZAM

Smalltalk is an interpretive system implemented on a stand-alone minicomputer; it provides "rapid response to trivial requests". Hence it meets the response time criterion for a conversational system. The extendibility and environment accessability criteria are also satisfied; their importance to SHAZAM, however, is somewhat subtler.

The extendibility criterion is perhaps best discussed by relating an anecdote. The incident occurred while Eric Martin, a non-programming animator, was working with SHAZAM shortly after it became operational. Eric requested that SHAZAM be modified to allow one to replicate a cel in multiple frames of a movie with a single command. This is particularly useful if one is ccnstructing cyclic motions. Because of Smalltalks's convenient extendibility, I was able tc add this feature in fifteen minutes. Furthermore, I was able to explain what I was doing, enhancing Eric's appeciation of possible system malleability in terms of his nascent understanding of Smalltalk. (Note: It is essential that a computer animator develop an ability to sense which aspects of a system's limitations are arbitrary and which are fundamental.) I contrast this experience to similar incidents with the use of GENESYS, in which I could add such a feature only overnight, and in which it was impossible tc explain to the animator what I was doing in terms of what he was doing.

The role of environment accessability for SHAZAM is equally dramatic. In GENESYS every access path to the animation data base had to be pre-planned and pre-programmed. These included such trivial functions as: retrieve a picture, given its name; retrieve the X coordinate of a particular cel in a particular frame of a particular movie; and, reverse the direction of movie playback. These functions, and many more, all come "for free" in SHAZAM because they may be expressed simply and directly in terms of the Smalltalk evaluator's mechanisms for retrieving and establishirg bindings. Furthermore, by cleverly including calls to the Smalltalk evaluator at appropriate points within interesting class definitions, one can interrogate and modify the local state of a particular instance of that class. The experienced SHAZAM animator can exploit this to explore and refine individual movies.

E. The Extensibility of SHAZAM

Available space precludes the presentation of a complete description of the method and style of Smalltalk extensions. Extensions of the operators, of the syntax, of data structures, and of ccntrol structures, may all be achieved. The Smalltalk class concept, like that of SIMULA, subsumes both procedures and data structures as they appear in conventional languages. Hence new data structures may be defined as easily as and in the same manner as new functions.

The design of the final version of SHAZAM followed a pattern that seems to characterize the development of many Smalltak programs. First, the essential Smalltalk classes, such as the cel, the movie, the window, and the menu, were identified. Next,the necessary local state of each instance of each class was specified, and appropriate instance variables defined. Then, the messages appropriate to each class were formulated. Finally, code carrying out the response to those messages was written.

IV. Conclusions

One particularly useful consequence of Smalltalk conversationality and extensibility is that an animation system designer way formulate, implement in preliminary fashion, test the viability of, and, if need be, discard numerous design ccncepts. The version of SHAZAM described in this paper is literally the fourth that we constructed in three man-months of effort. Each of the first three implementations explored a point of view with respect to embedding picture-driven animation within Smalltalk. Although none were completed, each trial implementation yielded useful insights which were ultimately incorporated into the final design. I know of no other environment where design ideas can be explored with ccmparable grace and fluidity.

luckily, SHAZAM is a relatively small system, because large systems constructed in this environment often run slowly, and must therefore be reprogrammed in less powerful but more efficient languages. The utility of conversational extensible graphics languages will obviously be enhanced if appropriate tools for performance measurement and analysis and for compilation are developed.

Another area for future research is the development of more general techniques fcr coupling free-hand and algorithmic specifications in computer animation. SHAZAM takes a step in this direction by allowing both cels and p-curves to be sketched or computed, but the underlying playback process is fixed, inaccessible to the Smalltalk programmer. Animation computation and playback processes should be expressible with all the powers of the conversational extensible graphics language; techniques for combining these processes and for enabling them to compute efficiently must be developed.

Acknowledgements

The implementation of SHAZAM was carried out by the author and Tom Horsley while working in the Learning Research Group (LRG) of the Xerox Palo Alto Research Center. Eric Martin provided many useful ideas and suggestions, and was the system's first user. The very elegant design of the windows is Tom's, inspired in part by a previous implementation of Alan Kay's. The display processor is the work of Steve Purcell, also inspired (I believe) by a suggestion of Alan Kay's. We are especially grateful to Steve and to Dan Ingalls, who came to our rescue on countless occations. We also thank Adele Goldberg, Chris Jeffers, Diana Merry, John Shoch, and Steve Weyer for their support, and Alan Kay for his inspiration and vision which permeates LRG.

References

[Eaecker 69a] Ronald M. Baecker, Interactive Computer Mediated-Animation, MIT Project MAC-TR-61, 1969.

[Eaecker 69b] Ronald H. Baecker, "Picture-Driven Animation", Proceedings of the 1969 Joint Computer Conference, 273-288.

[Paecker 70a] Ronald M. Eaecker, Lynn D. Smith, and Eric Martin, "GENESYS: An Interactive Computer-Mediated Animation System", 17 minute color sound film, MIT Lincoln Laboratory, Lexington Massachusetts, 1970.

[Eaecker 70b] Ronald M. Baecker, "Current Issues in Interactive Computer-Mediated Animation", Proceedings of the Ninth Annual Meeting of the Users of Automatic Information Display Equipment, October 1970, 273-288.

[Baecker 74] Ronald M. Baecker, "GENESYS--Interactive Computer-Mediated Animation", appears in [Halas 74], 97-115.

[Baecker 76] Ronald M. Baecker, Marjorie Guerin, and Michael D. Tilson, "An Algorithmic Computer Animation Facility for Research and Educational Filmmaking", 1976, in preparation.

[Burtnyk 71a] Nestor Burtnyk and Marceli Wein, "Computer Generated Key-Frame Animation", Journal of the Society of Motion Picture and Television Engineers, Vol.80, March 1971, 149-153.

[Eurtnyk 71b] Nestor Burtnyk and Marceli Wein, "A Computer Animation System for the Animator", Proceedings of the Tenth Annual Meeting of the Users of Automatic Information Display Equipment, October 1971, 3.5-3.24.

[Eurtnyk 73] Nestor Burtnyk and Marceli Wein, "Image Quality Considerations in Computer Animation", Proceedings of the 3rd National Research Council Man-Computer Communication Seminar, May 1973, 20.1-20.8.

[Cheatham 71] Thomas E. Cheatham, Jr., "The Recent Evolution of Programming Languages", Proceedings of the 1971 International Federation of Information Processing Societies Conference, 298-313.

[Dahl 72] Ole-Johan Dahl and C.A.R. Hoare, "Hierarchical Program Structures", Structured Programming, A.P.I.C. Studies in Data Processing No.8, Academic Press, 1972, 175-220.

[de Fanti 73] Thomas A. de Fanti, The Graphics Symbiosis System--An Interactive Mini-Computer Animation Graphics Language Designed for Habitability and Extensibility, Ph.D. Thesis, Dept. of Computer and Information Science, Ohio State University, 1973.

[Duff 76] Thomas D.S. Duff, Simulation and Animation, M.Sc. Thesis, Dept. of Computer Science, University of Toronto, 1976.

[Fisher 70] David A. Fisher, Control Structures for Programming Languages, Ph.D. Thesis, Dept. of Computer Science, Carnegie-Mellon University, 1970.

[Foldes 74] Peter Foldes, "La Faim (Hunger)", Color Sound Film, The National Film Board of Canada, 1974.

[Goldberg 74] Adele Goldberg, "Smalltalk and Kids--Commentaries", Xerox Palo Alto Research Center Report XPARC-LRG-3, June 1974.

[Goldberg 75] Adele Goldberg and Bonnie Tenenbaum, "Classroom Communication Media", Topics in Instructional Computing, ACM Special Interest Group on Computer Uses in Education, Volume 1, January 1975, 61-68.

[Guerin 73] Marjorie Guerin, A System for Ccmputer Animated Film Production in a Batch Processing Environment, M.Sc. Thesis, Dept. of Computer Science, University of Toronto, 1973.

[Halas 74] John Halas(Editor), Computer Animation, Hastings House, New York, 1974.

[Horsley 74] Thomas R. Horsley, SIMULOGO: A Student Simulation Language, M.Sc. Thesis, Dept. of Computer Science, University of Toronto, 1974.

[Kay 72] Alan Kay, presentation at 1972 ACM National Conference, Boston Mass., August 1972

[Kay 74] Alan Kay, presentation at First Annual Conference on Computer Graphics and Interactive Technique, Boulder Colorado, July 1974.

[Knowlton 64] Kenneth C. Knowlton, "A Computer Technique for Producing Animated Movies", Proceedings of the 1964 Spring Joint Computer Conference, 67-87.

[Knowlton 70] Kenneth C. Knowlton, "EXPLOR - A Generator of Images from Explicit Patterns, Local Operations, and Randomness", Proceedings of the Ninth Annual Meeting of the Users of Automatic Information Display Equipment, October 1970, 543-583.

[LRG 76] Learning Research Group, Personal Dynamic Media, Xerox Palo Alto Research Center Report, 1976.

[Mezei 71] Leslie Mezei and Arthur Zivian, "ARTA, An Interactive Animation System", Proceedings of the 1971 International Federation of Information Processing Societies Conference, 429-434.

[Papert 73] Seymour Papert, "Uses of Technology to Enhance Education", MIT Artificial Intelligence Laboratory Memo 8, June 1973.

[Shoup 74] Richard Shoup, presentation at First Annual Conference on Computer Graphics and Interactive Technique, Boulder Colorado, July 1974.

[Standish 67] Thomas A. Standish, A Data Definition Facility for Programming Languages, Ph.D. Thesis, Dept. of Computer Science, Carnegie-Mellon University, 1967.

[Standish 70] Thomas A. Standish, "Remarks on Interactive Computer-Mediated Animation", Proceedings of the Ninth Annual Meeting of the Users of Automatic Information Display Equipment, October 1970, 306-309.

[Tilscn 75] Michael D. Tilson, Editing Computer Animated Film, M.Sc. Thesis, Dept. of Computer Science, University of Toronto, 1975.

[Wein 74] Marceli Wein and Nestor Burtnyk, "Computer Animation", to appear in *Encycopedia of Computer* Science and Technology, Volume 3.