Small Computers in the Arts Network

SCAN

Proceedings of the Eleventh Annual symposium on small computers in the arts

Presented by:
Small Computers in the Arts Network
The Pennsylvania Council on the Arts
The University of the Arts

November 15-17, 1991
The University of the Arts
Broad and Pine Streets
Philadelphia, Pennsylvania

Small Computers in the Arts Network

SCAN

Proceedings of the Eleventh Annual Symposium on Small Computers in the Arts

November 15-17, 1991

The University of the Arts
Broad and Pine Streets
Philadelphia, PA 19102

Presented by:
Small Computers in the Arts Network
The Pennsylvania Council on the Arts
The University of the Arts

1991 Proceedings of the Eleventh Annual Symposium on Small Computers in the Arts

Presented by:
Small Computers in the Arts Network
The Pennsylvania Council on the Arts
The University of the Arts

Sponsors:
Ensoniq
Desktop Publisher
Delaware Valley Computer User
The School of Visual Arts, New York City, NY
Moore College of Art and Design, Philadelphia, PA
Hofstra University, Hempstead, NY

SCAN Activists:
Dick Moberg, President
Professor Thomas Porett, University of the Arts Liason
Misako Scott, Conference Coordinator
Brian Souder
Charles Lutz
Steve Beuret
Julie Shay

Newsletter & Proceedings Editor: Mark Scott

Cover Graphic: Lillian Schwartz

Thanks to: Ina Marcus, The University of the Arts, External Affairs

ISBN 0748-2043

Sales Information: This book is sold at a cost of $15.00 per copy.

TABLE OF CONTENTS

About the Small Computers in the Arts Network (SCAN)

The rapid development of the personal computer over the last decade and a half has opened doors in all walks of life that were only dreams in the past. For artists of all types, this means exciting new tools have emerged, bringing about interactivity between many areas in the arts; practitioners of music, graphics, sculpture, performance and arts education, college professors and engineers are finding themselves increasingly able to bridge technical, physical, and creative gaps as the use of small computers grows.

SCAN was founded eleven years ago, evolving out of the Philadelphia Computer Arts Group, which had held Philadelphia area electronic music concerts since 1978. SCAN is now a non-profit corporation. We publish a newsletter quarterly, in which we will publish artwork and short articles, as well as information regarding coming events (concerts, openings, publications, seminars, etc.) in the electronic arts field. SCAN provides an environment of personal and creative enrichment for all computer-assisted art enthusiasts, both seasoned and novice alike.

SCAN's biggest event is our annual symposium held at the University of the Arts in Philadelphia, a city rich in artistic tradition and influence. The conference serves as the medium for the exchange of new ideas, art forms, and technological innovations, propelling the arts down new trails. Through presentations, exhibits, performances, and shows, SCAN '91 will explore the new realities in art, music, art education, ethics, electronic music, multimedia, arts production, performance and the impact of software and hardware development in the production of art.

Dick Moberg, 11/91

Editor's Introduction

Well, this year's SCAN group has been doing large amounts of leg, hand and brainwork assembling the various Symposium activities. One strong element is this Proceedings, a very exciting collection of works.

Maybe for the first time in this year's Proceedings papers we are seeing a certain level of confidence beyond the microprocessor-based solutions being used or pursued. Maybe it is a refocusing from basic functionality problems to high level conceptual ones. Also, I think it would be safe to say that the PC industry as a whole has achieved a great deal of competence in the areas that it has been striving for, and only in the past year or so has this become clear to the general public.

For example, once we needed to strive to achieve a computer screen or printed image at "fool the eye"-believeable resolutions. With inexpensive 8-bit to 24-bit color systems now standard, this is a non issue, and the push is now toward believable sequences of or believable interactions with these images. It seems like the skills of a computer artist must expand into all the arts in order to access all that these black boxes have to offer us.

Computer-Aided Design, Desktop Publishing, Computer Animation technologies, Real-time Digital Signal Processors (DSP), MIDI, Digital Photography (yea Adobe PhotoShop!), 3D Modeling and Rendering, and especially Color Publishing (including scanning, proofing and separation technologies) have all come into more prominent profile as the past year has progressed.

Struggling technologies like Multimedia and Hypertext are only several small products away from mainstream applications as well. Many of the papers herein deal with the trials and tribulations of the artist/programmer searching for and/or building the "right" tools on microprocessor-based systems.

On the horizon are Neural Net (non-von Neumann-style) programming environments and interactive "Virtual Reality"systems. These are both clearly important areas with which to gain familiarity, while at the same time being several years away from comfortable desktop applications.

The activity in these and other fields demonstrate that the small/home/personal/artist computer revolution is not over yet, and that we are certainly on the verge of even larger breakthroughs in our efforts to achieve new media for communication, education and self-expression.

Mark W. Scott, 11/91

Neural Networks for Applications in the Arts

Peter M. Todd
Department of Psychology
Jordan Hall, Building 420
Stanford University
Stanford, CA 94305
todd@psych.stanford.edu

1 Introduction

Traditionally, computer applications in the arts have been pretty much like computer applications in any other field: you'd like the computer to create a certain sort of output, so you construct an algorithm for it to follow step by step which will result in that output. This is a fine approach when you know what you're going for and where the production process should be headed at each step, that is, when you can precisely specify the rules involved in generating the output. Programming a computer to invert a matrix or control an assembly-line drill-punch is certainly like this, and the production of a variety of types of art are amenable to this approach as well. But there are also a wide range of instances in which we as artists may not want to, or be able to, specify a set of rules by which our next computer artwork is to be created. Sometimes it may just be too difficult or time-intensive to list the intricacies of a particular creative process in a fashion definite enough for a computer to follow; and other times, we may have no idea how to even go about constructing an appropriate set of rules (what are the "rules" to follow in creating a Rodin sculpture?). At times like these, a new approach is needed.

A new approach to computer art applications is provided by the rapidly expanding field of neural networks. Rather than operate in the traditional style of preprogrammed rule-following systems, neural networks have the power to *learn* to produce specific types of outputs from specific inputs, based on examples they are taught. Thus, instead of having to specify *how* to create a certain artwork, the artist can instead teach a network *examples* of the desired output, and have the network generate new instances in that style. We need not specify the steps involved in creating a Rodin sculpture--we can just collect instances of the sorts of sculpture we'd like to get and use those to train a network. This sort of thing is the promise of neural network applications in the arts, stated in a very provocative and exaggerated manner. But we are beginning to realize this promise in limited domains, and the space for further applications and explorations of these techniques is wide open.

In this paper, I will briefly describe neural networks, some of what they can and can't do, some of what they have done already in the arts, primarily music, and some of what they can be applied to in other areas. I will also give a few pointers to the literature for further reading and exploration of these topics; I hope people will be encouraged to pursue all of these avenues and see where their creativity leads them.

2 Neural Networks

The standard von Neumann-style computer programming paradigm is based on the concept of a single powerful central processing unit that performs a sequence of operations on the contents of an external memory. The neural network computing paradigm, in contrast, is based largely on the notion of "brain-style computation" (Rumelhart, 1990), in which a large number of very simple processing units act simultaneously on a distributed pattern of data--hence the common name, parallel distributed processing (Rumelhart and McClelland, 1986). The analogy here is to the type of computation done by the vast numbers of simple neurons in the brains of living creatures.

There is a vast assortment of mathematical formulations of different neural network types, but for our purposes we can think about a simple case in which a network consists just of a set of simple processing units, connected by a set of weighted links. The currency of the network is numerical values--the units all take in numerical values passed to them on the links from other units, and they in turn produce a single numeric output value with they pass on their output links to still other units for further processing. A subset of the units will typically have a special status as input units, which take some pattern of "information" (represented as a set of numerical values) from the external world into the network for processing. Similarly, there is a set of output units, whose final processed values are taken as the end product of the network's computation. Along the path of links inbetween these two sets can be a collection of "hidden" units, which perform the network's main work. They transform the input values into successively more "useful" representations on the way to the final output.

All of the units in a network (of the sort we're describing here) do exactly the same thing, summing up their inputs from other units, and putting that value through a "squashing function" which maps the unit's output into a restricted range in a non-linear fashion. This non-linearity allows the network to make distinct categorizations and judgments about its input. Since all of the units act the same, the only way to get two networks with the same unit setup to act differently is to have different weights on their links between the units. And the easiest way to set the weights on those links is to use a learning method.

Learning is a key feature of neural networks, and a big advantage they have over other computational frameworks. Learning comes naturally in networks, but is typically ad-hoc, if possible at all, in rule-based systems. The weights in a network are typically learned by a type of method known as *gradient descent*, in which small adjustments are made to appropriate (mathematically determined) weights to lessen the network's performance error. This error is a measure of how far the network's actual current performance deviates from its desired performance--that is, how far the outputs the network is now creating differ from the target outputs in the examples the network is trying to learn. For instance, if the example we're training the network on is to sculpt a bust, and its output corresponds to a sculpted automobile, there would be a large error difference there, and a lot of weights would need to be changed in the network. If its output corresponds instead to a head without eyes, the error, and the necessary weight-changed, would be correspondingly smaller. Tiny adjustments are made to the weights during each step in training the network, and then the example is tried again, to see how far the target from the current output is now; and then again the weights are changed slightly. After many cycles of this test-and-adjust, the network's performance will hopefully have come to match the desired examples closely enough, and training can stop. After this, the network can be used to create new outputs.

(For more details on the actual implementation and mathematics of all this, presented from the standpoint of musical applications, see Dolson, 1991; Rumelhart, Hinton, and Williams, 1986, provides the original complete derivation of a particular popular learning method known as "back propagation of error"; McClelland and Rumelhart, 1988, provide a tutorial introduction to a variety of neural network approaches and the theory underlying them, and even provide software on diskettes for running your own simulations on Macs and PC's; and Trubitt and Todd, 1991, present these issues in a little more detail but at a very informal level, again for application to music.)

Because of the "massive parallelism" achieved by many units operating simultaneously on a distributed pattern of information, small changes in the network will affect its performance rather little. Removing one unit might not do too much to the network's performance error, because several other units also take part in the calculations it performed and can "fill in" for it, so to speak. In contrast, removing a single line from a standard algorithm can render the entire thing useless--catastrophic failure. A further consequence of this parallel distributed processing is that small changes in the inputs will result in small changes in the output. This means the network will generalize to new inputs in reasonable ways--if it sees something new but not too different from a previously-seen input pattern, it will output something similar to the associated old output pattern. So if for instance an input of values "1,0,0,1,0,1,1" resulted in an output of a sculpted ram's head from a specifically trained network, an input of values "1,0,0,1,0,1,0" instead might result in a similar output, but with slight shorter horns. This generalization to new inputs in sensible ways is the basis of many of the creative applications of neural networks.

So far most of the examples I've used for network art applications have centered on the production of new output, with little regard for the input end of things. But one of the tasks neural networks are best suited for is more standard input-output processing, taking a certain input set of values and transforming it in particular ways to generate the (more-or-less related) output. More specifically, the inputs and outputs can represent signals of some sort, whether auditory, visual, or otherwise, and the network can perform some kind of signal processing function, removing noise, rotating an image, filling in colors, etc. Networks can also be used in interesting dynamic ways, for instance by connecting their outputs back to their inputs to create a feedback loop that generates a sequence of outputs. As nonlinear dynamic systems of this type, sequential networks can have attractors and cycles that are useful in a variety of contexts (see Todd, 1991, for a musical application, and Jordan, 1986, for guiding arm movement). And as adaptive systems that learn to respond in appropriate ways in changing situations and environments, neural networks have found extensive use in the control of motion and navigation in robot and other systems. These are the sorts of tasks networks are particularly suited for, and as we will see, they can be particularly useful in a lot of artistic applications.

There are, though, still several things which the neural network approach is not good for at present. Networks are very good at learning the low-level details needed to perform a certain input-output mapping, for instance in learning to produce the next measure of music as output, given the previous measure as input. They are notoriously poor, however, at picking up the higher-level structure that organizes the lower-level features, again for instance missing the global movement away from and back to the tonic in the succession of measures in a piece of music. In fact, the higher-level the task in general, the fewer successful applications there have been of neural networks, where high-level here means things such as language processing as opposed to speech, image understanding as opposed to processing, motion planning as opposed to production. And focussing as they do on *sub*-symbolic processing, distributing meaning across a number of units and connections, neural networks have not made much inroad as yet into the traditional symbol-processing arenas of symbolic reasoning, syntax and semantics, expert systems, and rule-following. But then again, as I described in the introduciton, the beauty of the parallel distributed processing paradigm is that its strengths lie elsewhere.

3 Applications in the Arts

We now turn to the artistic uses of neural networks. Currently, this is a wide-open field; exploration has just begun in most cases, and we've barely scratched the surface of possibilities. The ideas below are mostly speculations on what networks *could* do, the sorts of tasks they *could* be applied to in the arts, sometimes based on applications that have already been done in scientific or engineering domains, and sometimes just based on imaginative speculation. As such, these ideas are intended to spark people's imaginations further in the search for innovative uses of this poweful and flexible new technology.

The main place where neural networks have been put to creative and artistic use so far is in music, as witnessed by the recent publication of the book, *Music and Connectionism* (Todd and Loy, 1991). Several applications have been done in this area, ranging from psychological models of human pitch, chord, and melody perception, to networks for algorithmic composition and performance control. Generally speaking, the applications here (and in other fields) can be divided into two classes: "input" and "output". The input side includes networks for recognition and understanding of a provided stimulus, for instance speech recognition, or modelling how humans listen to and process a melody. Such applications are useful for communication from human to machine, and for artistic analysis (e.g. musicologically, historically) of a set of inputs. The output side includes the production of novel works, applications such as music composition or drawing generation. "Input" tasks tend to be much more difficult than "output" tasks (compare the state-of-the-art in speech recognition versus speech production by computers), so most of the network applications so far have focussed on creation and generation of output, but continuing research has begun to address this imbalance.

On the "input" side in musical applications, Sano and Jenkins (1991) have modelled human pitch perception; Bharucha (1991) (and others) have modelled the perception and processing of harmony and chords; Gjerdingen (1991) has explored networks that understand more complex musical patterns; and Desain and Honing (1991) have devised a network for looking at the quantization of musical time and rhythm. Dolson (1991) has also suggested some approaches to musical signal processing by neural networks, including instrument recognition, generation, and modification. In this regard, musical applications of networks have much to gain from the vast literature on networks for speech processing (primarily recognition--see Lippman, 1989).

On the "output" side, several network models of music composition have been devised. Todd (1991a) and Mozer (1991) use essentially the dynamic sequential network approach mentioned earlier, in which a network is trained to map from one time-chunk of a piece of music to the following time-chunk (e.g. measure N as input should produce measure N+1 as output). The network's outputs are then connected back to its inputs for the creation phase, and a *new* measure 1 is provided to begin the network down a new dynamic path, creating one measure after another, and all the while incorporating the sorts of features it learned from its training examples. In this way, new pieces that have a sound like Bach or Joplin (or a combination of both!) can be created, if the network is first trained on these composers. But the problems mentioned earlier of lack of higher-level structure emerge, and these compositions tend to wander, having no clear direction, and seldom ending up anywhere in particular. Approaches for learning and using hierarchical structure are being devised, and Lewis (1991a) describes one such method, in which the *inputs* to a network, rather than the weights in the network, are modified during a learning stage, to produce an input which has a specified form or character. Kohonen et al. (1991) present still another method of composition, which uses a network-style approach to build up a context-free grammar that models the music examples it's trained on.

Networks can also be used to generate musical performance parameters and instructions, as Sayegh (1991) demonstrates in his paper on a network method for choosing correct chord fingering for a simple melody. Many other musical performance applications are possible, from synthesizer control to automatic rhythmic accompaniment generators; Todd (1991b) discusses some of these possibilities along with further ideas for musical applications of neural networks.

Neural networks have been applied to a variety of image-processing tasks, from video image compression to aspects of computer vision such as image segmentation and object recognition. As signal processors, networks should find wide application in image enhancement, color adjusting or altering, edge and line modification, texture processing, etc., all based on learned mappings from input pictures to desired outputs. As another example, Chen et al. (1990) have developed a network for choosing a palette of colors (for use on a Macintosh screen) from a small input set of preferences.

At a higher level, networks can be used to generate actual drawings. The tricky part here is figuring out how to encode the drawings into a numerical form that the network can work with; Lewis (1991b) uses spline coefficients to encode simple drawings that are learned by a network similar to the one he has used for music (1991a). While the his results are so far admittedly rather crude, they point in a direction of great promise. Writing, calligraphy, and the creation of new fonts is another area networks can profitably tackle. Grebert et al. (1991) have developed a system which will generate a complete font in a certain style, given just a few letters in that font to begin with. Again often the network's output is questionable, but it does an interesting job of generalization, to say the least, and enhancements of this technique could yield a bonanza of new letter and writing styles. (The area of neural network handwriting recognition is also being widely explored, but again there as for speech, the problems are harder on the input side.)

Besides processing images and drawings in more-or-less straightforward ways, networks could be used for much more sophisticated manipulations. For example, a network could be trained to take an image of a face as input, and produce as output an image of that face now displaying any of several chosen emotions: smiling, frowning, staring, tongue stuck out, etc. Cottrell and Metcalfe (1991) and Fleming and Cottrell (1990) have investigated networks for face, gender, and emotion recognition, and have found that to achieve these tasks, the networks develop internal representations of the input faces which could be manipulated at the output level to alter those faces in systematic ways. This opens up the possibilities for a wide range of very powerful image manipulation techniques, altering scenes in meaningful ways, putting people and animals in different poses, changing the bloom on plants, etc.

From there, it is only a small step to the manipulation of *moving* images, especially represented as successions of static frames. Neural networks could straightforwardly be trained to perform sophisticated tweening between frames in animation, for instance. Sequential networks could be used to produce the step-by-step movements of objects following some trajectory in space, while the types of networks just described could adjust the expressions and gait of animated characters (see de Garis, 1991, for a network that controls the walking of an animated creature).

But even more exciting would be to let neural networks control the actions and behaviors of actors in a visually-displayed world entirely on their own, without following a preordained animation script. By imbuing artificial creatures in a virtual world with the ability to move, sense, eat, fight, hunt, court, etc. independently, we could create "living paintings" in which we watched a herd of sheep grazing and moving about on a landscape, or the interactions of two birds in flight, or the slow and steady movement of an ant colony across a jungle floor. Researchers in the field of artificial life (Langton et al., 1992) are beginning to realize some of these goals. The "brain-style computation" of neural networks is a very natural choice for simulating a creature's behavioral and cognitive mechanisms (Miller and Todd, 1990). Ackley and Littman (1992) have developed a system in which neural network-controlled creatures roam around a two-dimensional world, looking for food, avoiding obstacles, dodging predators who also hunt them down, and otherwise acting the way real organisms do; when these interactions are displayed on a computer screen in real-time, they are very compelling and engaging. Networks could also be used to control independent and realistically-behaving simulated "pets" and other creatures that a human could actually interact with in a virtual reality system. Here again, obviously, the creative angles are endless.

Networks can also find interesting use in real-world kinetic applications, for controlling the motion of robots and vehicles from single arms to self-driving automobiles (Miller, Sutton, and Werbos, 1990). Pomerlau (1991) uses networks to guide the navigation of an autonomous land vehicle (ALVINN) by visual tracking of a road surface. Mel (1990) has developed a system (MURPHY) that controls robot arm motion based on visual input of the arm and goal's location. Pole-balancing, in which a movable platform (a cart) is shifted around to keep a pole balanced on it remaining upright, is another common control problem to which networks have been successfully applied. Artforms including kinetic sculpture, machine dance, automated music conducting, etc., are fertile areas for neural network applications.

Finally, as mentioned earlier, neural networks can be used in human-computer interaction, to help create interactive artworks by following the performance and instructions of human artists, dancers, conductors, etc. in real time. Preliminary work has been proposed for network processing of gestural inputs from Powerglove-like input devices, and from Max Mathews' radio drums; virtually any dynamic input of this sort (even EEG signals in one application!) is amenable to processing by a neural network which produces appropriate interpreted commands as its output. With speech and handwriting recognition capabilities as well, networks may be able to provide unprecedented levels of communication back and forth between artist and computer.

4 Conclusions

The current status of neural networks in the arts is one of great promise, but little realization. At best, networks can presently be used as intelligent composer's aids in the creation of new pieces of music, based on learned examples. This is a valuable service, and one at which the network, as an artist's apprentice, can get better and better as it learns more from the artist. Such a support role is probably how neural networks will first appear in applications in other artistic domains as well, generating small ideas and variations which the human artist can then incorporate or disregard in the most aesthetic fashion.

There is some debate over whether the neural network approach to art, and in particular music, is an appropriate one (see Todd and Loy, 1991, section IV). Some argue that the reason computers have been so valuable in art is precisely that they've allowed us to imagine processes by which we'd like to create artworks, specify a very precise set of rules to carry out that process, and then implement those rules to fulfill our imagination. Neural networks, they argue, have eliminated that ability, and chained us back down to only those artforms and styles that we can collect examples of. But this argument, of course, is invalid--if we can create an algorithm to generate new artistic styles in the traditional manner, then we can use those computer-generated works as further examples on which to train our networks. Thus networks need not constrain us at all, but rather add yet another tool into the artist's toolbox. And it is ultimately the products of those tools that we will judge artistically, not the tools themselves.

(For more information on a variety of types of neural networks and their applications, see the yearly proceedings of the Neural Information Processing Systems (NIPS) and International Joint Conference on Neural Networks (IJCNN) meetings, from which several of the papers in the references are taken.)

References

Ackley, D., and M. Littman (1992). Interactions between learning and evolution. In Langton, et al.

Bharucha, J.J. (1991). Pitch, harmony, and neural nets: A psychological perspective. In Todd and Loy.

Chen, J.R., R.K. Belew, and G.B. Salomon (1990). A connectionist network for color selection. In M. Caudill (Ed.), *Proceedings of the 1990 International Joint Conference on Neural Networks*, vol. II. Hillsdale, NJ: Lawrence Erlbaum Associates, pp. 467-470.

Cottrell, G.W., and J. Metcalfe (1991). EMPATH: Face, emotion, and gender recognition using holons. In R.P. Lippmann, J.E. Moody, and D.S. Touretzky (Eds.), *Advances in neural information processing systems 3.* San Mateo, CA: Morgan Kaufmann Publishers, pp. 564-571.

de Garis, H. (1991). Genetic programming. In B. Soucek (Ed.), *Neural and intelligent systems integration.* Wiley.

Desain, P., and H. Honing (1991). The quantization of musical time: A connectionist approach. In Todd and Loy.

Dolson, M. (1991). Machine tongues XII: Neural networks. In Todd and Loy.

Fleming, M., and G. Cottrell (1990). A neural network model of face recognition. In M. Caudill (Ed.), *Proceedings of the 1990 International Joint Conference on Neural Networks,* vol. II. Hillsdale, NJ: Lawrence Erlbaum Associates, pp.

Gjerdingen, R.O. (1991). Using connectionist models to explore complex musical patterns. In Todd and Loy.

Grebert, I., D.G. Stork, R. Keesing, and S. Mims (1991). Connectionist generalization for production: An example from GridFont. In *Proceedings of the 1991 International Joint Conference on Neural Networks,* vol. II. Piscataway, NJ: IEEE Inc., p. 105-109.

Jordan, M.I. (1986). Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Erlbaum Associates.

Kohonen, T., P. Laine, K. Tiits, and K. Torkkola (1991). A nonheuristic automatic composing method. In Todd and Loy.

Langton, C.G., C. Taylor, J.D. Farmer, and S. Rasmussen (Eds.) (1992). *Artificial Life II.* Redwood City, CA: Addison-Wesley.

Lewis, J.P. (1991a). Creation by refinement and the problem of algorithmic music composition. In Todd and Loy.

Lewis, J. (1991b). Probing the critic: Approaches to connectionist pattern synthesis. In *Proceedings of the 1991 International Joint Conference on Neural Networks,* vol. I. Piscataway, NJ: IEEE Inc., p. 85-90.

McClelland, J.L., and D.E. Rumelhart (1988). *Explorations in parallel distributed processing.* Cambridge, MA: MIT Press/Bradford Books.

Mel, B.W. (1990). Vison-based robot motion planning. In Miller, Sutton, and Werbos.

Miller III, W.T., R.S. Sutton, and P.J. Werbos (Eds.) (1990). *Neural networks for control.* Cambridge, MA: MIT Press/Bradford Books.

Miller, G. F., & Todd, P. M. (1990). Exploring Adaptive Agency I: Theory and Methods for Simulating the Evolution of Learning. In D.S. Touretzky, J.L. Elman, T.J. Sejnowski, & G.E. Hinton (Eds.), *Proceedings of the 1990 Connectionist Models Summer School.* San Mateo, CA: Morgan Kaufmann, pp. 65-80.

Mozer, M. (1991). Connectionist music composition based on melodic, stylistic, and psychophysical constraints. In Todd and Loy.

Pomerlau, D.A. (1991). Rapidly adapting artificial neural networks for autonomous navigation. In R.P. Lippmann, J.E. Moody, and D.S. Touretzky (Eds.), *Advances in neural information processing systems 3*. San Mateo, CA: Morgan Kaufmann Publishers, pp. 429-435.

Rumelhart, D.E. (1990). Brain style computation: Learning and generalization. In S.F. Zornetzer, J.L. Davis, & C. Lau (Eds.), *An introduction to neural and electronic networks.* San Diego: Academic Press.

Rumelhart, D.E., Hinton, G.E., and Williams, R.J. (1986). Learning internal representations by error propagation. In Rumelhart and McClelland.

Rumelhart, D.E., and J.L. McClelland (Eds.) (1986). *Parallel distributed processing: Explorations in the microstructure of cognition. Vol. 1: Foundations*. Cambridge, MA: MIT Press/Bradford Books.

Sano, H., and B.K. Jenkins (1991). A neural net model for pitch perception. In Todd and Loy.

Sayegh, S.I. (1991). Fingering for string instruments with the optimum path paradigm. In Todd and Loy.

Todd, P.M. (1991a). A connectionist approach to algorithmic composition. In Todd and Loy.

Todd, P.M. (1991b). Further research and directions. In Todd and Loy.

Todd, P. M., and Loy, D. G. (Eds.) (1991). *Music and Connectionism.* Cambridge, MA: MIT Press.

Trubitt, D. R., & Todd, P. M. (1991). The Computer Musician: Neural Networks and Computer Music. *Electronic Musician, 7(1)*, 20-24.

Recent Work In Music Understanding

Roger B. Dannenberg

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213 USA
Email: dannenberg@cs.cmu.edu

Abstract

Interaction with computers in musical performances is very much limited by a lack of music understanding by computers. If computers do not understand musical structures such as rhythmic units, chords, keys, and phrases, then interaction with computers will necessarily be difficult and cumbersome. Research into Music Understanding by computer aims to raise the level of human computer interaction in musical tasks including live music performance.

1. Introduction

Music Understanding is the recognition or identification of structure and pattern in musical information. Music Understanding is important because it opens the doors to high-level interaction between musicians and computers. Research at Carnegie Mellon University has led to a number of interesting developments which are summarized here.

The following sections briefly describe five projects related to Music Understanding. The first two describe computer accompaniment systems, which listen to live musical performances and synchronize pre-stored computer music accompaniments. The third project is a system for analyzing the harmonic and rhythmic content of an improvised solo in order to follow a jazz improvisation. This work led to further investigations of the "foot-tapping" or beat detection problem. Music understanding and intelligence have been applied in the Piano Tutor, which is the fifth project described.

One of the results of this work is a better appreciation of the difficulty of these tasks. Some directions for future research and conclusions are given following the descriptions of Music Understanding projects.

2. Computer Accompaniment

When human musicians perform together, they typically listen to one another and synchronize their music according to a musical score. In contrast, most performances with computers require that humans follow a computer-based "sequencer" that has no listening abilities whatsoever. An alternative approach is to build a computer system that can listen to the human musician and synchronize its performance. I call this task *computer accompaniment*.

Computer accompaniment has been described in the literature [3, 2, 5, 9, 7], so this section will only sketch our approach to the problem. As shown in Figure 1, a computer accompaniment system consists of several stages. In the first stage, the Listening Task detects note onsets and pitches in the Solo Performance. This data is sent to the Matcher, which compares the live human performance with a stored score. It is assumed that the intended performance is completely notated. This assumption (and the score) gives the computer a great deal of information. (Sections 4 and 5 discuss systems where this assumption is not made.)

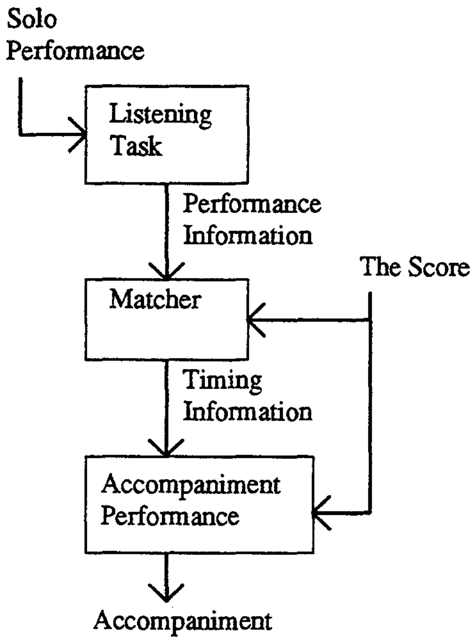


Figure 1: The correspondence between a score and a performance.

The result of the Matcher is an indication of where and when the performance corresponds to the score. This information can be used to estimate the performer's tempo and current location in the score. This in turn can be used to perform an accompaniment, which is also stored in a score (no attempt is made to compose or improvise accompaniment in these systems). The output consists of real-time control information for a synthesizer.

One of the problems of this Accompaniment Performance stage (see Figure 1) is performing the accompaniment in a musical manner even when the performer is missing notes and changing tempo. The Accompaniment Performance stage uses a number of rules about musical performance that help it to respond appropriately and musically when new information is received from the matcher. As a result, the computer accompaniment system performs in a fairly autonomous manner; it is guided by the human performance, but quite capable of performing on its own as well.

The most advanced accompaniment system to date can also handle a small degree of improvisation in the form of trills, grace notes, and glissandi, which do not match up note-for-note with the score [5]. This system has a number of features for controlling the accompaniment, for example limiting the range of tempo adjustment or ignoring input during a steady-rhythm passage. This system has been used by professional musicians in performances.

3. Polyphonic Accompaniment

The first accompaniment systems only worked with monophonic input, that is, input without chords or overlapping notes. This is of course a major drawback for keyboard performers, so a new system was developed for polyphonic accompaniment. Referring again to Figure 1, it can be seen that the challenge in making a polyphonic system is developing a matcher that can match polyphonic performances to polyphonic scores. Two matchers were developed for this purpose [2]. The resulting matchers work quite well and allow accurate following even in the presence of many performance errors.

4. Following Improvisations

Computer accompaniment is based upon traditional (Western) music making in which a composition governs what notes are played by the performers. In jazz and other improvisational styles, this information is not available. However, improvisations are not without structure. In particular, jazz improvisations usually have a tempo, measures, and chord sequences among other features.

Working with Bernard Mont-Reynaud, I developed a computer system that listens to a blues improvisation played on trumpet. The goal of the program is to accompany the trumpet with a rhythm section of synthesized bass, drums, and piano. This requires that the program understand a fair amount of structure in the solo part. Notice that in this system, the rhythm section will listen to and follow the soloist rather than the other way around.

"The Blues" is a sequence of chords that repeats every 12 measures, where a measure is a musical unit of 4 downbeats. The solo harmonizes with the chord sequence, so the solo part gives some indication of the underlying chord progression. Certain pitches are more likely to occur in combination with one chord than another. However, any note in isolation could occur almost anywhere in the 12 measures, so the problem is to infer a precise location from a large number of ambiguous indicators.

Our tempo-tracking software is based on the idea that note onsets typically occur on either an upbeat or a downbeat. Once a tempo is established, the system predicts where note onsets will occur and compares these predictions to the actual performance data [8]. When performed notes correspond closely to predictions, the current notion of tempo and where beats occur in time is adjusted slightly to make the predictions even better. In this way, a slowly wandering performance tempo will tend to "pull" the estimated tempo, and tracking occurs.

Once the tempo and beat locations are determined, a statistical approach is used to find how chord changes relate to the solo. A table of probabilities is used. The table lists the likelihood that a given pitch will occur on a given downbeat or upbeat (these will be called simply "beats"). Each column of the table corresponds to one of 96 downbeats and upbeats, and each row corresponds to one of 12 possible pitches (octaves are ignored), so the table has 1152 entries. Suppose the first note of the solo is an F. Then the F row of the table gives an estimate of the likelihood that the solo started at each of 96 possible beats in the 12-measure blues. Now suppose the next beat of the solo is a G. The G row of the table gives the probability that the G was played at each possible location. Now, taking the F probability of the first column times the G probability of the second column gives a combined likelihood estimate of the combination F-G being played at the beginning of the 12 measures. Similarly, taking the F-G probabilities from some other pair of columns gives a likelihood estimate that the F-G combination was played elsewhere.

Extending this process, we can get likelihood estimates for the entire solo starting on any beat. Figure 2 shows a graph of this likelihood estimate. (See [4] for more details.) It is interesting that the curve shows a 4-measure periodicity that reflects the fact that 12-measure blues has 3 somewhat similar 4-measure phrases. The peak at zero and at 96 is the same peak (the graph repeats itself every 12 measures or 96 beats). This peak correctly indicates the most likely starting point for the 12-measure blues chord progression.

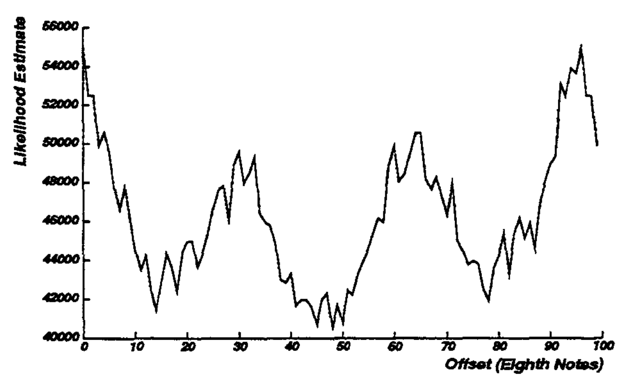


Figure 2: Likelihood estimates (arbitrary units) of the solo starting at different offsets in a 12-bar blues progression.

A real-time implementation of this system demonstrates some ability to recognize and follow a blues improvisation. However, the program requires a full 12 measures of performance data before a location estimate can be made, and the system is not nearly as good as humans at recognition. This experience led to further research on the beat tracking, or "foot tapping" problem, which is described in the next section.

5. Rhythm Understanding

A fundamental musical skill is the ability to recognize a pulse or beat in a music performance. One approach to this "foot tapping" problem was describe in the previous section. In this approach, an estimate of beat location and tempo is adjusted to fit incoming data points. Unfortunately, this approach fails quite often, especially if there are sudden tempo changes. Once a failure occurs, it is hard to recover because adjustments tend to be made due to random coincidences between anticipated beats and actual note onsets. The more responsive the tracker is, the more likely it is to become confused.

Paul Allen and I developed a new approach to solve problems with earlier trackers [1]. Our basic new idea is that trackers are thrown off by input that is misleading and should be ignored. Suppose at each decision point, the tracker made two choices instead of one. For several beats, the consequences of each decision could be carried out, and then the result that seems best could be retained and the other one discarded. We implemented a system that considers two or three rhythmic interpretations of each incoming note onset. At any time, the beat tracker is considering tens or hundreds of alternative ways of "tapping its foot" to the music.

Some interpretations make more musical sense than others. For example, an interpretation where the beat is steady is generally preferred as is an interpretation that requires only simple rhythms. Based on a musical evaluation, interpretations are constantly pruned from the search to make room for better ones. Figure 3 illustrates how interpretations, represented by circles, give rise to multiple interpretations of new events (each new event corresponds to a new row) or are pruned from the search (represented by crosses).

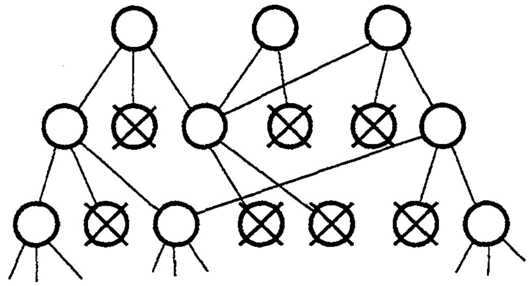


Figure 3: Three levels of search.

A real-time implementation of this algorithm shows that the technique is successful in recovering from bad decisions that would have thrown off earlier trackers. However, the system still has difficulty in separating good interpretations from bad ones, and by considering so many interpretations, it is hard not to prune the correct one occasionally. The performance is very dependent upon the musical input, making evaluation difficult, and more evaluation is needed to better characterize the limitations of this approach.

6. The Piano Tutor

The Piano Tutor is an intelligent computer system for teaching beginners to play the piano [6]. The Piano Tutor makes extensive use of music understanding to support an instructional dialog with the student. In a typical interaction, the Piano Tutor delivers a multimedia presentation to the student and asks the student to perform an exercise. The student performs, but usually makes a mistake. The Piano Tutor corrects the student and asks for another attempt. This continues until the student masters the exercise. Then the Piano Tutor selects new material for the student and the interaction cycle repeats.

Music understanding takes place on three levels. First, the Piano Tutor uses computer accompaniment technology to follow student performances. The Piano Tutor can play musical accompaniments to student performances and also turn pages of music on a computer graphics display. Second, student performances are analyzed by the Piano Tutor to determine if the student is having problems, and if so, the likely cause of the problem. For example, a duration error might be accounted for by a misunderstanding of ties, a problem keeping a steady tempo, forgetting to release a note at the beginning of a rest, or any number of other possibilities. The Piano Tutor looks for the most significant error (in a pedagogical sense), finds an explanation for the error, and then computes an appropriate remediation for the student. The third level of understanding has to do with modeling the student as a developing musician. The Piano Tutor keeps track of what skills the student has mastered and uses its model of student progress to select appropriate lessons for the student. As the student masters new material, the model is updated. As a consequence, lessons are tailored to the individual needs of the student.

7. Summary and Conclusions

Music understanding requires the recognition of pattern and structure in music. I have presented an overview of five projects in music understanding, all conducted at Carnegie Mellon University. Some of the projects, such as computer accompaniment and the Piano Tutor have produced surprisingly good results. Others, such as following improvisations and "foot tapping" show how difficult more general problems of music understanding can be.

Many other tasks are still open problems to be tackled. These include following the performance of an ensemble, with or without a score, integrating pitch information in the rhythm understanding task, following vocal music, and the use of learning to improve the performance of various music understanding systems.

There are many reasons for continuing research in music understanding. Computer music system interfaces can be improved if they can deal with musical structures and commands at a high level. Music understanding can shed light on human cognition, and there is much related research in the area of Music Psychology. Finally, music understanding can have a great impact on music theory and music formalisms. Music understanding is still a new field, and one can expect many breakthroughs in the future.

8. Acknowledgments

This paper is based on a talk prepared for the International Wenner-Gren Symposium on Music, Language, Speech, and Brain. Many colleagues have contributed to the projects described here, including Joshua Bloch, Bernard Mont-Reynaud, and Paul Allen. Marta Sanchez and Annabelle Joseph conceived of the Piano Tutor and are co-principal investigators on the project, which also benefited from the work of Peter Capell, Ron Saul, Robert Joseph, John Maloney, and Hal Mukaino.

This work has been made possible largely through the Carnegie Mellon University School of Computer Science and was partially supported by Yamaha (computer accompaniment) and the Markle Foundation (the Piano Tutor).

References

[1] Allen, P. E., and R. B. Dannenberg. Tracking Musical Beats in Real Time. In *Proceedings of the 1990 International Computer Music Conference*. Computer Music Association, 1990.

[2] Bloch, J. J. and R. B. Dannenberg. Real-Time Computer Accompaniment of Keyboard Performances. In *Proceedings of the 1985 International Computer Music Conference*, pages 279-290. Computer Music Association, 1985.

[3] Dannenberg, R. B. An On-Line Algorithm for Real-Time Accompaniment. In *Proceedings of the 1984 International Computer Music Conference*, pages 193-198. Computer Music Association, 1984.

[4] Dannenberg, R. B. and B. Mont-Reynaud. Following an Improvisation in Real Time. In *Proceedings of the 1987 International Computer Music Conference*, pages 241-248. Computer Music Association, San Francisco, 1987.

[5] Dannenberg, R. B. and H. Mukaino. New Techniques for Enhanced Quality of Computer Accompaniment. In *Proceedings of the 1988 International Computer Music Conference*, pages 243-249. Computer Music Association, San Francisco, 1988.

[6] Dannenberg, R. B., M. Sanchez, A. Joseph, P. Capell, R. Joseph, R. Saul. A Computer-Based Multi-Media Tutor for Beginning Piano Students. *Interface - Journal of New Music Research* 19(2-3):155-173, 1990.

[7] Lifton, J. Some Technical and Aesthetic Considerations in Software for Live Interactive Performance. In *Proceedings of the 1985 International Computer Music Conference*, pages 303-306. Computer Music Association, 1985.

[8] Longuet-Higgins, H. C. and C. S. Lee. The Perception of Musical Rhythms. *Perception* 11:115-128, 1982.

[9] Vercoe, B. and M. Puckette. Synthetic Rehearsal: Training the Synthetic Performer. In *Proceedings of the 1985 International Computer Music Conference*, pages 275-278. Computer Music Association, 1985.

The Abstract Visualization of Music: Then and Now

Bruce Wands
MFA Computer Art
School of Visual Art
141 W. 21 St.
New York, NY 10011

Abstract
This paper deals with the abstract visualization of music from two viewpoints,the traditional and the contemporary. I will present each method and, in doing so, endeavor to shed some light on the process of creating abstract animation. Although traditional techniques might seem "old fashioned" to some, they are as valid a path today as they have aways been. In the 1920's, the animation of music was done largely by two methods, stop motion and cel animation. Soundtracks were recorded on magnetic film stock and transferred to exposure sheets or bar sheets and then animated by manual methods. Oscar Fischinger was one of the pioneers in the abstract animation of music. His production techniques and work are discussed. In the 1990's, computer graphics systems and MIDI have greatly modified the means by which one can abstractly animate music. By discussing these methods in the context of his own work, and that of others, the author hopes to demonstrate a practical approach to this subject that can be taken by both artists and musicians. The relationship of the development of visual ideas to musical structure is presented. The technical details of how to read a soundtrack are also given.

Introduction
By nature, music is invisible. It is heard, but not seen. Much of the appreciation of music stems from the individual interpretation each person makes while experiencing it. Music can evoke emotions, thoughts and memories that are unique to each person. So, why visualize music ? To answer this question, one must look beyond the music and to the experiencing of it. Music can be listened to in private or in a concert setting. In private, the listener often lets his mind be taken by the music. In a concert setting, the performance often dictates the visual impact of the music. When viewing film or videotape, the environment focuses the listener's attention on the screen. It is this aspect upon which this paper focuses. The abstract animation of music is a combining of the visual and aural elements to create a singular experience. By using abstract visualizations, one can hope to reach a much wider cross-cultural audience.

Musicians often speak of seeing colors or patterns when they are playing music. These patterns are often described as abstract forms. As a musician and artist, I try to put these forms into illustrations and animations that can be appreciated by the listener visually as well as aurally. For me, there is an analogy between the harmony in music and the design elements in an illustration. The melody will often suggest a line or color selection. I do not try to depict the music literally, but rather to take it's elements and use them as the basis for the abstract piece. I then work with the piece in the computer from a visual standpoint to create the final illustration or animation.

The idea of a visual component to music is not new. Annie Bessant and C.W. Leadbeater discuss music as generating visual structures in their book "Thought Forms" first published in 1901.They put forward the idea that music triggers a "thought form" that is expressed by the musician from the original ideas of the composer. These forms take shape and build themselves the same way that the musical composition does. These thought forms take the shape of large abstract images emanating from the source of the performance.

More recently, in an article in the Wall Street Journal, Prof. Manfred Clynes has spoken of "shapes" which are characteristic of different composers. When people were asked to move their hands to "conduct" a piece of classical music, he found that different people all used similar motions when "conducting" the same composers. He has used these "shapes" to write a program that composes computer music.

History

A look at the history of the visualization of music begins with the evolution of visual projection systems. As early as the 1700's, composers were experimenting with devices called "color organs", which related and projected colors to sounds. Perhaps the most obvious early example of music visualization is the combination of fireworks with the 1812 Overture. This is an early marriage of music and visuals in a very dramatic form.

Walt Disney's "Fantasia" is perhaps the best known commercial exploration into the animation of music. Some of the pieces were abstract, while most had character animation and a story line. Well known for his work at the National Film Board of Canada is Norman McLaren, who did drawing by hand on film and other experiments with optical sound tracks.

One of the early pioneers in the abstract visualization of music on film was Oscar Fischinger. His work is composed of abstract shapes that move along with the music. Some are tightly synchronized, while others are not. He started his film work in the early 1920's using a wax slicing machine that revealed a sequence of abstract colors and shapes photographed one frame at a time. Since color film had not been developed yet, he tinted various pieces of film. In the late twenties, he created a number of black and white "Studies" that were closely synchronized with music. They were drawn as black shapes on white paper and then the film was used in a negative form, giving white images on black. Fischinger came to the United States from Germany in 1936. While here, he worked for Paramount and MGM creating a variety of animated abstract films. He also worked with Disney for a short time on the "Fantasia" film, but resigned over creative differences he had with the Disney staff. His final film, "Motion Painting Number One", was completed in 1947 and was tightly synchronized to Bach's Brandenburg Concerto #3. It was animated by painting on a canvas in front of the camera and then adding plexiglass sheets in front which were also painted on. The film was eleven minutes long and was shot in one "take" over the period of six months.

A more recent pioneer in computer animation with music is John Whitney. He did a lot of early work using devices such as a modified anti-aircraft sighting mechanism and early computers to generate his films. This was the birth of what was called "motion control". That is, computer control of the operation of motion picture cameras. His films consist of abstract images that look somewhat like sound waves undulating and reacting with each other.

Musical Structure

Musical structure takes many forms. The style of music generally dictates its form. Classical music has its own set of rules. Jazz is based on improvisation, and so on. It is not the intent of this paper to describe all forms of music, but rather to elaborate on the author's approach to composition and how it relates to his own abstract visualization of music, as well as to suggest ways by which other artists and musicians can approach this process.

The three main elements of music are melody, harmony and rhythm. Each of these elements can be a focus for the abstract animation of music. Melody is the path by which the music dictates meaning to the listener. Is the melody quick, slow, melancholy or lively ? By structuring the melody in various ways, the listener can be lead in many directions. From a visual point of view, the melody consists of a rising and falling set of notes of varying duration and pitch. It can be line that creates the difference between positive and negative space. Or, it can form a border that is constantly changing as it passes the camera. The pitches can also be assigned to a variety of colors which will change as the notes change.

Harmony creates resolution and tension in a musical piece by the formation of chords and an underlying structure upon which the melody resides. Songs are written in a key and this key dictates which chords can be used in the composition. To an animator, this harmony can lead to a set of recurring images, colors or themes which form a bridge between the melody and the rhythm. These can be used as backgrounds upon which the animation moves, or can act as a set of recurring patterns or shapes which give the animation form and structure.

Rhythm represents time in music. All songs have a definite length. There is generally an introduction, a middle section and an ending. In addition to this, there is a time signature and a tempo. All these things can be incorporated into the process of visualizing music. It is convenient to compose music that is in some ay a multiple of the frame rate of video (30 frames per second) or film (24 frames per second). In this way, the frame counts can more easily be located when reading the soundtrack. Rhythm is perhaps the strongest of the three elements of music, since it forms the basis of the music and can most effectively reinforce the sychronization between the visuals and the sounds.

When listening to a musical piece that one is going to animate, it is important to consider each of these three elements individually. Listen to it once for only the melody, then listen to it solely for the rhythm, etc. By narrowing the focus of your attention, visual ideas can be more easily generated.

Visual Structure

Once a piece of music has been listened to and certain visual elements have been suggested, how does one now put it into a final form that is synchronized with the soundtrack? One of the first things to do is to sketch out roughly what images the music is suggesting. It is important in the creative process not to go too quickly with one's thoughts. By experimenting with a variety of treatments, one can gain a clearer idea of what one is trying to say. These rough sketches will eventually become many sketches and will take a sequence related to the music. This is what is called the storyboard. It is a visual description of the final film. It is important to mention that making a film or video is an evolutionary process. From your first idea to the final film, there are many stages. It is common to redo sections of you piece many times before the final version is decided upon. However, it is easier to redo storyboards that are drawn in ten minutes than it is to redo the final artwork that may have taken many hours to create.

One common technique of musical composition used by film composers, and myself, is to create a rhythm track first. This method is often used for live action films. The composer looks at the film and creates a rhythm "hit" for each important event in the particular visual sequence. Once this set of "hits" is made and recorded on tape, a rough version of the film or videotape is edited along with this rhythmic bed. This is played over and over until it begins to generate ideas for the melody and music. This technique can be applied to animation as well. Normally, a certain set of visuals are suggested by a musical piece. These can be sketched roughly and then synchronized visually on film or tape to these hits. When played back, it will form a skeleton upon which a more developed piece can be made.

Traditional Soundtrack Analysis

The traditional methods of synchronizing animation to music involve using magnetic film stock. The original music was generally recorded as a 1/4" full track master. This master was then transfered to magnetic film stock. This stock is just like 16 mm or 35 mm film except that it has a magnetic coating on it (similar to magnetic tape). Given the fixed length of a frame of film, 16 frames per foot for 35 mm and 40 frames per foot for 16 mm, the music that was transferred to the film now can be located accurately to a 24th of a second. This is done using a synchronizer. The magnetic stock is placed in the synchronizer, a zero frame reference is located and the track is "read". Reading a track is the time consuming task of running the stock over a sound head in the synchronizer and writing down the foot and frame locations of each sound. This is done on an exposure sheet or a bar sheet. This sheet becomes the map by which the animation is done. By having an absolute reference on film for each sound, when the animation is photographed to that reference, both sound and picture can be edited together in perfect synchronization.

Modern Soundtrack Analysis

A more modern apprach to synchronizing animation to music is through the use of MIDI, the musical instrument digital interface. MIDI was a musical standard introduced in the early 1980's that allowed manufacturers to make various keyboards, synthesizers and drum machines compatible with each other. The MIDI standard involves the output in digital form of various musical parameters, including note on, note off, pitch, velocity, aftertouch, etc. These are referenced to MIDI time code which tells the exact time of these events. his data is recorded by sequencing programs allowing multitrack recording and playback. MIDI is attractive in that it retains all the nuances of the original perfomance, yet it can be edited in a manner similar to that of a word processor. Passages can be cut and pasted, rearranged, looped, and harmony parts added without additional musical performance. This altered data can then be played through the synthesizer it was recorded on or any other MIDI device. Thus, a complex melody could end up as a rhythm track. MIDI has revolutionized the music recording industry. Every major studio now uses MIDI.

In addition, MIDI and video synchronization is evolving rapidly. This has come about through the linking MIDI time code with SMPTE time code, the standard for the video industry. By using the proper synchronizers, audio recording decks, MIDI devices and computers can be locked in a frame-accurate manner to video recorders.

This has many applications when it comes to the abstract animation of music. Sequencers now have a variety of graphic editing functions, where the music can be seen in a variety of graphic forms. This in itself can be an aid to developing visual ideas for a musical piece. Through using screen savers and other printing techniques, these digital maps of the music can be used as an exposure sheet or bar sheet, thus saving the time consuming step of reading the soundtrack as done in the traditional method in film.Another method for the synchronization of animation with music is to actually program the computer. This is a very time consuming process, but the rewards can be very striking. Wayne Lytle's film, "More Bells and Whistles" uses computer programming, 3-dimensional modeling and MIDI to create its effects. The video starts with a marimba and then adds additional instruments as the music becomes more complex. CGEMS (Computer Graphics/Electronic Music System) is a program which correlates the musical score with graphic parameters. The music was recorded in a MIDI studio and then rendered using Wavefront software. There is an amazing synchrony in this piece between the visual elements representing the music and the music itself.

There are many commercially available software packages that allow for the mixing of visual elements and sound. This new area is called Multimedia and is the lastest buzzword in the personal computer business since Desktop Publishing. One of the first to demonstrate this was Macromind Director for the Macintosh. A comparable program on the IBM PC platform to Director is Autodesk Animator Professional. Ovaltune is a visual composing environment by Opcode. New on the horizon is Apple's Quick Time, which allows for the digital recording and playback of video sequences. Many other software packages using the principal of visual with sound are available and will continue to be released. It is up to the artists and musicians to create art with these packages. It must be remembered that the computer is no more than a creative partner. The old phrase "garbage in, garbage out" must be rewritten to be "inspiration in, inspiration out".

Final Product

Once you have written and recorded the soundtrack, read the track and created the final visual elements, they must be output to either film or videotape. The film approach uses either a stop motion camera or an Oxberry animation camera which allows these images to be photographed one frame at a time. Video cameras hooked into a single frame editing deck work in a similar manner. There are some computers that can play animation and music in real time. It is not the scope of this paper to describe these techniques in detail. However, the final marriage of image and sound is usually done by recording the music on film or videotape, then recording the images on film or videotape and then editing them together into a final product.

Conclusion

The use of abstract images to visualize music has been around for about seventy years. From the early pioneers like Oscar Fischinger to recent computer artists like Wayne Lytle, there have been a variety of technical methods employed. This artform ranges from hand-drawn animation with manually reading a sound track on film to programming a computer and recording the music using MIDI. The development of Multimedia software will continue to provide a set of sophisticated tools for the musician and artist. Although these tools have radically changed, there still remains an interest in the abstract visualization of music.

About the Author

Bruce Wands is the Faculty Advisor to the MFA Computer Art Program at the School of Visual Arts in New York. He also teaches graduate and undergraduate courses in 3D Animation, 2D Animation and Computer Video. In 1989, he was awarded an NCGA Educator Scholarship. He is an award-winning independent producer who has produced animation, titles and music for such clients as AT&T, United Technologies, General Motors, Air Safety Foundation, RCA International and others. He became involved with computer animation in 1976 when he was an animator for the Spectacolor computer animated billboard in Times Square, New York. While there, he designed the computer animated opening for NBC's Saturday Night Live. He has a BA from Lafayette College and an MS from Syracuse University.

Suggested Reading

Russett, Robert and Starr, Cecile, Experimental Animation: Origins of a New Art, DeCapo Press, New York, NY 1988

Thomas, Frank and Johnston, Ollie, Disney Animation: The Illusion of Life, Abbeville Press, New York 1981

Leonardo, International Journal of the Contemporary Artist, Maxwell House, Fairview Park, Elmsford, NY 10523

Electronic Musician Magazine, PO Box 41094, Nashville, TN 37204

Animation Magazine, 4676 Admiralty Way, Suite 210, Marina Del Ray, CA 90292

Collaborating with the Computer: Generating Meanings from Random Sequences

by Stephen A. Wroble

Introduction

Computer art is often discussed within the context of the cinema. (See BERTON, FOX) This is understandable when we realize that the most visible forms of computer art today are presented in a cinematic format. Advances in the technology of animation and special effects are directly tied to the movie and broadcasting industries.

We think of interactivity as a kind of user controlled cinema. (See DAVENPORT et al, CRANE, HOLSINGER) The viewer acts as editor to produce the finished work, individual elements are arranged according to the user's own desires. The viewer is now a participant, and so has a greater responsibility. He or she cannot remain passive but must take an active part in the presentation. In a sense, the artist and the user have become collaborators. This is an exciting new way to approach cinema.

The meaning is still controlled by the artist. Although the user has complete control of the sequencing, the artist controls the elements and the choices. The user can move down predetermined pathways, but the artist prevents the user from moving in meaningless directions. In many cases, the individual elements are self contained. Conclusions are predetermined and the arrangement doesn't contribute significantly to the meaning.

Within the structure established by the artist, the user has complete freedom. The choices, as far as the artist is concerned, are random. Staying within a structure helps preserve the meaning. If the artist has a message, the structure must be fairly rigid in order for the sense of the message to be preserved.

Yet, while the computer makes new approaches to cinema possible, it also offers the potential to be expressive in other ways. By interacting with the computer itself, rather than the viewer, it is possible to arrive at a different kind of expression. The computer can be used as an artistic medium in itself, rather than as a tool within another medium.

Through it's ability to generate random numbers, the computer can be made to take part in the creation of meaningful sequences. My work is based, in principle, on the work that has been done in the areas of computer generated poetry and music - except that I have been using images rather than words or notes. In effect, the computer becomes the editor in a way analogous to the user in an interactive presentation.

But I can ask the computer to make many more choices than a human user would be willing to make. The result is an expression that I believe has meaning, but not necessarily a meaning that I control. It's possible for someone viewing my work to arrive at a meaning that I am not aware of. That's the exciting part.

Before I go into a discussion of computer applications, it would be productive to look at how meaning itself is generated and how messages are communicated.

Semantic Theory of Meaning

From the field of communication we know something about how messages are passed from sender to receiver. In order for communication to exist, the receiver must be able to properly decode the message - a shared set of symbols is required. In the arts, socially defined symbols are not the norm, yet viewers often find meaning in the work. How does the receiver, in that case, make sense of the expression?

In order for something to have meaning, there must be both content and context. Content refers to the form of the message, it could be symbols like letters, or it could consist of something more artistic: notes in music for example, or even colors, shapes, and textures.

Context refers to how the message is conveyed, in language, the context may be grammar or syntax. In art we would call it design or style. Context can exist on several levels.

The most obvious level is the social context. Most

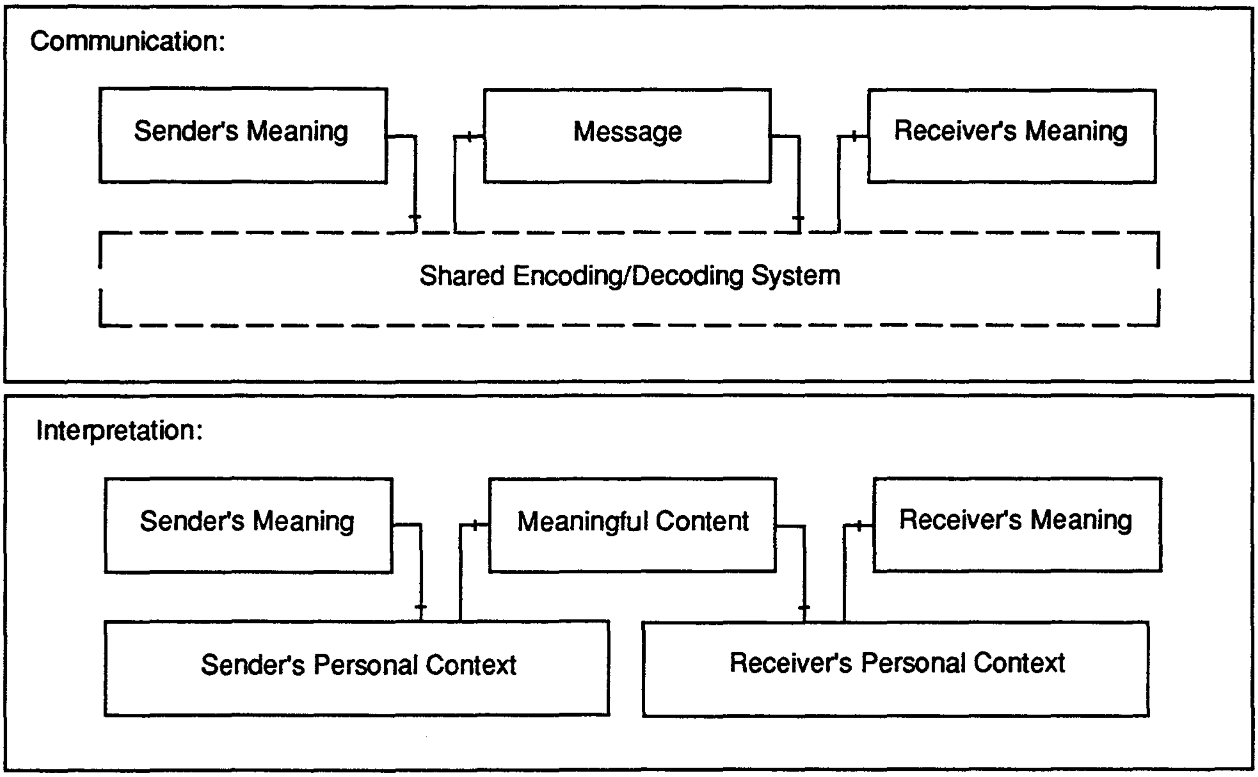


Figure 1

discussions of communication theory deal entirely on this level. Not only do we have socially assigned meanings for symbols like letters, we also share cultural associations that affect the way we perceive certain images. White stars on a blue field together with red and white stripes carries a meaning to most Americans.

The most basic level is the immediate context. What came before, what comes after, and what comes simultaneously, all affect the meaning of the content. This is well known in the cinema where the editing has great impact on the meaning of the film.

A third level, and the most important to this discussion, is the personal context. What an individual knows can color the meaning of new information. Certain colors or arrangements of objects may carry connotations for people which will affect the way those things are interpreted. The personal context is probably used more in the arts. The artist arranges elements in a way that has meaning for the artist. The viewer is able to interpret those elements in a way that has meaning for the viewer, regardless of the artist's intentions.

In his paper on "Generative Techniques in Graphical Computer Art," John Lansdown also makes the analogy of language in discussing meanings in art. He refers to 'modules' of information and their arrangement. He points out that "Ambiguity and multiple meanings are inherent in such things" as music, poetry, dance, and so on. (LANSDOWN pg 67)

The reason there are multiple meanings is that each viewer brings their own context to the piece. The viewer receives the content but interprets it through a personal context to find meaning. It should be possible to receive a message that is not necessarily sent. In other words, people could be able to find meaning even if the artist didn't intend a message.

(SEE FIGURE ONE)

Semantic Meaning in the Cinema

In the early days of cinema, some important work was done on theories of meaning within the context of cinema. During the 1920s, filmmakers began to seriously examine how the audience's understanding of the film could be enhanced through the sequencing of images. Most important, I think, was the work of some filmmakers working in the Soviet Union, they developed a style of film editing which is now called Soviet Montage. The Soviet filmmakers formulated some theories about how the viewer perceives the stream of images in a film. These techniques made the cinema a more expressive medium.

Lev Kuleshov, working in the '20s, showed that by linking fragments together, meaning could be created. His technique was to carefully choose images and assemble them in a certain order with each image adding to the image before to affect the overall meaning.

He performed some simple experiments which showed that audiences could understand a message which the actors never intended. For example: by sequencing images of an actor with images of food, the audience believed the actor was acting hungry. If the same shots of the actor were intercut with other scenes, the audience concluded that the actor was acting some other way. The audience clearly reached a conclusion based on sequencing rather than acting. (GIANETTI pg 132)

In a similar fashion, Kuleshov showed that through editing, audiences could be made to believe that two people were holding a conversation, when in reality they had never met. Kuleshov demonstrated that the sequencing of images was more important to the meaning of a film than the images themselves.

The most well known of the Soviet Montagists, at least to most Americans, is probably Sergai Eisenstein. He believed that meaning could be achieved through the 'collision' of images. The juxtaposing of extremely different images would result in meanings not contained in the images themselves. While Kuleshov assembled meanings by linking images in sequence. Eisenstein believed that new meanings could be synthesized in the mind of the viewer by observing elements in sequence. His images were not self contained, but only meaningful when part of the whole.

What's important to us is the idea that meaning can be constructed through juxtaposition. If we consider the semantic theories of meaning, we see that the editor is manipulating the context of the message.

The theories of film editing could be compared to the theories of the Cubists. (Film historian Eric Rhode draws parallels to the Futurists. See RHODE chapter 4) We see reality broken into fragments and then reassembled in a way that is not essentially real, but carries greater meaning than simple realism. A distinction is made between information and meaning (or between content and meaning.) The way the information is presented contributes to the meaning.

In some ways, the Soviet Montage school is considered rather heavy handed. Kuleshov believed that the editor must control every juxtaposition of every image, both within the frame and in the sequencing, otherwise the viewer would impose their own interpretation of the scene. There is not much room for the viewer to form an original idea. All associations are dictated by the editor.

Yet, if Kuleshov and Eisenstein are right, the viewer should be able to reach an understanding of any two images, whether the juxtaposition is intentional or not. Random sequences could even be subject to interpretation if the viewer can make use of a personal context to draw conclusions.

Computer Generated Randomness

Work has been done since the 1950s on the use of computer generated randomness in the creation of aesthetic expressions, particularly in the areas of music and poetry. The work of these poets and musicians involved a kind of controlled randomness, there were limits to determine which elements were selected at each step. Meaning was controlled by limiting the pool from which elements could be chosen as well as by creating a context.

In most cases, the computer was given a set of criteria by which to test the randomly selected elements. In the case of music, for example, the computer would test each randomly selected note to see if it conformed to the given rules of style before accepting it.

Theo Lutz published some of the earliest computer generated poems in 1959. He used the rules of grammar as the context to give meaning to the randomly selected words. A noun and an adjective were linked by a conjunction, for example. By limiting the vocabulary from which elements were chosen, the resulting poems made sense. (FRANKE pg 97)

Similar work was done in purely visual constructions made by artists like A. Micheal Noll. His work "Gaussian Quadratics," shown in 1965, investigated the visual effects of programmed randomness. A series of lines were constructed from end points which were determined randomly. (GOODMAN pg 24)

It should be pointed out that a measure of control is needed in the randomness. The artist must impose some order for the work to be meaningful. In the case of poetry, a purely random list of words, would simply be a list. But by limiting the randomness, meaning can be achieved. In a Gaussian distribution, the random points tend to cluster around a mean value; it is not a pure randomness.

In his book *Computer Graphics/Computer Art*, published in 1971, H.W. Franke examined the creative process itself in relation to the computer. He pointed out that the artist makes certain choices in the creation of an aesthetic work. The choices are not random, but governed by the artist's experience and intuition. But, at the same time, there is the element of chance coupled with the artist's ability to recognize meaning in accidental configurations. Serendipity happens.

Franke expressed what he called "The Random as The Generative Impulse." He pointed out that if the context is too well controlled, the element of chance is eliminated and creativity is hampered. A measure of randomness is desirable to make the unexpected possible. "As chance destroys order it creates more complex structures and achieves the unexpected, the unforeseen, to which we react with surprise." (FRANKE pg 117)

Notice that an accidental juxtaposition itself is meaningless unless the user is able to recognize something meaningful in it. The user must be able to interpret the result. In computer generated constructions, we are expecting the viewer to impose their own context on the content in order to arrive at an understanding of the work.

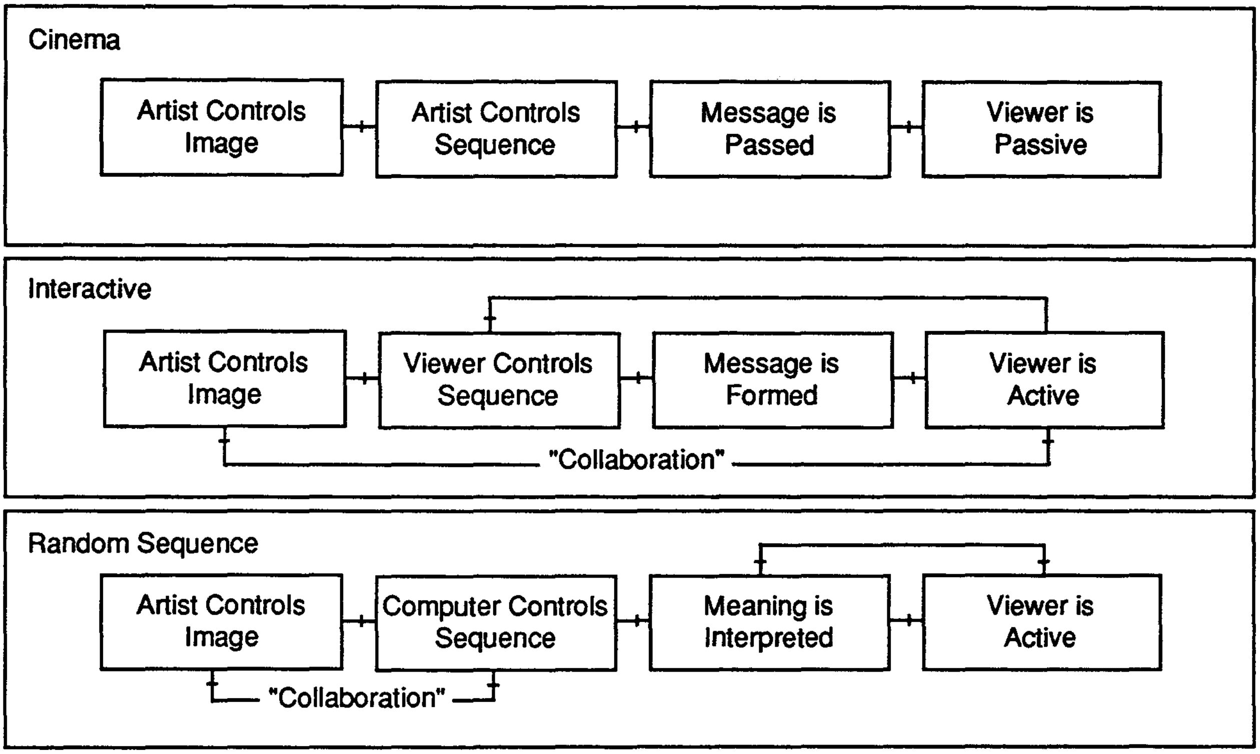


Figure 2

Revelation Through Chance

It should be mentioned in passing that the idea of finding meaning through random juxtapositions is not new. An idea current in the Middle Ages concerned the possibility of revelation through chance. I won't go into all the details of this medieval philosophy because most of them don't apply here. The idea is based on the assumption that because everything is part of a whole, each part must have a relationship to every other part. If one can discover the relationships, one can somehow become wiser. By examining random relationships, one can arrive at knowledge that didn't exist before. A simplified version of this idea is behind the use of Tarot cards and other such forms of divination.

This idea was current before the Renaissance turned us toward more rational approaches to knowledge, and so has nothing to do with logic. Nevertheless, in terms of the subject of at hand, this idea is still relevant. It is possible to apply the general principles of this medieval philosophy to a small set of related elements rather than to the universe as a whole. There can be no doubt that Eisensten's audience arrived at an understanding, and that Lutz's poetry made sense. Meaning, in all these cases, depends entirely on the user's ability to interpret - to apply a context to the content.

Summary of Work in Progress

In an interview in 1988, computer artist Lillian Schwartz said about her work done in the early 1970s: "I love the idea that back then you interacted with the machine and the machine became a collaborator. And because of a mistype some wonderful new thing would happen. I miss that" (quoted by HOFFMAN pg 235)

It is interesting that Ms. Schwartz used the word 'collaborator' in describing the way she interacted with the computer. The reason, of course, is that results were sometimes unexpected. She had gone beyond the simple command/response process. Accidents became a part of the creative process, requiring the artist to continuously re-evaluate the direction of her work.

Also interesting is the implication that things are not that way anymore. The opportunities for serendipity don't exist for most of us. Computers have become so highly structured that even the accidents are predictable.

My work is an attempt to get beyond the structure. I am not trying to send a message in the traditional sense of communication, but I am inviting the user to discover a connection between the various aspects of an idea. Because the viewer must draw on their own context to find meaning in the random sequence, it is hoped that the viewer will be more intimately involved with the piece. It is possible for the viewer to learn something which I don't already know.

It is not necessary for me to go into a discussion of the nature of randomness from a mathematical point of view. Nor will I discuss the merits of the various kinds of random number generators now in use. For my purposes, it is only necessary to have a degree of unpredictability, so very primitive kinds of pseudo-randomness are acceptable. The important thing is to be able to achieve unexpected juxtapositions which are not the result of human determination.

My work involves the random sequencing of words and images. Working entirely with off-the-shelf software, I create a data base of elements which are related thematically, and allow the computer to display those elements in a random sequence. It is hoped that the viewer will perceive relationships between the elements and so arrive at an understanding, which may or may not be similar to my own understanding.

Essentially, I create a series of loops. The selection and display of a single element is accomplished by each execution of the loop. The computer may exit the loop or move to another loop by means of exit commands embedded in the data base. Eventually the computer will select the command instead of another element.

The elements displayed during the execution of a loop are related by theme. This is how I limit the randomness to ensure that some kind of meaning is possible. The collection of elements in the data base, represents my total understanding of the theme. But they are fragments. My work could be understood as a kind of deconstruction technique, with the computer in charge of the reconstruction process. The viewer is therefore invited to reach their own conclusions.

I am more concerned with the philosophical implications than with the purely visual. Randomly generated images are not the same as randomly generated meanings. There is a potential for new levels of intellectual involvement, beyond that available in conventional visual arts.

Conclusions

The emphasis in traditional cinema and broadcasting is on the projection of a given message. These forms of communication must therefore be highly structured and based on a context shared between the sender and receiver.

With CD-ROM technology we can have something that approaches interactive cinema. When we move into the interactive environment, the problem is to give freedom to the receiver without losing the structure. The ultimate meaning is determined by the artist who creates the original elements and provides the choices for the user. Glorianna Davenport, from MIT, has pointed out that interactive installations often become a series of complete messages which are downloaded in a sequence determined by the user. (DAVENPORT pg 14) But we still have a sender/receiver relationship.

The artist also creates the elements and limits the choices when the computer becomes the editor, but the meaning no longer has to be predetermined. The computer can work fast enough to create the necessary context. The user/viewer must become more intellectually engaged in order to interpret the sequences. It is the interpretation itself that creates the meaning.

(SEE FIGURE TWO)

The computer can become a participant in the creative process just as a user/viewer can. When the computer is allowed to control some aspect of the process, we move away from the traditional command/response function of the computer as tool, and move toward something more unique: the computer as collaborator. If we move outside the context of cinema, we can begin to see the potential of a new medium.

Bibliography

Authors in all caps are quoted in the text. The others have bearing on the subject but are not specifically cited.

Berton, John Andrew - "Film Theory for the Digital World: Connecting the Masters to the New Digital Cinema" - Leonardo, Supplemental Issue, 1990, pg. 5-11

Crane, Gregory - "Hypermedia and the Study of Ancient Culture" - IEEE Computer Graphics and Applications, Volume 11, Number 4, July 1991, pg. 45-51

Davenport, Glorianna, Thomas Aguierre Smith, and Natalio Pincever - "Cinematic Primitives for Multimedia" - IEEE Computer Graphics and Applications, Volume 11, Number 4, July 1991, pg. 67-74

DAVENPORT, Glorianna - Interview with Simson L. Garfinkle "Picking up the Narrative Thread with MIT's Glorianna Davenport" - New Media, Volume 1, Number 4, June 1991, pg. 14ff.

Fox, Edward A. - "The Coming Revolution in Interactive Digital Video" - Communications of the ACM, Volume 32, Number 7, July 1989, pg. 794-801

FRANKE, Herbert W. - Computer Graphics/Computer Art - Phaidon Publishers, New York, 1971

GOODMAN, Cynthia - Digital Visions: Computers and Art - Abrams, New York, 1987

GIANETTI, Louis - Understanding Movies - Prentice Hall, Englewood Cliffs, New Jersey, 1990

Hamilton, Edward A. - Graphic Design for the Computer Age - Van Nostrand Reinhold, New York, 1970

HOFFMAN, Kenneth E. and Jon Teeple - Computer Graphics Applications - Wadsworth Publishing, Belmont, California, 1990

Holsinger, Erik - "Ulysses: A New Hope for Education" - New Media, Volume 1, Number 4, June 1991, pg. 24-26

LANSDOWN, John - "Generative Techniques in Graphical Computer Art: Some Possibilities and Practices" - in Computers in Art, Design, and Animation, Springer-Verlag, New York, 1989

RHODE, Eric - A History of the Cinema - Hill and Wang, New York, 1976

Youngblood, Gene - Expanded Cinema - Dutton & Co., New York, 1970

Computer Artists as Interactive Performers

The First Digital Transmission Via Satellite of a Real-Time Drawing

Lillian Schwartz
Bell Laboratories Rm #2D503
600 Mountain Avenue
PO Box 636
Murray Hill, NJ 07974-0636

Purpose In Making the Entry

For the first time we are able to interact with other artists and perform in real time for audiences around the world by transmitting digital data signals over a telephone line via satellite. The purpose in making this entry above submitting any one of my other works, created with technology using better resolution and infinitely more colors, is based on exploring this new form of computer art where the performance would be a form of theater.

Ever since I started adapting computers in 1968 as a medium for art, including graphics, film, video, special effects, multimedia events, and sculpture, I have directed my efforts to extending the technology to afford new ways for artistic expression for myself and other artists. Often I pose the problem and work at solving it. Sometimes requests and suggestions are presented by others. Any path that helps extend my development as an artist is a challenge as well as a necessity.

Consequently, when AT&T Public Relations asked me if it was possible to find a way to create a work of art that would be transmitted in real-time over a telephone line to celebrate the 500th Anniversary of the Postal Bundespost Museum, in Frankfort, Germany, I automatically said yes. I was excited about a real-time performance piece over long distances and felt I would find a means to make this event happen. I was intrigued with the idea of creating for a live audience in another location, or possibly collaborating on an international level in an interactive mode. This was a unique opportunity to explore another way of working with expanding *Pixellence,* the computer art medium.

Description of Work Process

Since this would be the first time a drawing would be transmitted over a telephone line there were no previous guidelines. The first requirement would be to keep the amount of data transmitted to a minimum, even if it meant working in a lower resolution and with fewer colors than I am currently accustomed to. But the potential for adding another dimension in working with the computer medium was compelling.

After consulting with my colleagues I opted to use a recently invented pressure sensitive pen and tablet. I would be able to sketch varying line widths as easily as I could with a conventional pen or brush but I would be limited to a palette of 8 colors. I would work with red, green, blue, yellow, black, white, and grays, and have the capability of mixing these colors. Anything more would slow down the transmission.

The advance was only partly technological. I could now bring back or put into it my early traditional method of handling a tool and drawing/painting and, I could also put in the intellectual understanding and use my expertise in animation, traditional painting, understanding effects of color, and their impact on the perceptual responses of the audience.

Once I had the tool in place I turned my attention to the project itself Sometimes when I work I follow my intuition to see what will work and what doesn't But this project had quite different boundaries than those I had worked with before. I would be creating for a live audience and with time constraints. And, while the real-time work would be animated, the final creation would be viewed as a fixed image, not on the back-lit screen where it was realized, but on the opaque surface of paper used with the decided-upon Tektronix color printer. The major issue, however, was what I would paint.

At first I thought Abraham Lincoln would be a good choice. And then Benjamin Franklin, our first postmaster General. I did sketches of the two as well as one of an American Indian. Any one of these portraits would have worked for this purpose but I wanted an image that had meaning for me and one that the audience would easily recognize. Then I hit upon Beethoven.

I often play music while I work and had a Beethoven piano concerto in my disc player. As soon as I decided on Beethoven, with his music to set the rhythm for my brush strokes, I could turn my attention to the mode of operation I would take towards a hopefully successful performance and finale.

I had a subject but now needed to address questions concerning the performance. Since the final image would be still should I then work at creating a painting emphasizing stillness? How can I best keep the audience's attention? Where would they be seated in relation to the screen. How large would the auditorium and the screen be? Would their be multiple screens? Should I avoid perspective and use color and line alone to draw the viewer in? I had learned very early on in my art career that the placement of greys in large areas with the positioning of pure colors as accents most often realized this effect.

In the first stages of the practice sessions I treated all the questions with the most negative answers. I filled the screen with the face and allowed for just enough of the hair to afford a recognizable clue. I studied the the overall shape of Beethoven's head, the ringlets of the hair, and the identifiable frown. I worked on the features in a meticulous fashion and then abstracted the face. Even under the best circumstances it would be difficult for an audience to perceive tiny plxel-size marks. Besides, I knew I would not have the time in the allotted 7 - 8 minutes for such precise detail on September 27, the scheduled transmission date.

The Dry Run

Bob Boie, my colleague and the inventor of the pen, left for Germany one week before the performance with a duplicate set-up including a pen, tablet, software, computer, and printer. We scheduled a dry run between Germany and the US to test that all was working.

For the dry run I concentrated purely on the use of the tool and my imagination, but deliberately restricted myself to sketching heads, mostly of Beethoven. It was exciting to watch the images being drawn back and forth, knowing that they were coming over a telephone line in real time and from so great a distance. In reality this practise session on September 26 was the first international transmission of art over a telephone line. All went well and I went back to my preparation for the big day.

The Performance

We decided that the clue for the start of my performance would be a message from Dr. Werner to me written on the screen. I started the disc player with the Beethoven music and waited At the scheduled time Dr. Werner wrote, *Good morning Lillian* on his screen as I watched it appear on my screen. I then wrote *Good morning Thomas,* erased the screen, and started the performance.

I began by laying in colors from my pre-selected palette. Then I rapidly sketched in Beethoven's hair. I knew the hair and the face would look like chaos at first. I followed my plan sketching round and round, searching out the contours of the head.

Even though the Maestro's hair is a strong identifier I didn't think the audience would guess who I was drawing until the characteristic frown and other features were drawn in. I was glad I decided to sketch the wiry, unkempt hair first since I found I had a bit of stage fright and my hand was shaking. By the time the hair was almost in place, and I began feeling out the structure of the face, I had calmed down. I could now work more slowly. I searched out the positions for the eyes, nose, and mouth, going around the face, inside and out, and then in the final minutes positioned stronger lines to emphasize the cheek bones, added a few strokes to suggest the clothing, and continued to pull out the features in reds, blues and purples.

I deliberately worked around the periphery of the screen, touching here and there in the middle to get a sense of placement of features to capture the attention of the viewers. If I had drawn the features first I would be drawing in one restricted area and with small strokes which may not have been discernible to the viewer, and therefore boring. The larger, rapid positioning of different size strokes and changing colors held the spectators attention, and helped draw them into the performance. I also changed colors frequently to deliberately cause the viewers to pick up on another color to be brought into the process itself. This allowed for the audience to be aware of the steps in the creative decisionmaking.

Bob told me that the audience was busy talking and restless during the previous talk presentations but, when I started the painting everyone sat perfectly still - captured by the colors moving around the screen, anticipating the outcome, and working to identify the image.

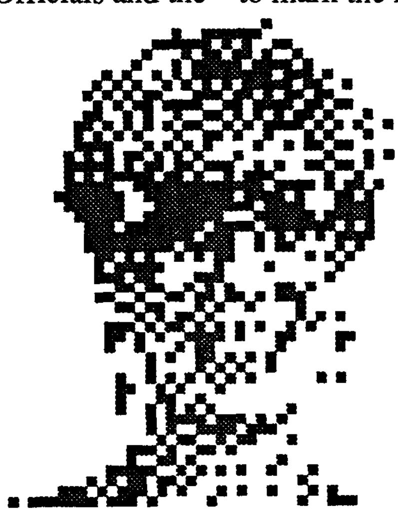
The Final Stage: A Color Print for The Museum

The only way the final image of Beethoven could be presented Immediately to the Officials and the Museum was to send the image to a computer-controlled printer.

Over the years I have made etchings, lithographs and silkscreens, but I have always been bothered by the lack of spontaneity. Any changes were laboriously made between each pulled print, the reworking of the composition, the inking, and pulling another print. Watching the print dry, aware of the differences when the ink was wet or dry. Hours and hours alone with an assistant.

Now with a printer I can work by myself. The image can be captured at any moment. The changes are made immediately. I see the print, return to the computer to alter the image, and print again to test the changes. With the printer close by I can work with great speed and responsiveness. It is the closest I have ever come in printing to what it's like to paint. Each print is an original. They are not reproductions.

In most other more traditional printing processes the image seems to be below the sheet of paper but with a computer printout the process is more direct. There need be no operator. A command from the computer sends the image to the printer which then cogitates a few seconds and the image appears directly as a waxy imprint directly onto the paper. Actually the image appears to right on the surface of the paper. So what I tried for in the final work was to get the beauty of the color itself on paper. It's there right on the surface before us. It was this final work of Beethoven that is hanging in the Museum to mark the first digitally transmitted drawing.



Social Domain Programming

Utilizing Media & Technology within the Context of the University's Urban Environment

In the computer graphics course called *Elements of Electronic Image*, DeskTop Publishing and Presentation Graphics are immediately linked in the first week's class to the media needs of the university's community, our surrounding city. The distinctive character of these 'courses' lies in this Practicum, essentially offering a concrete, factual experience wherein students are given texts and design problems drawn primarily from the publishing needs of government aligned agencies and institutions. These texts and graphics are often printed and used within a larger social setting before the student's first semester coursework is completed.

Shifting Educational Focus:
A Practicum-Centered Context

This Practicum, woven through an enlarged social system, contrasts with University courses that are usually compartmental, within a department's program, blocking out a degree which lies at some distance from qualifying the student for re-entering the social context just outside the quadrangle of the college. Yet, on a daily basis, that same urban student, and particularly the inner city student, crosses and recrosses the harsh realities of the street on his way up the academic ladder. Traditional application of knowledge, outlined and abstracted, appears at a very marked distance from the contrasting reality of life experience in the city.

Practicum Examples:

The daily crisis ***Bellevue Hospital*** Complex must face in surgeries, convalescence and preventative medicine needs communicative documentation. Good design wasn't essentially in their plan, yet Brooklyn College's students developed fifty pages of bi-lingual handout information all enhanced with the addition of graphics improving the patient relations dialogue.

The New York City Department of Cultural Affairs provided over eighty pages listing spaces where artists might hang their work free. Brooklyn College's students reduced the number of pages in half while providing an easier to use document that primarily scaled typeface rela-tionally to information searching. The mailing costs were also reduced by half while the clear enhancement of well designed texts probably doubled their usability.

The Outreach Program:

On Sunday mornings our design students, as volunteers, teach design to inner city elementary school children. This strategy recognizes the ecological principle of giving back into the culture from where one originally received his first help. The children are brought to the project through Teach For America and Brooklyn Church Groups. The inspirational ideal of a peace corps in inner city education gave rise to the concept that Teach for America members could, as talent scouts, search out elementary school youths

Social Domain Programming

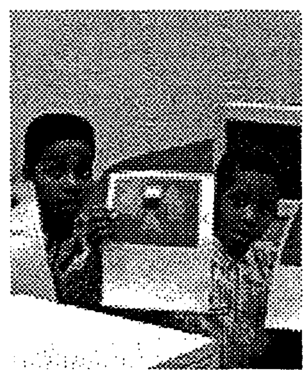
Utilizing Media & Technology within the Context of the University's Urban Environment

whose natural visual literacy in sketching or writing could be catalyzed by interactivity with Brooklyn College students who show them DeskTop Publishing.

Visual Literacy, as we conceive it, offers a new imagistic threshold toward gaining a broad literacy. The image and its relevance to textual flow became the keystone for a deeper fusion with the concerns of Euro-centric traditional education.

Though our students are newly initiated to our media sequence, the outreach forced them to articulate their knowledge in clear form to the youths brought by Teach For America members – once again securing the core of their own knowledge around the immediate articulation of our living laboratory for media expression.

Integration of this socially active role of students through the Practicum and the outreach within the fabric of the city is a natural event in our courses. After all, they are about media, mixing skills of photo management, text construction through typeface design, page and layout skills related to printing, video and still photography, as well as video mixture of graphics, design and live presentation. All this technology happens in a living context among people. Students who heretofore have been the passive victims of techno-gap, can learn to act and conduct themselves with respect for intellectual rights as well as improve the moral fabric of the city. Industry has never been more alert to this promising turn of events in educating that can, with your help, supercede the current fiscal crisis. Our program organizes traditional roles and seeks liaisons within the electronic industry. It promulgates a modern, active response for working through the problems of the city, and re-inspires integrity to urban living with the latest modern tools available.



Foundations have not readily grasped our program and project as suitable for their grants, while industry has quickly lent help. Interweaving education and design may not appear at first glance as appropriately educational or anything beyond the ordinary until you notice the quality portrayed in the design for Bellevue. The next advance of our work develops video and presentation formats for convalescent patients through closed circuit cable at Bellevue. Sustaining the Outreach with TFA, this Fall will begin the task of teaching the use of English through page design. The attached video and document are proven reality. We seek your support to stabilize and support the energy that has infused these results.

Bellevue & Brooklyn College Design Initiative

This project is a coordinated effort of a design group of seven Brooklyn College students with the patients relations staff of Bellevue who, over one summer, completed an innovative design format for more than fifty patient handouts. These documents serve Emergency Room, Pre-0p, Post-Op and a variety of clinical needs to discuss times, places and procedures with a diverse patient population. The original, plain, typewritten sheets are synthesized into visual-text combinations and woven into a simple, readable image – like a sign.

Creatively this initiative has four distinctive features. The designers showed a remarkable linguistic sensibility. Particularly their work acknowledged that injuries and emotional states are the palpable reality of Bellevue's daily work. This feeling was communicated in a respect for the patients' personal needs. When you are in the traffic of pain, under emotional burdens, you cannot stop to read lengthy documents.

Physical crisis demands clarity. The image of these designs focuses on that narrowed ability under duress to perceive information.

Ernesto Santos traveled from Cuba across Europe, Asia and finally after three years arrived here.
He is a regent's scholar with both a major in philosophy and fine arts.

Good design centralizes that flow of information like a river picking up tributaries – sensitive to a whole terrain. Design really visualizes cognitive psychology, following the natural and organic pattern making part of the mind.

Until he was nine years old, Ted was raised and schooled in Japan. On his father's side he has direct lineal descent from a samurai clan. His mother is American – he sits squarely between these two cultures as an art major, painter and professional musician who on weekends plays funk blues.

Providing that careful studied work is sensitive to distinct natural patterns that spring up between the information and the design. Accessibility in an articulate form is our image of information. Hierarchies must not entrap this information, but rather they serve to integrate user and usage, within clear pathways. This ordering is graphically the physical presence of the

The parents of Davis Yee both came from China. He has, at 20 for several years completely supported his education as a computer systems and graphic designer. Always busy on campus, he's also systems and graphic designer for the school's paper The Excelsior.

document – Design is visual logic. Through these strictly focused demands our basic format maintained its aesthetic ground, providing an open acceptance of adaptations from many clinical sources. Often bi-lingual, our designs flexibly

Ching Man Cheng, a Computer Science major left Hong Kong eight years ago. After finishing high school in Canada, he came to Brooklyn where between working 7 days a week at three outside jobs he pursues interests in film making, video and computer design.

covered widely divergent directives. Information is now smoothly tracked through icons marking the paths laid between image and text. The final designs are visually unified telegraphic, sign-like banners. The bi-lingual issue is particularly interesting considering the varied background of the designers. What is not known and cannot be seen is the fact that each designer has himself

Victor Buttaro came from Argentina at the age of three. He was instrumental in setting the template design for our Bellevue Format and is making the transition into computer graphic design parallel to completing a traditional art major concentration.

traveled far to be in America. In their short youthful lives each has crossed two or more cultures and a number of continents to arrive in the USA. Among them Ernesto Santos traveled from Cuba across Europe, Asia and finally,

Mordecae Zibkoff was educated through his high school years in Colombia and came back to the USA as a teenager. His dynamic enthusiasm and intuitions often leap the notorious restrictiveness of computer design. This infectuousness complements his workaholic productivity.

after three years, arrived here. Mordecae Zibkoff was born in Colombia and came to the USA as a teenager. The parents of Davis Yee came from China as did Ching Man Cheng who himself came from Hong Kong.

Ted Matsushita's family traces back across more than four hundred years, a continuous samurai tradition. Victor Buttaro came from

Ronaldo Kiel finished his M.F.A. this May and continues post-degree work guiding students in the Macintosh Lab.

His paintings compound numerical theory and visual fields.

Argentina, while Ronaldo Kiel, the graduate coordinator who is of both German and Italian extraction, arrived here for his graduate work from Brazil.

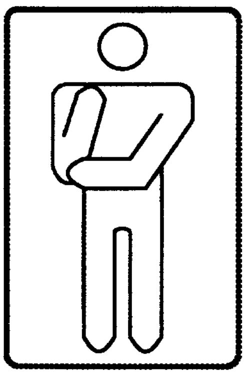
Creatively and personally, it has been a happy experience for me to see so much work completed at such a high level in the short time of one summer. This remarkable endeavor shows that the high technology tools lent to our living laboratory are not only used to help train first generation college students to bridge the technological gap, as I promised, but equally to return across that bridge and help the very human needs of the Bellevue Hospital Complex.

Prof. Richard Navin

GENERAL INSTRUCTIONS FOR HAND INJURIES

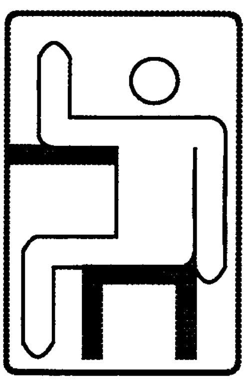
BELLEVUE HOSPITAL CENTER * OPD-HAND CLINIC * 1st FLOOR (1W) * (212) 561-6405

1



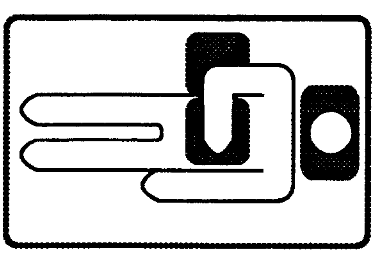
When walking, hold hand straight up, supporting elbow with opposite hand.

2



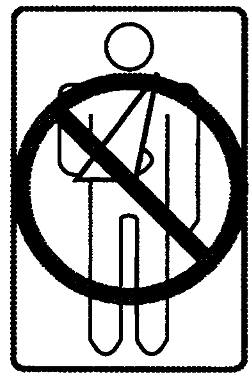
When seated rest elbow on table or arm of chair with hand straight up.

3



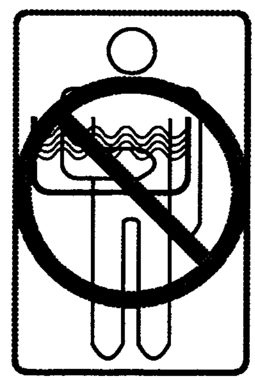
When reclining, place one pillow at your side to rest your elbow upon; and another pillow on your stomach, to rest your hand on.

4



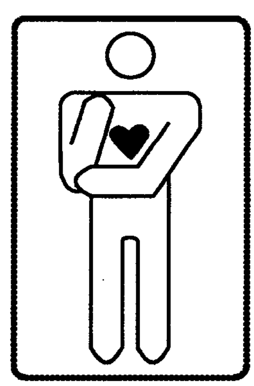
Do not wear a sling!!!

5



Do not soak unless specifically instructed by M.D.!!!

6



REMEMBER
Your hand is elevated when it is above your heart.

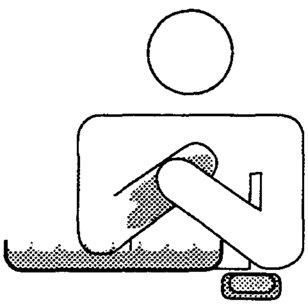
#1-APPROVED 3/91

IMAGE & COMMUNICATIONS DESIGN PROJECT-BUTTARO

INSTRUCCIONES PARA LOS PACIENTES: DESPUES DE QUITARSELE EL YESO

CENTRO HOSPITALARIO BELLEVUE * CUIDADO AMBULATORIO DE ENFERMERIA

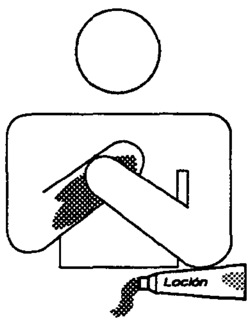
1



Lávase suavemente con jabón y agua la parte afectada. Séquese con delicadeza.

2

Untese loción o crema a la piel.

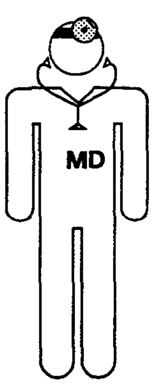


3

No se quite, escarbe, ni se rasque la piel seca. Poco a poco la piel reaparecerá más normal.

4

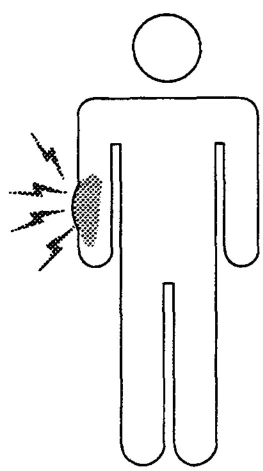
Pregúntele a su médico cuales actividades puede hacer Ud.



5

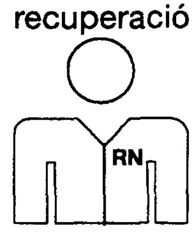
Acuda a todas las citas que le dió su doctor para ir al Departamento de Rehabilitación.

6



Puede que tenga dolor o hinchazón en la parte afectada por un tiempo. Siga las instrucciones del doctor para que le ayuden acortar el período de recuperación.

7



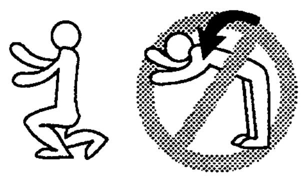
Las enfermeras siempre están dispuestas a explicarle las instrucciones y contestarle sus preguntas.
Llame a la Clínica Ortopédica Lunes a Viernes de 8:30 a ___________.
Teléfono 561-6447
o Clínica de la Mano 561-6405.

IMAGE & COMMUNICATIONS DESIGN PROJECT-SANTOS-YEE

DO'S AND DON'TS FOLLOWING NOSE SURGERY

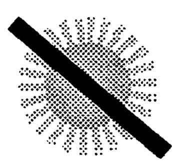
BELLEVUE HOSPITAL CENTER * AMBULATORY CARE NURSING

1 **Don't lean forward for two weeks (if you have to pick up something bend from the knees with head erect.)**



2 **Sleep on your back, with head elevated for two weeks, following the removal of the nasal splint (sleeping on one side can cause the nose to be bent on one side.)**

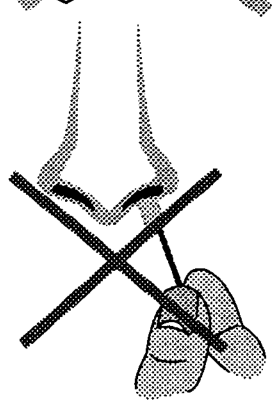
3 **Stay out of the sun for one month.**



4 **Try not to sneeze for three weeks after surgery (ask for medication against sneezing.)**

5 **Don't blow your nose for three weeks. (It's alright to blow out gently without touching your nose.)**

6 **Do not put anything up your nose, not even sterilized Q-tips.**



7 **Following the removal of the splint, wash the nose several times with Neutrogena, a soap that is obtainable in most drug stores.**

8 **Anything can be eaten, but soft foods that require less chewing will seem easier to manage.**

9 **If you feel well enough, you can cook, market and care for your personal needs.**

10 **Do not exercise or engage in strenuous physical activity.**

11 **Hair can be washed in the beauty parlor the following week (it is alright to lean head back but not forward.)**

IN CASE OF HEMORRHAGING

Put head back, apply ice packs and call your doctor.

Sometimes there is a loss of sense of smell and taste.

Don't worry, these senses will return within three months.

IMAGE & COMMUNICATIONS DESIGN PROJECT-ZIBKOFF

Computers as a Vehicle for Integrated Creativity

Joan Truckenbrod

New directions in artistic expression and communication design are multidimensional, integrating different modes of human experience. Artists are exploring the potential of interactive art forms facilitated by computer systems that are used to integrate sound, images, text and animation. Designers are expanding the traditional elements of design to include voice, music, movement and time. Thus artists and designers are creating multidimensional experiences that weave together a complex set of experiences. Artists and designers integrate music, sound or voice with drawings, typography, photographs and movement in time. The vehicle for this integrated creativity is a computer studio. Thus, the computer is an essential component of art and design education today and will be increasingly so in the future. Computer systems provide a natural vehicle for teaching students design principles, composition and color theory. Computers can be used in a spontaneous, fluid manner for exploring the visual dynamics of a composition. Increasingly, students are becoming familiar with how computers work and can use them effectively in the learning process. The significant potential of computers, however, is in the stimulation of creative thinking and in the creation of new forms of artistic expression. The computer is a multidimensional medium that provides the artist and designer with different modes of expressing and communicating ideas and information. These modes include visual imagery, text, sound, music, movement and animation. No other medium facilitates this diversity of expression and communication in the creative design process. An introductory foundation course, introducing all artists and designers to the potential of computer systems, is essential in developing the understanding and skills necessary to integrate computers into the creative process in a wide variety of disciplines.

I have developed such a course at the School of the Art Institute of Chicago, titled *Experimental Computer Imaging*, in which I use my own book as the basic text [1]. This course is available to all students at the school and is a prerequisite for other courses in the Art and Technology area. The concept of 'imaging' in this course includes visual imaging, sound imaging, interactive imaging and creative movement. The objective of the course is to integrate computers into the creative process in a multidimensional, multimedia manner, and to establish a methodology of experimentation and exploration that expands the students' creative thought processes beyond the traditional boundaries of art and design. This course creates an awareness of the potential for developing new directions in the arts and design professions. Experiences in visual imaging, sound imaging, animation, interactivity and multimedia, involving computers, stimulates creative thinking for the evolution of new forms of communication design and artistic expression.

Students begin by exploring imaging processes that are unique to the computer studio. Computer systems provide the opportunity for drawing, sketching, and painting and creating imagery using techniques of digital photography with video digitizers and scanners. Students use these devices to create various abstractions and interpretations of their subjects, incorporating the distinct visual characteristics (involving the synthesis of hand-drawn imagery and videodigitized and scanned images at varying levels of resolution or pixelation) permitted by the medium. These images can be further developed through image processing and image synthesis techniques. Complex compositions are created that cannot be created with any other medium. As a transition from visual imaging to sound imaging, students work creatively with different visual representations of text and the integration of text and image. Then the aesthetics of the text and image composition are translated into a sound landscape involving voice, environmental sound images and music compositions. Different types of sound-capture or sound-sampling are explored together with methods for sound processing and sound synthesis. There is a close analogy between visual-image processing and sound processing. The aesthetics of the visual realm are translated into the aural realm. There are interesting historical antecedents to this intimate link between sound and visual perception, as illustrated in Kandinsky's book *Sounds,* in the work of John Cage and in the simultaneous vision of sound and color of the Russian composer Alexander Scriabin [2].

In addition projects are introduced in animation that explore movement and time as compositional design elements. Next, students explore the concept of interactivity and the aesthetics of interactive artwork. Thus, students are involved in multidimensional modes of communication design and artistic expression. Students are challenged to develop their own ideas integrating different modes of expression. In fall 1990 students will be involved in an interactive compact disc project and will exhibit their artwork via telecommunication in various places throughout the world.

References

1. Joan Truckenbrod, *Creative Computer Imaging* (Englewood Cliffs, NJ: Prentice-Hall, 1988).

2. Wassily Kandinsky, *Sounds* (New Haven, CT: Yale Univ. Press, 1980); John Cage, *I–VI* (Cambridge, MA: Harvard Univ. Press, 1990); John Cage, *Empty Words* (Middletown, CT: Wesleyan Univ. Press, 1981); Bulat M. Galeyev, "The Fire of *Prometheus:* Music-Kinetic Art Experiments in the USSR", *Leonardo* 21, No. 4, 383–396 (1988).

Reprinted by kind permission 1991, **LEONARDO**,
Journal of the International Society for the Arts,
Sciences and Technology

Paradigm Shifts in Design Education

Hari Nair
Director of Academic Computing
Rhode Island School of Design
Providence, RI

Where are we now?
We are at a crossroads. computer technology has already had a profound impact on medicine, engineering, finance, management, telecommunications, physics, chemistry and other scientific disciplines. The real revolution seems to be imminent through the newfound capabilities of the computer in visual applications. Artists, designers and other visually oriented professionals were somewhat reluctant to take on the technology mainly due to the limitations of the medium. Now, things are different. Computer images are beginning to meet the expectations of these professionals in terms of resolution, quality and reproduction. A third wave in computing is likely to unleash the power of the computer technology and ultimately expand the realm of artistic experience and visual expression. The paradigm shifts will not be very different from those of the renaissance era or the industrial revolution.

Marshal McLuhan has defined three stages in the evolution of new technologies and their influence on art, which can be roughly interpreted as follows:

> Phase one involves imitation of an earlier technology or form of expression. Photography imitating 'realism' or video imitating 'film' during the early stages could be seen as examples.
>
> Phase two is the point of departure or liberation from this predecessor form. There is great awareness of the special and unique capabilities of the medium during this time. We know about photography going through this stage not too long ago.
>
> Phase three is the real revolution where an entirely new art form evolves.

As far as computer graphics is concerned, we are somewhere in phase two of this evolutionary process. Artists and designers are just discovering the virtues of this new medium. The early development of computer technology did not carry any clues about its potential impact on visualization, because the origins of computing were based on alphanumeric information processing. Even though the original computing concepts were inspired by Jacquard looms which used punched cards for programming, it was not until Ivan Sutherland's ground breaking work, the famous 'sketchpad' in the early sixties, that computer graphics became a reality and an acceptable outlet for the tremendous processing power of computers. We have seen a series of breakthroughs in semiconductor technology, integrated circuits, microprocessors, data acquisition, storage and output devices since then.

Paradigm shifts in design education
The new hybrid process is already affecting the way art and design is taught in our universities. The artists and designers are required to be familiar with more and more options available to them throughout the creative process. Familiarity with digital image creation and reproduction methods are becoming as necessary as photography or xerography. Three dimensional modeling on the computer is becoming an effective substitute for physical models at least during the early stages of design conceptualization. It may also mean fundamental changes in the way foundation level exercises are structured. For example students acquire

basic knowledge about three dimensional geometry through design exercises in paper, wire, metal and clay. They will also have to be taught some fundamental principles behind computer generated geometry dealing with wireframe modeling, surface modeling and solids modeling. Some understanding of boolean geometry will help enhance the experience and lead to a better grasp of manmade forms.

The children who grew up in the home computer era are either in high school or finishing school just about now. Art and design schools should be reviewing their curriculum structures and technological resources in order to deal with the influx of this new breed of scholars.

Planning for the future

Curricular planning and resource acquisition are closely linked. A three tiered model is recommended (see bifurcation model chart). It may be necessary to provide centralized classroom facilities to offer entry level computer graphics courses for incoming freshmen. The second tier may have decentralized resources where the individual departments are encouraged to acquire computer equipment appropriate to their disciplines. There should also be centralized advanced computing facilities primarily for use by seniors, graduate students and faculty researchers. Faculty training and retraining are going to consume more time and resources than planned. Educational institutions will be forced to take an aggressive stance in fund raising and grant proposals.

**'The Nair Labyrinth -
Traditional and Electronic media Paths!'**

The relationships and fits between the traditional and electronic media in art and design process are still being investigated. There is a certain amount of hesitation on the part of artists and designers trained in the traditional schools to accept the computer as a tool in the creative process. The other extreme view which promotes the complete replacement of the traditional process with electronic means is not very popular either. The following diagram (Nair Labyrinth) visualizes a real life evolution of the two approaches. The creative process in art and design is becoming more and more of a hybrid between the traditional and electronic processes.

The two paths, traditional and electronic processes, used to be pretty linear in the precomputer era.

Technological influences (the development and proliferation of various technologies, such as xerography, photography, video, still video, hypermedia, computer printers, scanning, digitizing, fax and computer aided manufacture) in near proximity started affecting the linearity of these processes.

In the eighties, the two curves started developing some interlapping and overlapping. The extent of these interlaps varied depending on the fields or disciplines under consideration. For example graphic design has followed the developments in printing technology and desktop publishing more closely than say, illustration.

Now, it is possible for a designer or artist to follow a path of choice combining the benefits of both processes during the creative cycle.

We will soon start seeing a blending of these curves in the future making the demarcations fuzzier than ever.

Hypertext and Hypermedia: Computer Interactive Instruction in Music

William L. Berz
Rutgers—The State University of New Jersey
Music Department
PO Box 270
New Brunswick, NJ 08903-0270

With the increasing presence of computers in schools, numerous authors have outlined the importance of this technology in music teaching. However most music teachers and authors have stressed the use of technology in traditional ways, either computer-based instruction (CBI) or in the utilization of application programs (sound manipulation or notation programs). Recently, several CBI programs in music have been developed that do not fit the traditional mold; at least in part, they allow the learner greater freedom of navigation through the lessons as well as the ability to mix instructional media: the computer and the audio compact disc or videodisc. This mixing of media and non-traditional linking of information holds tremendous potential for teaching, especially in music and the other fine arts.

Overview of Hvpertext and Hypermedia

Hypertext is an approach to information management where data is connected by a series of links that allows nonlinear or nonsequential organization of text. Conklin (I987) states that "the concept of hypertext is quite simple: windows on the screen are associated with objects in a database, and links are provided between these objects (p. 17). "Hypermedia is simply an extension of hypertext that incorporates other media in addition to text" (Yankleovich, Haan, Meyrowitz, & Drucker, 1988, p. 81). With a hypermedia system, authors can create a hypertext-type system that links text, sound, images, and video. Jonassen (1988) states that "the ultimate hypermedium is probably compact disc-interactive (CDI) Hypertext and Hypermedia or digital video interactive (DVI)" (p. 14). This ability to link sound and text is of particular interest and importance in music for several reasons including both the practical—the need for large storage media for music data—and the ideological—the need for a combined medium of text and sound (music) as part of an instructional process.

Vannevar Bush, President Roosevelt's Science Advisor, is credited with first describing the concept of hypertext in his 1945 article "As We May Think." Bush (1945) describes the system as:

> a device in which an individual stores his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory (p. 101).

This idea was carried further by Douglas C. Englebart and Ted Nelson some twenty years later. Englebart adapted Bush's ideas to the computer in the 1960's. In 1968, he demonstrated his Augment system at the Fall Joint Computer Conference in which he worked collaboratively on a hypertext document with a colleague some 500 miles away (Smith & Weiss, 1988). During this same basic period, the theory of hypertext was refined and expanded by Nelson who actually coined the words hypertext and hypermedia (Locatis, Letoumeau & Banvard, 1989).

There is still some confusion over the use of the terms multimedia and hypermedia. The term hypermedia seems more appropriate when multimedia applications can be accessed in a similar manner as hypertext—to move interactively between different media; multimedia is a more general term that would be used to describe any combination of mixed-media. Hypermedia then, seems to be the preferred term for most educational applications with interactive mixed media.

Hvpertext Structure

A hypertext system is a nonlinear organization of text. A hypertext system is comprised of a body of information that is linked together in some type of relational manner; it more closely models the structure of human mental processing than would conventional, linear presentation of ideas, as found in books. In most computer environments, and particularly in HyperCard-based applications, the information is presented in a series of what can be seen on the monitor at one time, roughly the equivalent of a 3x5 card. This pocket of information is a node. In hypermedia systems, this node can contain information other than text including sound or video. The system then supports a series of links that interconnect the individual nodes. There are a number of ways in which the user can navigate through the system, depending on the system organization and structure. The normal navigational procedure is to follow the links to examine each node, being able to move to certain key designated areas if so desired.

Commercial Hypermedia Applications in Music

In the past few years, several hypermedia applications have been developed in music. The following commercial applications are available at the time of this writing: "Beethoven 9th" and "The Rite of Spring" (authored by Rodney Winter, produced by Voyager) and "Beethoven String Quartet No. 14", "Brahms, German Requiem" and "Mozart, The Magic Flute" (produced by Warner New Media); reportedly, an additional application by Voyager will soon be Hypertext and Hypermedia released. These programs combine CD-audio with instructional, HyperCard stacks. All are very large (over 1000 cards) with fully interactive sound and possibilities for non-linear navigation. The Warner stacks require large amounts of hard disk space (approximately 6 M and up); the Voyager stacks require less (1-2 M).

Winter's "Beethoven 9th" was the first of this new species; a closer examination details the strength and weaknesses of the present state of hypermedia in music. This program pairs an audio-CD recording with two HyperCard stacks to study both the music itself and general European social and cultural history of the period. The stacks are divided into five chapters: A Pocket Guide (a CD controller based on the musical form of the piece), Beethoven's World (a tutorial on Beethoven), The Art of Listening (a music appreciation tutorial), A Close Reading (a step-by-step listening guide to the symphony with exploratory cards coordinated with the audio presentation), and The Ninth Game. Supplemental information can be accessed through a pull-up glossary and hypertext links.

The content of the program is absolutely outstanding. The use of graphics is restrained and very much in keeping with the overall tone of the program. The actual presentation of material does present some problems however. Although the program is written in HyperCard, the interface is surprisingly not HyperCard or Macintosh-like. For example all menus "pull-up" rather than "pull-down". Dots rather the normal method of bold-faced or italicized type show the hypertext links. There is much reliance on textual menus rather than on icons or pictures. Apple (1989) suggests that instead of textual interface, stacks should include "visual indicators of user progress" (p. 48); sliders might be a good indicator, such as those shown in Hypertext and Hypermedia the Apple Handbook, Figure 2-8 (p. 31). This treatment would be more in line with the general Macintosh interface that relies on visual rather than verbal cues.

Macintosh sound is used to produce many of the smaller small sound examples; even with the additional sound resources in the program, this can he a serious limitation in this as well as most Macintosh applications that do not employ MIDI. The limitations of the Macintosh sound are particularly striking when compared to a compact disk.

The Beethoven program, especially as a first effort in the field, is absolutely brilliant. Even with its imperfections (ie. interface, navigation), it serves as a fine model for future software development. The linking of tutorial screens with CD-sound is an excellent practical example of hypermedia.

Original Applications: Design Considerations

With the advent of object-oriented programming environments, such as Apple's HyperCard, hypermedia development is made relatively easy. Additionally, special programming tools are available. Such additions include CD-controllers (Compact Disc Lesson Creator by Charles Boody, distributed by AABACA; The Voyager CD AudioStack, The Voyager Company) and MIDI Controllers (MIDI Play, Opcode; HyperMIDI, EarLevel Engineering).

Through these types of controllers virtually any audio-CD can become part of a hypermedia application. Any videodisc can be accessed by using similar devices. Of particular interest to musicians is the University of Delaware Videodisc Music Series, a set of four music videodiscs containing full-motion video and slide banks Hypertext and Hypermedia (available through the University of Delaware, Instructional Technology Center).

A major concern of stack developers is that of interface design, which can be broken into four separate issues: access points, the creation of the user interface, search strategy features, and flexibility versus complexity.

First, access points must be defined. In hypertext documents very word potentially could be an access point. In hypermedia documents, access points can contain other elements such as sound, video, and graphics, besides text. Marchionini and Shneiderman (1988) point out that:

> links at every word to every word are clearly not desirable from the perspective of user or system performance. The trade-offs in machine overhead and user cognitive load' (in the form of overchoice) must be weighed carefully (p. 78).

Therefore key factors in system design should consider what access points should be defined, how fine the access points should be, how the points should be linked, and what links should be visible to the user. Most current models allow browsing and access to certain keywords.

Conklin (1987) points out that navigation concerns can be a significant problems in hypertext. Users face two problems: they must know where they are within the total environment, and they must know how to get to another place in the structure that they know to exist.

Also, Conklin (1987) identifies a problem labeled cognitive overhead. The user must expend cognitive energy to make the many Hypertext and Hypermedia navigational decisions that arise. These decisions may actually hinder memory of the material that is intended to be learned because of the interference in encoding. Additionally, when a student is working on a tangent area, he may forget the primary material; again, the encoding process is confused, this time with additional, possibly irrelevant information. This issue is further compounded in hypermedia applications where additional stimulation is added. Students can drown in excess information and stimulation.

The second broad category is general user interface. A positive interface enables users to access information by providing clear directions and feedback. After a selection is made, the user must receive appropriate and understandable feedback in order to maintain meaningful interaction.

The third category deals with search strategy features. Each user may employ a different research strategy. New users may simply browse, moving form one card to the next, while others may jump from area to area. There does seem to be evidence that new users do use the system differently than experienced users.

The fourth category is somewhat related to the third, that of flexibility versus complexity. As features are added, complexity is increased thereby adding to cognitive load. Solutions to these problems include using familiar metaphors (cards), providing multiple levels of application with user preference, providing examples, providing an interface similar to other applications (ie. Macintosh stacks follow Macintosh interface guidelines), providing navigational information, and providing adequate help. The addition Hypertext and Hypermedia of more complex media links, such as the addition of MIDI sound, would add strength to the program but probably would add to the learner curve.

At Rutgers, a pilot project is being conducted at the time of this writing to study the feasibility of using hypermedia applications in music appreciation classes for non-majors. With the goal of teaching identification of orchestral instruments, a HyperCard stack of approximately 300 cards was developed to accompany a disk from the University of Delaware Videodisc Music Series. Students can see and hear each instrument separately via the videodisc; supporting information is provided on the computer screen. Students have considerable navigational freedom by using hypertext links and through buttons controlling the videodisc and Macintosh sounds. Although the study is just in process, student reaction has been favorable toward the hypermedia approach.

Conclusion

Hypermedia instruction is of tremendous value for education in the fine arts because of apparent limitations of current methods. The ability to control visual images and sound, and link these stimuli with text and graphics is of significant interest to art educators. Unfortunately because of the newness of the field, there is only a very limited body of research studying the actual use and effectiveness of hypermedia instruction; only some basic feasibility studies have been performed at this time. Art educators must study the approach to see how it best can be employed.

References

Apple Computer, Inc. (1989). HyperCard Stack Design Guidelines. New York: Addison-Wesley.

Bush, V. (1945, July). As we may think. Atlantic Monthly, 176(I), 101-108.

Carlson, P. A. (1988). Hypertext: A way of incorporating user feedback into online documentation. In E. Barrett (Ed.), —Text ConText, and HyperText: Writing with and for the Computer (pp. 93-110), Cambridge, MA: The MIT Press.

Conklin, J. (1987). Hypertext: An introduction and survey. Computer, 20(9), 17-41.

Jonassen, D. H. (1988). Designing structured hypertext and structuring access to hypertext. Educational Technology, 28(ll), 13-16.

Locatis, C., Letoumeau, G., & Banvard, R. (1989). Hypermedia and instruction. Educational Technology Research and Development, 37(4), 65-77.

Marchionini, G., & Shneiderman, B. (1988). Finding facts vs. browsing knowledge in hypertext systems. Computer, 21, 70- 80.

Yanklelovich, N., Haan, B. J., Meyrowitz, N. K., & Drucker, S. M. (1988). Intermedia: The concept and the construction of a seamless information environment. Computer, 21(l), 81-96.

Creating A Computer Animation Training Facility

E. Kenneth Hoffman
Seton Hall University
Department of Communication
South Orange, N.J. 07960

ABSTRACT

This paper begins by discussing equipment and hardware requirements for 3-D desktop animation facilities. It then describes the options found in 3-D animation systems and describes the steps in creating 3-D animations. It includes a description of student projects in a first semester animation class, and concludes with a discussion of costs related to setting up an animation training facility.

INTRODUCTION

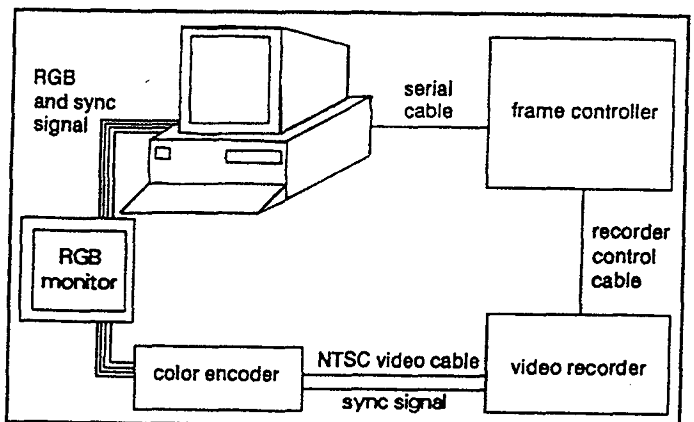
Dramatic increases in computer performance and equally dramatic decreases in computer price have brought high quality animation to the desktop computer. Todays top of the line desktop computers run approximately thirty times faster than models introduced less than ten years ago and have ten times the random access memory and hard disk storage capacity. They cost about the same or less. The price of a frame controller may still equal the combined price of the computer and peripherals, but may not be needed because its function can now be duplicated in software running on the host computer. Broadcast quality video recording equipment is expensive but broadcast quality may not be needed, and service bureaus provide reasonably priced recording services for special projects that require better output.

When 3-D animation software for the PC was first marketed in the early 1980's it came with a hefty price tag. It wasn't unreasonable to expect a $20,000 price tag for what was then the top-of-the-line software. Today, the selection is considerably broader, and the price ranges from a few thousand dollars to a little over ten thousand dollars. Needless to say, software capability has increased to the point where we expect (and can find) near-workstation performance and features in software for desktop installations. Only a few years ago photo-realistic ray-tracing and 3-D rendering with attached shadows could only be found in the most advanced software running on the most advanced equipment. These features are now taken for granted in top-of-the-line desktop animation software.

EQUIPMENT

Most animation software will run on minimally equipped hardware systems, but since the cost of computer components is now so reasonable it is assumed that the buyer will purchase a system which will optimize the performance of today's animation software. Because 3-D animation is computationally intensive it is best to purchase a computer with the fastest CPU and maximum RAM available. An MS-DOS computer should have at least a 386-level CPU (25Mhz) and additional co-processor chip. The system should have 8 to 16 megabytes of RAM, and include an 80MB or greater hard drive. Most software requires two monitors--one for the menu and the other for graphics. The menu monitor can be a standard VGA compatible RGB monitor, or multi-sync monitor. VGA graphics adaptors can now display 32,768 colors at standard VGA resolution of 640 X 400 or 800 X 600. The additional color capability may be important if, in addition to stand-alone animation programs, you plan to run paint and desktop publishing programs under Windows 3.0 or later. The graphics monitor should be compatible with the graphics card. Most MS-DOS animation software will run on a TARGA or ATVista compatible card. What is applicable to MS-DOS systems is generally applicable to Macintosh computers. Buy a system that will accept expansion cards such as the Macintosh II, with the fastest CPU (68030 or 68040), and maximum RAM.

Additional hardware will include a mouse or digitizing tablet, videotape recorder, frame controller, and possibly a color encoder and sync generator. Inventory existing equipment. Your facility may have an AV quality video recorder that will work in single frame mode. The color encoder is a device that takes separate RGB signals from the computer graphics card and a sync pulse from an external sync generator and encodes them into one NTSC standard video signal. Many graphics cards output NTSC compatible signals, however, so the color encoder and sync generator may not be necessary (though picture quality is not as good when using graphics card NTSC output).



Equipment configuration for animation workstation.

The frame controller works as an edit controller allowing the VTR to perform single frame edits. It is connected to the computer via a serial port and starts and stops the recorder. When a frame of video is ready for recording an image-ready control signal is sent via a serial port connection to the frame controller. The controller starts the videotape recorder, which records one frame (or one video field) of animation. The recorder then sends a frame-recorded signal back to the computer, and a new frame of animation is rendered to the screen. This process is repeated 30 times for every second of animation recorded on videotape.

Single frame video recording is the weakest link in the animation process. Even with state-of-the-art video equipment this crucial frame-by-frame step can be nerve-racking for the graphic artist. Advances in solid state memory technology will make this operation more tolerable even for small operations. As the cost of solid-state memory decreases it will be possible to store a second or more (30 frames plus) of animation in a separate frame buffer for the tape recorder. The contents of the buffer will then be dumped to the tape recorder as a single block of animation similar to the Digital Video Effects (DVE) operation. The videotape recorder will be used only once for every second of animation instead of thirty times per second of animation, saving wear and tear on the tape recorder's mechanical parts. Currently, large animation buffers of this type are expensive and only found at animation facilities that can justify their considerable expense. The buffers must hold over fifteen megabytes of information, roughly 500 KB per frame. They must also be able to read a full frame of information to the graphics screen in a fraction of a second. In the future such devices will become more affordable and may eventually eliminate the need for a frame controller. Digital recording, and playback of compressed image files may also eventually eliminate the need for frame-by-frame videotape recording.

PURCHASING ANIMATION SOFTWARE

Learn as much as possible about the software available. Read reviews in trade magazines. Go to regional and national graphics conventions. Look for featured "shoot-outs" (product comparisons) among competing software packages. Talk to as many vendors as possible. Take a test run on the software, applying it to your particular graphics need or problem. Above all, don't take anything for granted. Be skeptical of "gee wiz" features and impressive output. If you haven't done it yourself, don't believe it even when features are shown to you. Ease of use and the ability to move quickly between menus is more important than fancy options that may get little use and are difficult to use. Go to "boutique" shops--value-added retailers [VAR] that configure graphics stations specifically for your needs. Talk to graphics systems consultants. Find out what they recommend. You may want to pay their prices if continued support is an important consideration.

3-D ANIMATION PROGRAMS--FEATURES AND PROCEDURES

Model Construction

Three-dimensional model construction, sometimes called object definition, can be done in several ways: Models can be created interactively from 2-D shapes with options found in 3-D creation menus (extrusion, revolving, and cross-sectional modeling); they can be created procedurally (Bezier patches and fractal generators); or they can be created from primitive graphic shapes using solids modeling.

The artist may begin the modeling process by tracing a sketch of the object into a 3-D modeling program. The sketch (possibly from engineering drawings) is placed on a digitizing tablet and traced into the database by noting each end point or line intersection. After the shape has been digitized it can be further modified with a shape editor. Shape editors modify existing shapes by selectively changing their dimensions. They also contain Bezier tools that smooth out curves drawn by hand or fit a smooth mathematically defined curve between two end points and two or more control points not on the line. Bezier tools can quickly change a straight line into a uniform "S" curve.

When you have finished editing the 2-D shape, it can be three-dimensionalized by one of several methods. Extrusion creates block shapes like 3-D letters and cylinders. Revolving is used for creating symmetrical shapes that appear to have been turned on a lathe, like globes and goblets. Using this method, a 2-D shape is swept around an axis into Z space to create the 3-D object. Cross-sectional modeling can be used in place of the above methods but it is most often used for creating asymmetrical shapes -- a bottle with a square footprint (four flat sides) and a round cap on top, for example. Asymmetrical shapes can also be created interactively from existing objects by pulling and tweaking individual lines. Three dimensional models can also be defined a point at a time with 3-D digitizers, and constructed from side and front view photographs of 3-D objects with special photogammetry software.

Establishing the Motion Path

Motion path is a general term used to describe the trajectory of an object during the course of an animation. Programs vary in the methods used to create the motion path. Some programs allow you to place the object where you want it to begin its move, and you then add additional positions as interim key frames. The program then creates a curve through the key frames which becomes the motion path. Other programs use a scripting approach where you type in the opening and ending coordinates, as well as additional information regarding object rotation and sizing.

Control features found in Animation Editors. Animation editors provide control options for various types of movement: X, Y, and Z rotation; scaling (sizing); translation (placement in the world coordinate space); and perspective. In addition, the animation software controls the placement and movement of light sources. Light sources can be animated so that they move around the object.

Animation software also provides a means of controlling the velocity of an object. Beginning and ending velocity are controlled by the length of the "taper," or what is called a "faring" or "ease" -- the gradual acceleration or deceleration of an object. Tapers insure that an object will start gradually, reaching full speed and then decelerate smoothly. The artist specifies the length of the taper, which can be as short as one frame for instant acceleration, or 40 to 80 frames for a gradual acceleration. Full velocity is reached at the end of the taper.

Metamorphosis. Another feature found in animation editors is the ability to "metamorphosize" one shape into that of another. The technique was used by Peter Foldes in a series of films, the most famous being "Hunger," (1974). Foldes digitized a series of pen sketches and then used them as key frames. He metamorphosized one frame into the next. A belly dancer in the arms of a grotesque man transforms into a pillar of pots, which crumbles to the floor from under his arms. He kicks the pots and they become the walls of a kitchen. The film won the "Prix du Jury" at Cannes in 1974 and was nominated for an Academy award.

Hierarchal Control. To accurately control the movement of complex models like the segmented body of a cartoon character, hierarchies of dependent objects are created, so that when you move the upper arm of the character, the lower arm and hand move as well. This kind of control makes creating realistic movement much easier for animators because they do not have to worry about hand movement until it is time to move the hand independently of the arm. A more complex model may even have independently moving fingers. The fingers will automatically turn as the hand turns, but they will move separately when needed.

Testing the Animation--Motion Preview Pencil Tests. Quality of movement is checked occasionally by running pencil tests of the animation. The term is a hold-over from traditional cell animation, where pages of preliminary pencil outlines were flipped back and forth to check the smoothness and character of the movement. A computer graphics pencil test serves the same purpose. A small section of the animation (one or two sequences depending on the amount of computer memory) is computed in wireframe mode and saved in the frame buffer at low resolution (one quarter or less normal screen resolution). If the animation contains many complex models, bounding boxes may be substituted for the actual models in the animation. The frames of animation are then displayed on the screen.

Wireframe Tests. After a sequence or short animation is completed, a full wireframe test is recorded on videotape or film. This test runs longer than a pencil test and usually is in higher resolution. Besides checking spatial movement it verifies pace and rhythm.

Adding Detail -- Sets, Costumes, and Surfaces

After the wireframe test is approved the animator focuses on set details--chairs, lamps, potted palms, and so on; and surface detail--color, shading, texture, reflectance and transparency. Background props and set decoration not associated directly with moving objects are added to the environmental space. Walls, windows, wallpaper design, tables, and patterns of light on the floor are added. Texture maps are created for objects that need more surface detail than simple flat or smooth surface shading can provide. In short, the scene is decorated and objects are given surface attributes that will be used in the final version of the animation.

Lighting

As in a traditional filmmaking or theatrical production, lighting is an essential component. Most programs give the animator at least ten hypothetical light sources to work with. Individual objects like table lamps, can be highlighted without effecting the rest of the scene (local or object specific lighting), and sets can be lit for overall illumination similar to light falling from the sun (infinite lighting). Besides fixed position lighting, lights can be made to move around the object if necessary.

Render Tests

Render tests are conducted at various stages in the production to check surface highlights, texture maps, lighting and special effects. These tests are usually conducted at less than full spacial and temporal resolution to save time and money. Instead of rendering every frame, only every other frame is rendered. Overall timing is maintained by printing every frame twice. The animation is the same length but lacks image detail and nuance in rapid movements. In addition, individual still frames can be output to a film recorder at full resolution if a better detail check is needed.

The final step in the 3-D animation process is the recording of individual frames of animation on videotape or laser disk. Computer generated animation images are recorded out of real time, frame-by-frame, in a fashion similar to time lapse photography. As each new frame is generated, the software sends a "record" signal to the frame recorder, which then starts the video recording process. After the frame is recorded, the tape recorder sends a "recording completed" signal to the animation software, and another frame of animation is rendered. The recording process can be lengthy. It is not uncommon for a single frame of animation to take five to ten minutes to render. At thirty frames per second, a 30 second animation may take several days to record. I have found that most student animation projects take two to twenty-four hours to record.

STUDENT ANIMATION PROJECTS

Before taking the course in computer animation students take an introductory course in paint and 3-D modeling which introduces them to basic computer graphic terminology, concepts and procedures. The animation course consists of a series of graded exercises that familiarize students with basic commands found in animation software, and techniques used in tradition animation. In the first assignment they are asked to extrude their initials with the 3-D software and then import them into the animation program. Using options such as X, Y, Z rotation; scaling; and translations on the X,Y and Z axis; they assign a motion path to their initials. The project is recorded on videotape and played back in real time so that they can see the relationship between the number of frames specified for a given movement and the actual speed of the animation. Students are then asked animate a simple sphere so that it appears to squash and stretch as is it bounced on the ground.

The next project reinforces the students' understanding of software parameter for motion path manipulation, and introduces them to the concept of changing one shape into another using the metamorphosis option. Students are asked to create a logo and animation for the annual festival of student film, photography, and video projects. This assignment introduces students to the concept of designing a logo with eventual movement of that object in mind. Segmentation is built into the object so that portions of the model can perform independent manipulations when it is animated. The best of these animated logos are then edited together and are used to introduce the film and video portion of the festival.

The third and final project is a 30 to 60 second narrative that incorporates several three-dimensional shapes, texture maps and/or reflection maps, and independently articulated movement. It is meant to demonstrate the students proficiency with the software and with animation techniques, and becomes the centerpiece of their animation portfolio.

DESKTOP 3-D ANIMATION COSTS

Cost will vary, but you can figure that total costs for an initial animation system (3-D animation software, desktop computer, graphics boards and peripherals, frame controller, encoder and sync generator) will be about $20,000. Add another $2,500 to $20,000 for the tape-record or laser disk recorder, depending on video quality that is needed. Most schools already have a 386-level or Macintosh II computer, and single-frame VHS recorder, so the costs may be somewhat less. A recent trend among animators is the use of animation recording services--regional facilities that will produce the final videotape from files created by the animation program (model files and motion-path files). Many animators find this to be a cost-effective way of creating animation, one that does not tie up resources in expensive recording equipment. Services may soon be using photo-realistic Renderman software from Pixar Corporation to render animations created in Autodesk 3-D Studio and other desktop animation packages. Of course, you will want to see preliminary wireframe tests before sending the files out to the animation service, so a VHS recorder and controller or a 16mm frame-by-frame motion picture camera may still be needed.

Portions of this paper are based on material found in *Computer Graphics Applications*, written by the author. Special thanks to Albin Wicki, video engineer at Seton Hall University, for information related to video and frame controller hook-ups.

BIBLIOGRAPHY

Magazines

"3-D in Perspective," Macworld, December 1988.
"3-D Tools Bring PC Images to Life," PC Week, July 29, 1991.
"Autodesk Animator..." PC Magazine, November 28, 1989.
"First Time Animator," True Imaging, September/October 1988.
"Mac Video, Take II," Macworld, February 1991.
"Move It!" Macworld, June 1989.
"Multimedia: Is it Real?" Macworld, April 1990
"Star Quality," Macworld, June 1989.

Books

Glassner, Andrew S. 3D Computer Graphics. Second Edition. New York: Tab Books. 1989.
Hoffman, E. Kenneth, & Jon Teeple. Computer Graphics Applications: An Introduction to Desktop Publishing & Design, Presentation Graphics, Animation. Belmont, California: Wadsworth Publishing. 1990.
Kerlow, Isaac Victor, & Judson Rosebush. Computer Graphics for Designers and Artists. New York: Van Nostrand Reinhold Company. 1986.
Laybourne, Kit. The Animation Book. New York: Crown, 1979.
Weinstock, Neal. Computer Animation. Reading, Mass.: Addison-Wesley, 1986.

1-D & 2-D STORIES: AGING AT THE SPEED OF LIGHT

Mary Beams
Northern Illinois University
De Kalb, IL 60115
(815) 753-8725
Internet EMAIL address: mary@art.niu.edu

Abstract

This paper, 1-D & 2-D STORIES: AGING AT THE SPEED OF LIGHT, involves a HyperCard collection of stories about aging as seen through three characters: a middle aged woman, her elderly father, and the woman's young daughter. My presentation discusses the aesthetic and technical process of creating these stories.

While each individual story has a beginning, middle and end, the stories link up so the viewer can follow different viewpoints of the same event. Time is linear for individual characters, but time interwoven loses its linearity. The characters' memories and interaction disturb linear time.

The challenge of fitting linear stories into an interactive format has held a fascination for many artists and writers over the years. This paper examines the problem of how to remain flexible, when your material is steadfastly linear and you are determined to be interactive.

Background

Ten years ago I participated in a project funded by the NEA and sponsored by New York University's Alternate Media Center. Directed by Martin Nizenholtz, The Electronic Art Gallery invited artists and writers to produce a work for a new medium being tested: Videotex.

My project was to build an environment attached to which were sixteen different narratives. I made a house through which the viewer could wander; stories were linked to different objects in each room. Until I became acquainted with the limitations of NAPLPS graphics, I envisioned a "hand-held camera" walking through this house, with the viewer able to stop in any room, touch any object and call up an animated story. It has taken ten years for computer graphics to reach the level where this larger vision might be possible.

In fact back in 1981 I had to settle for a doorbell outside the house which listed a menu of the rooms. Navigation from screen to screen consisted of "More," "Back" and "Main." It was not very interactive. The graphics were low-res and limited. Animation was also limited and did not occur on each screen.

My screens were known for their inventive use of textures, but what I did not know was how drawing those textures severely hindered the display time. The unexpected switch from 9600 baud display during screen creation to 300 baud in the videotex test homes made for some excruciatingly slow animation and angry viewers. The Electronic Art Gallery did not become a household word; in fact neither did Videotex.

The experience of making work which would be seen on an individual basis at the convenience of the audience was appealing. It made the audience an active co-creator in the work. I did not give up on an audience wandering at will through stories and animation, and my current project takes up that notion of stories with choices. As mentioned earlier, it is a collection of stories about aging as seen through three characters: a middle aged woman, her elderly father, and the woman's young daughter.

The stories exist on HyperCard because of its accessibility to a wide audience, the viewer's control over encountering the work, and the ability to mix text, graphics and animation in one piece.

One appeal of HyperCard is the flexibility and ease with which it can facilitate the expression of ideas. It is not difficult to use this program to forge a complicated database into a persuasive flow of information, or to use HyperCard to make an entertaining game. It is a tool in the hands of many, who use it for individual purposes.

I have a very personal purpose, to make illustrated fiction which uses a conventional linear storytelling method, but is not limited to linearity. I want to tell stories which go beyond linearity, which are personal and specific and which appeal to a larger audience.

Making 1-D into 2-D

While each individual story has a beginning, middle and end, the stories link up so the viewer can follow different viewpoints of the same event. Time is linear, 1-D, for individual characters, but time interwoven loses its linearity. The characters' memories and interaction disturb linear time. The stories are equally balanced with text and illustrations. Some animation is present, but the underlying expectation is that viewers will read the stories screen-by-screen, choosing to continue at their own pace.

It was quicker for me to write the stories than to illustrate them. A story can be written in 3 hours, but a series of pictures takes up to 18 hours to perfect. Initially, the more I tried to make interactive stories, the more linear the stories became. They refused to go anywhere but straight ahead, marching from beginning through the middle right up to the end. They did not leave much opening for connecting with each other. I originally planned to have alternative endings for each story, but I quickly abandoned that plan. "We each come to only one conclusion," they insisted.

As the body of stories grew, it became clear that while one character may be on a linear path, the characters would come and go in each other's stories. So to cross-reference them became the obvious way to interact. This made the one-directional stories become somewhat interactive. They became two-dimensional, in that while each story is linear, it bears a relation either perpendicular or parallel to other stories. Meanwhile they seem to lack the three-dimensionality of making sense when linked in any direction. Each story's insistence on linearity makes too much cross-referencing either redundant or confusing.

Creating the Stacks

- Concept

AGING AT THE SPEED OF LIGHT involves a 46 year old woman, Jane, who is the divorced mother of six year old Becky. Jane's father provides a mirror to reflect a new perspective on the insights Jane realizes from living with her daughter. Sometimes the same incident is filtered through two or more characters' perspectives. In some stories one experience will recall a similar insight gained from another experience. In each character's own way they are all concerned with growing older.

Of the three main characters, only Jane is aware that she is concerned about aging. The others simply age. Secondary characters provide insights for Jane about the aging process. Money, sex, loss and death are agents for placing aging into perspective for Jane. Jane, her father, and Becky deal with loss of such things as pets, art work, legs, and relationships. Sports cars figure in all three lives. Jane's father provides personal lessons which Jane claims slowly yet passes on to Becky in fanciful empowerment. Jane's sexuality causes her to wonder about her father's and to worry about Becky's.

This series of stories is still in progress. At this writing, there are rough drafts of approximately sixteen stories. Three of them have been illustrated, and the format for the rest has been designed. In addition, there is a stack available to which other people may contribute.

-Structure and Techniques

Because this is a work in progress many details will change prior to the work being finished. The format is a photo album. A series of snapshots form the basis for menu screens, with buttons leading to other stacks. Occasional buttons mid-stack branch from one character's version of the story to another's. Buttons are simple and lead the viewer from screen to screen or when branching, off to another stack. I have not made complicated connections among all the stories.

For example, in HOT FLASHES the woman's insights into how her father handled her catching on fire are explored. WINGS OF FIRE explores the woman's daughter's fantasies based on that same event. Now the viewer comes to branches in the story and can choose to follow a different character. Linearity becomes a loop, ending back where it began so a viewer can travel down another path.

Navigation must be kept clear, or the audience becomes easily frustrated and moves on to other pursuits, unwilling to cope with confusion. Experience from Videotex days, translated to the more powerful interactivity of HyperCard, identified a minimum navigational requirement of "Next, Previous and Home." HyperCard has built into the program even more control for the viewer. Consequently, it's easy for the viewer to quit and go do something else.

Some buttons on the AGING stacks are hidden, usually attached to a character. Viewers are told to look for these hidden buttons from which they branch to another character's version of the same event. The main screen has buttons attached to each character or object, buttons which lead to the appropriate stack. Basic navigation is as consistent as possible, located on the background and remaining in the same spot from stack to stack.

Illustrations have been made entirely from the HyperCard tools, using the mouse. I felt the quality of the mouse line was important to the illustrations. Some mouse-drawn images might never have been conceived during the easier drawing process of ink on paper, subsequently scanned into the stack.

The stories were initially written on Microsoft Word, revised and then entered into HyperCard. In the stacks I made an effort to have text from one screen replace the previous screen's text, although in some cases the text moves to a completely different location. Some of the stories have a more rigid format.

- Interactivity

From the outset I wanted these stories to be interactive. Initially I had planned to have alternative endings to the stories, and to provide sections which viewers could use as blocks to construct their own stories. While there are examples of interactive storytelling such as certain novels with multiple endings, in general the more successful interactivity comes from games, and follows an adventure or problem-solving thread to lead the viewer through many paths to one or several conclusions.

I was surprised by the insistence that these stories remain linear. What I needed to say refused to be said in the more open-ended structure of a do-your-own story. These stories had a point and demanded to get to that point their own way.

My initial intention was to examine the way in which time is affected by the interaction of artist and audience: artist gives audience certain tools, audience and artist co-create the artwork. I assumed that providing choice would affect the sense of time.

I hadn't counted on the art itself being a player, in this case insisting on being linear. It may be that in working with the material longer, the stories will loosen their grip on time. At this point I am not interested in changing the project, i.e. in developing a game instead of telling the stories, because the stories insist so strongly upon being told.

So to be able to play with interactivity, I had to come up with an alternative within the linear structure. I conceived of these stories as seeds for interactivity, and developed a method by which others may contribute to the work. By using phrases from each story, I removed them from their linearity and gave them to others to re-form.

Students were given key phrases from the stories. They were asked to use these phrases to make their own illustrations or pieces. Without first knowing what I had written, the students supplied their concept of what happened in phrases such as "At Christmas Mom went up in flames just after Becky had yelled at her for serving the cheese with nuts on it," and "for a brief moment, Mom wore wings of fire."

To keep their interpretations original I did not show them my own stories. The phrases given them were evocative, suggesting tales more mysterious than my stories actually were. The imaginative images the students produced provide evidence that artist and audience together can make interesting art.

I hope to have portions of the stories on HyperCard at the SCAN conference, and invite others to make stacks from those fragments. At least one stack in the series contains phrases from the stories, but no pictures. Participants will be invited to add their own illustrations, and to add new phrases if they wish.

Distributing

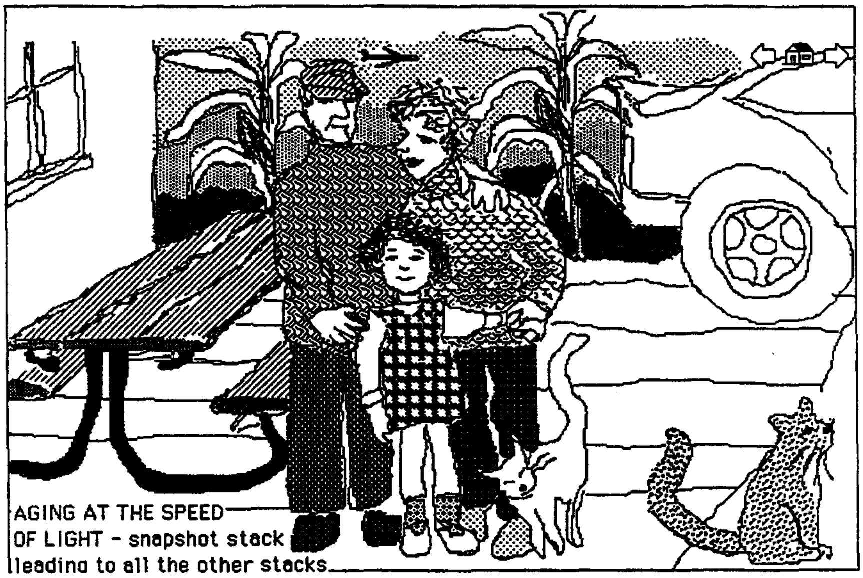
There are many options for distribution of these stories. One option is as a stack distributed to the broad HyperCard user base. Another option is to publish the stories as a comic book for the humor market, or as written but not necessarily illustrated stories. Yet another option is to create wall art from the screens.

It's important to be flexible when using the computer as a tool. Interim or final outputs do not have to be on a computer. A project can be conceived and designed on a computer but mounted on the wall or published through more traditional means.

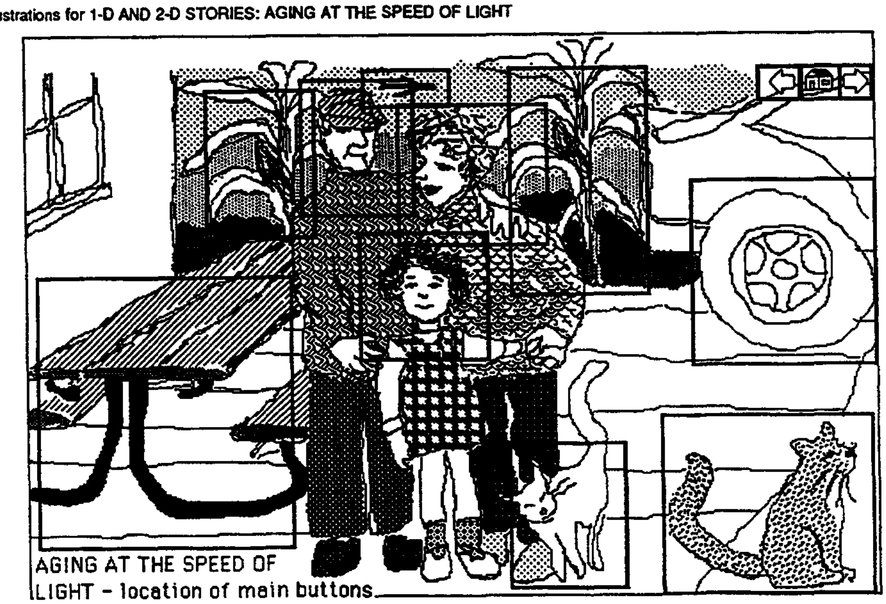
Conclusion

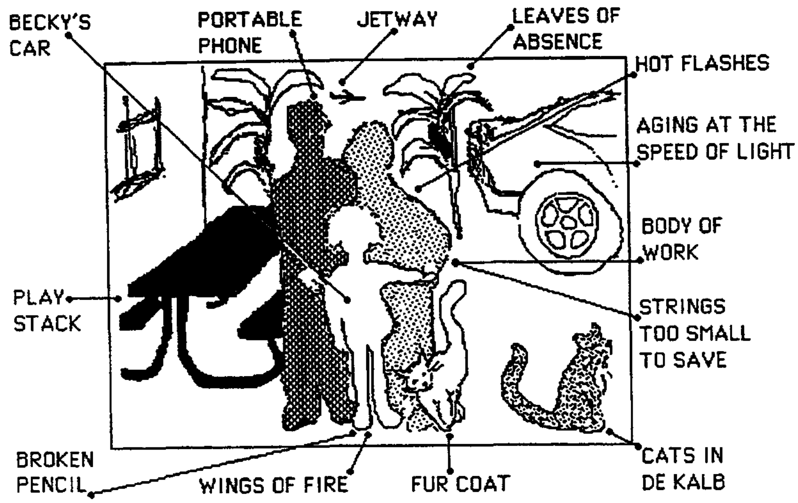
In each of those options I have used HyperCard as a design tool, mounting the ideas on an electronic easel, working on them, then dismounting the stories when it is appropriate to enhance or display in other media. I have discovered that providing interactivity is not an easy or obvious path to follow, but the chance to play with an audience and to provide elements with which an audience can play is an aspect of creating art which I want to pursue.

Illustrations for 1-D AND 2-D STORIES: AGING AT THE SPEED OF LIGHT



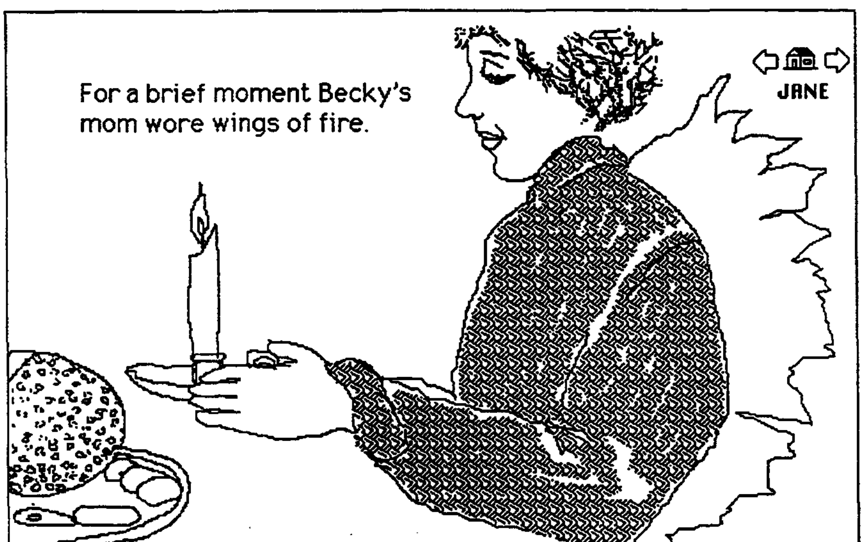
Illustrations for 1-D AND 2-D STORIES: AGING AT THE SPEED OF LIGHT





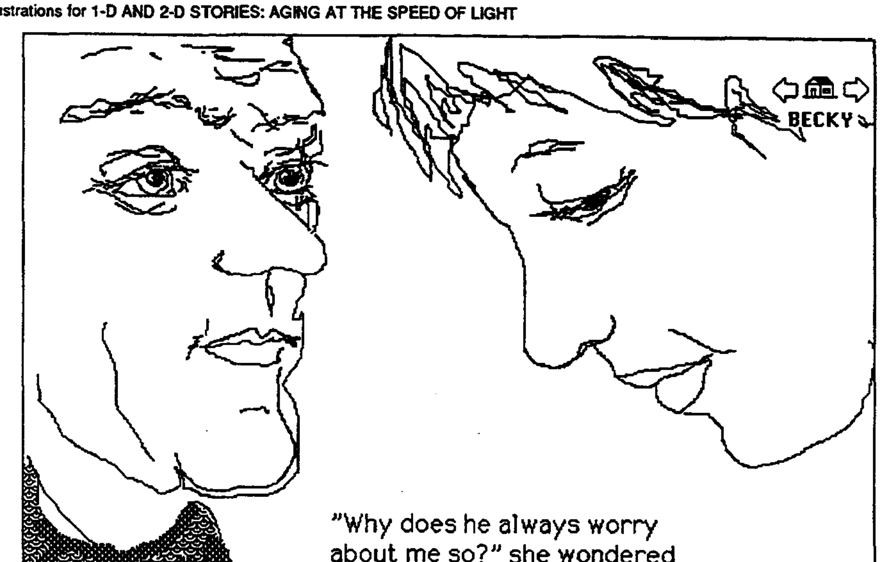
AGING AT THE SPEED OF LIGHT - diagram of stacks to which buttons lead

Illustrations for 1-D AND 2-D STORIES: AGING AT THE SPEED OF LIGHT



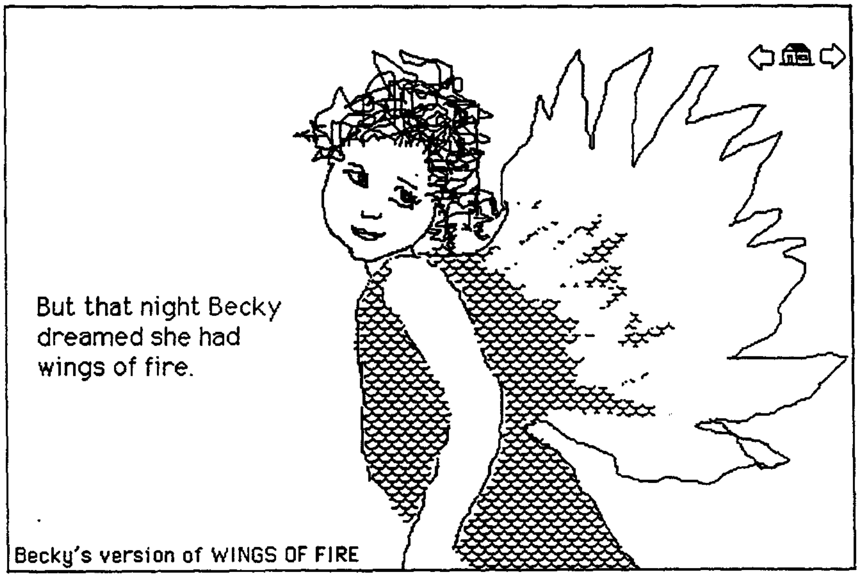
HOT FLASHES - additional button under HOME branches mid-stack to new stack

Illustrations for 1-D AND 2-D STORIES: AGING AT THE SPEED OF LIGHT

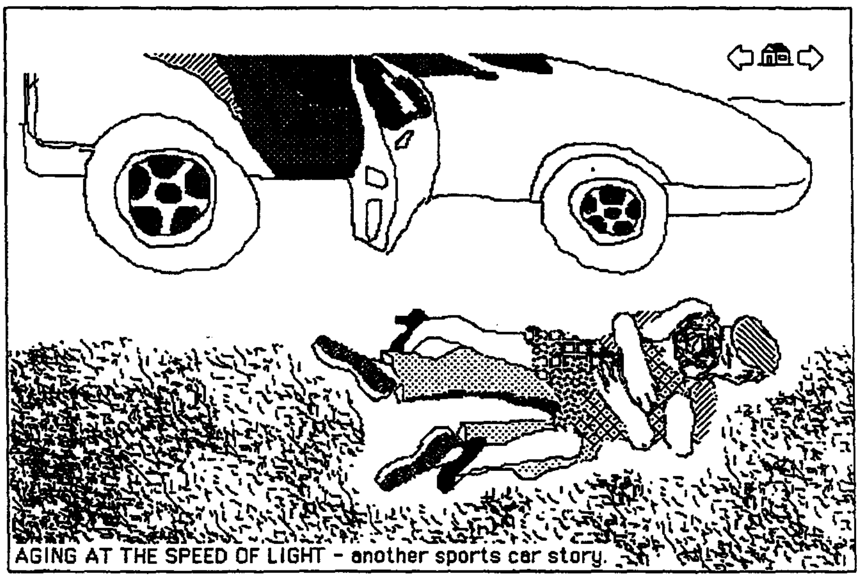


HOT FLASHES - screen in mid-stack of Jane's version leads to Becky's version

Illustrations for 1-D AND 2-D STORIES: AGING AT THE SPEED OF LIGHT



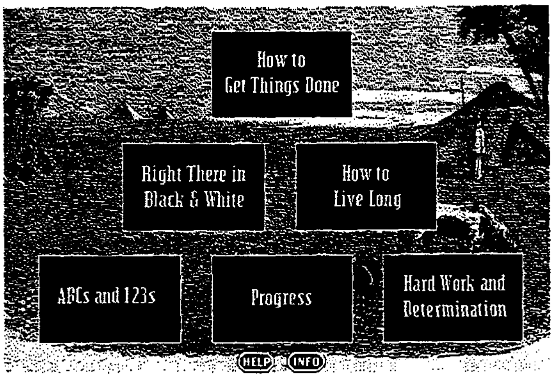
Illustrations for 1-D AND 2-D STORIES: AGING AT THE SPEED OF LIGHT

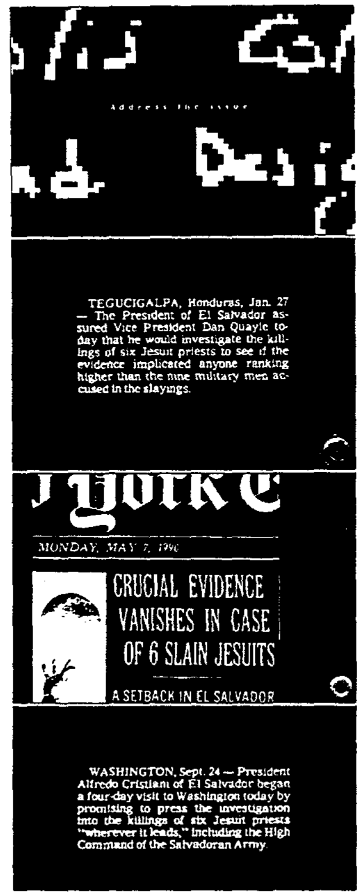


The Pyramid

a HyperCard Book

Colette Gaiter





The Pyramid is about the difference between what is believed to be true and what might actually be true. This interactive book shows how perceptions and point of view affect events and personal values. The book itself takes supposedly objective information (like news stories) and puts it in a visual context that changes meaning without changing the words.

For example, in the first stack that I developed, called "How to Get Things Done", I made an attempt at visually translating "newspeak". I wanted to see if I could develop a formula for how a news story or societal event actually develops and plays itself out in the media–with or without a conclusion.

I am using HyperCard because it is the most appropriate medium for expressing my ideas. My work is sue-oriented and issues are never neat and clean and linear–there are always peripheral events and philosophies operating on them. HyperCard is perfect for exploring these kinds of "messy" issues that defy linear presentation. There is never any one source of the problem, or any one simple solution.

The issues I look at, such as consumerism, homelessness, racism, and sexism, are dealt with constantly in other media and art forms. What I hope

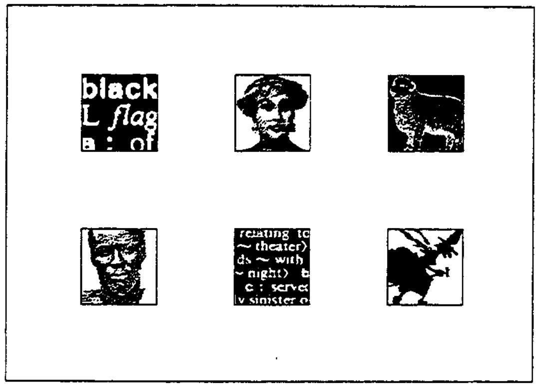
to bring to the discussion is a multi-sensory (reading, hearing, seeing) experience that may offer another hybrid perception of collective cultural information.

The Pyramid represents American societal structure–the majority on the bottom striving to work their way up to the top. I use images from how-to booklets and magazines from the 1920s through the 1950s are to illuminate a culture that cultivates insecurity, institutionalizes racism and sexism, and believes that those at the top have earned their places with superior qualities and performance.
I am interested in exploring how

hierarchies and power relationships are developed and maintained in our culture.

At the top of the pyramid is "How to Get Things Done". Presented in the style of how-to booklets, this set of "rules" interprets the process of using power and authority.

The technology used in this piece is a part of the concept. I deliberately used old, non-state-of-the-art hardware and software to comment on the value of equipment in relation to an individual's ability to communicate ideas. My stack is very accessible to viewers because it uses the lowest common denominator technology.



The rough, pixellated quality of the images reiterates the point that what the image represents is more important than what it literally looks like.

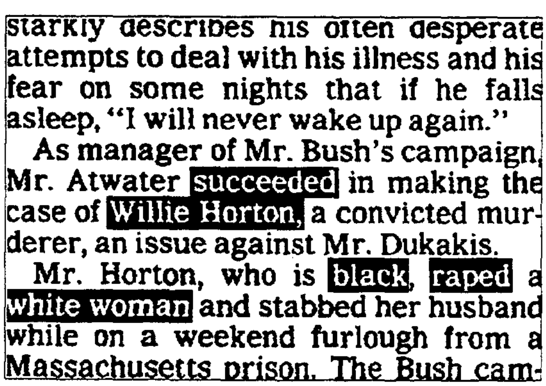
The interface is designed to be relatively transparent and intuitive. The buttons are usually located in the same place on the screen. There is some continuity throughout all the stacks in interface design.

I tried to use one of the inherent flaws of multimedia (the fact that the audience is trapped inside a set of parameters set by the creator that can't be altered) to guide the viewer subjectively through my thinking on various issues. Some choices can be made, but the point of view is always mine. In future projects, I would like to experiment with pushing the limits of true interactivity, by allowing the viewer or user to really alter the course of events and create a unique point of view out of what is available in the closed computer environment.

In this piece, I was also interested in experimenting with "reading" in a new medium. I used some video and animation techniques of making text readable only in a peripheral way by leaving it on the screen for a short period of time and making it read in relationship to what it was surrounded by. The purpose was to create an ambience of meaning, rather than a specific meaning. Images and pieces of text became a kind of visual shorthand.

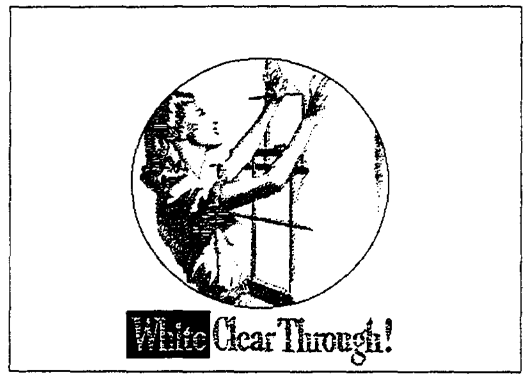
Timing and sound are also integral parts of some of the stacks. Transitions (such as dissolves) have their own connotations and meanings which had to be considered in context with the images. I became more aware than I have ever been of the sensory language we share through entertainment media.

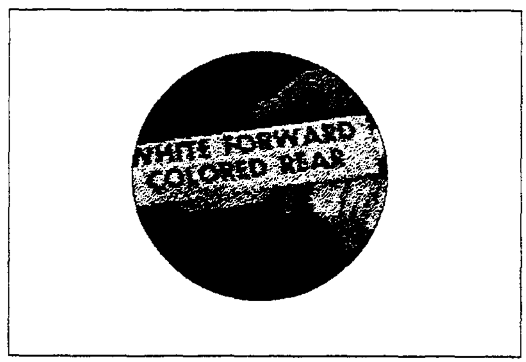
Using the metaphor of a book was the most appropriate way to develop this piece. My point was to show how subtle, ubiquitous, and complete the

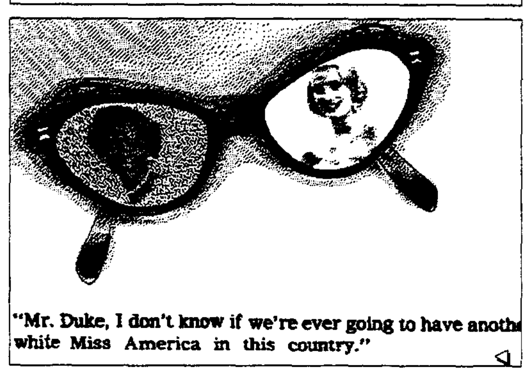

starkly describes his often desperate attempts to deal with his illness and his fear on some nights that if he falls asleep, "I will never wake up again."
As manager of Mr. Bush's campaign, Mr. Atwater succeeded in making the case of Willie Horton, a convicted murderer, an issue against Mr. Dukakis.
Mr. Horton, who is black, raped a white woman and stabbed her husband while on a weekend furlough from a Massachusetts prison. The Bush cam-

media messages we receive are. One image could not have shown this. If many images were presented in traditional linear media like video or books, the viewer would either be forced to watch or read the entire thing, or miss critical parts of it. I wanted to avoid the sensational media habit of condensing information into meaninglessness.

Effective communication of societal values and ideas is done slowly over a long period of time. What we receive as "news" is just the reporting of freak accidents in the regular business of life.







The more complete messages are not as immediately obvious.

Objects in Mirror May be Closer Than They Appear:

Possible Low Resolution Rendering of Virtual Spaces

The interactive and versimilitude of visual cyberspaces is usually measured in number of polygons generated per second--but do they have to keep *looking* like polygons? I question if extending the realm of real-time 3D solids is the best or only field of computer graphics to continue to generate virtual worlds.

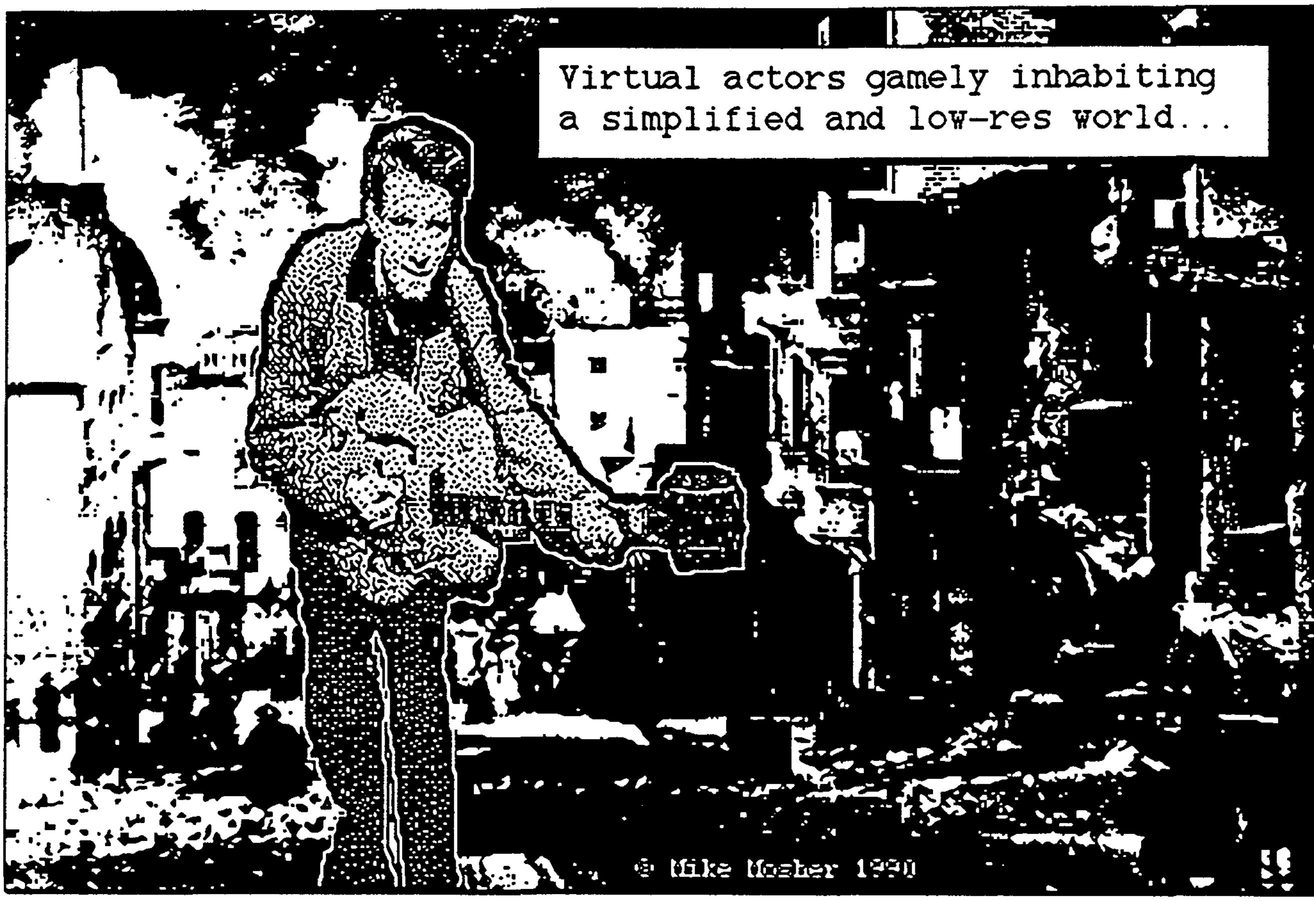
A wide variety of art historical and visual studies research can inform and inspire cyberspatial algorithms, not just those of Renaissance perspective and space. Some artists of the 1890s (Aubrey Beardsley, Eugene Carriere, Odilon Redon) created arresting, sensual black & white artworks in their own evocative and mysterious spaces while respecting the limitations of prevalent low-resolution publishing and printmaking technology. If these works inspire cyberspace designers, watch for virtual "Beardsley Spaces", worlds of demented and decorated wiry curiousities clad in flat blacks and starched whites that meld beautifully into their solid backgrounds.

As I continue to be impressed with the conception and design of the Macintosh desktop hypermedia construction kit HyperCard, I see the same visionary elegance in its outgrowth HyperScan. HyperScan, intended for rapidly slapping imagery into HyperCard, was designed by Bill Atkinson (apocryphally, over a weekend) to ship with the Apple Scanner. It offers sixteen different imaging dithers for a variety of different graphic looks to the scanned image, all in the 72 dots per inch resolution of the Mac screen. Plenty of visual variety, I suspect, for a cyberspace.

Has anyone experimented with a real-time virtual world with no more resolution than a Macintosh? Bit-mapped screen images generally have memory requirements greater than those of shaded polygons--but what about in lower screen resolutions, and in black & white? What are now perceived as limitations in sensual quality might convey "only" as much information as a very grainy black-and-white film? Some adult film artists have discovered the Fisher-Price Pixelvision 2000, a camcorder for children available from 1987 to 1989. I believe that black & white, and lower resolutions, are insufficiently pursued as avenues to convincing virtual worlds. Avenues that might be a comfortable tradeoff that boost processing speed and power.

Software engineers might create a " dark and dirty" black-and-white real-time virtual world replacing the cool flat poster-color arenas of current systems. Rather than one made of color surfaces I'd like to try entering a somewhat ambiguous space generated in ever-changing random monochromatic dithers. Not how *much* processing is possible, but how *little* resolution and color is required for convincing depiction of spaces within which we can move? The murkiness of the peppery dither might simulate the shifting haziness that the human eye experiences in low-light conditions. The result--like all cyberspaces--would be highly stylized, but not necessarily "unreal" or unpleasant.

mikemosh@well.sf.ca.us


Virtual actors gamely inhabiting
a simplified and low-res world...
© Mike Mosher 1990


...grainy personalities glimpsed
yet individualized with the
visual pepper of the night.

MIKE MOSHER
2561 Mardell Way
Mountain View, CA 94043
415-961-4104

email: mikemosh@well.sf.ca.us

Hucklefine Defined

5/5/91

I have produced a "HyperNovella", executed in Apple's HyperCard for the Macintosh computer, as a contribution to the exploration of desktop hypermedia as an artist-generated ("fine arts") medium. Hucklefine is an illustrated reworking of Mark Twain's 1885 *Huckleberry Finn* in what seems a proper form for a hundred and five years later.

A reading of *Huckleberry Finn* produced a few words or motifs from each chapter, a subjective aide-memoire, a funhouse-mirror Cliff's Notes. These motifs or phrases were used or updated through contemporary analogies to be developed into single-paragraph Hucklefine chapters. The prose was shaped into episodic chunks to fit a field occupying half of the Macintosh screen.

Randomness of chapter order interested me, and randomness was used in the cut-up novels by William Burroughs, Kathy Acker and Harold Nourse. Marcel Duchamp and Brian Eno have experimented with forms of it in the visual arts and music. At first I was content with a "Go Any Card" button after a fixed title card. Renowed HyperTalk hacker Mike Larkin developed a script for Hucklefine that randomly displays one of its forty-three chapters paired with a randomly chosen illustration. The order and design of each single-screen chapter is thus never the same twice. He also suggested the method of each chapter's appearance--the visual effects--be random as well. He also created a "mileage" indicator, a row of dots approaching a candle, by which the reader knows porportionately how far into the stack she is. A candle to lead you through the caverns of the story.

For Hucklefine's content I looked to *Huckleberry Finn.* Regarded by the academic canon as America's greatest novel, the American roadapple road epic tradition began with this rather strange yarn of escape from drunken fathers, faked disappearances of children, travelling confidence men, talentless tent shows, cross-dressing, clan fueds and a near-lynching. More disturbing than Poe's morbidity, *Huckleberry Finn* becomes irradiated by hero Huck's growing concious distrust of the American strangeness, "the peculiar institution" of slavery and the race relations that have followed it. I have set my Hucklefine lightly into an interracial landscape of pop musicians, an arena Black and White have both defined themselves with arresting sound and image and intermingled in American culture.

In Hucklefine I also allude to the highly choreographed novel *Hopscotch* by Nobel laureate Julio Cortazar, who claimed its 155 chapters could be read in linear order, or in the alternate one he provided. Italo Calvino described a literary form with multiple short, dense narratives emerging from multiple beginnings as an "open encyclopedia" or "hyper novel", in one of his *Six Memos for the Next Millenium* (Harvard, 1988).

Hucklefine's visual imagery sometimes relies on its own randomness for the connections, the reverberations of meaning, even before that of its relation to the text. The illustrations to Hucklefine are collages as well, each image passed before the eye of the Apple Scanner before any further manipulation. The landscape of art (and by extension, the Arts) becoming more collage-like was compared to a bulletin board in John Berger's *Ways of Seeing* (Penguin, 1974) in discussion of the effect of industrial-scale proliferation of art imagery. The Apple Scanner and cut-and-paste capabilities HyperCard further the collage aesthetic.

Not long after its introduction by Apple Computer in 1987 HyperCard was employed in the creation of interactive children's fiction by Amanda Goodenough. More ornate software works developed in HyperCard by others--like Manhole and Cosmic Osmo--showed hyperliterature to be a cousin in the lineage of the"total" multisensory work of art that went from ornate Wagnerian opera a century ago to manifest itself in this century--sometimes at great corporte cost--in the cinema. Many cinematic choices were put in the hands of the spectator in the 1980's, in electronically interactive pieces by artists Lynn Hershman, by Stephen Wilson and interactive literature by Edinburgh's Gordon Howell.

The culmination of this process in the near future could be all-encompassing virtual realities, now solving problems of space and structure before moving on to surfaces. Will they soon be brimming with a plethora of imagery like Flaubert's *Temptation of Saint Anthony*? Links and webs of association and juxtaposition forge inclusive Right Brain generalist epiphanies. Be inspired to connect the unconnected, the philosophically unconnectible, every scripted stitch in the potential fabric of meaning. Confined only to the computer, hypermedia is a tempest in a CPU teapot. Artists of the 1990's, call for the Hypermediation of Modern Life!

Sample Hucklefine Chapter

42

Caught, captive again. The just-us system in action. Sure as Jerry Lee Lewis keeps losing wives under mysterious circumstances, James keeps getting hauled up saddled by debts. So look, I just may as well confesses the entire plot and take Youth Home for a couple of years.
As the baliff is leading us in I see the front page of the Wall Street Journal lying on the bench. That leveraged buyout of his old record company by MegaCorp could very well mean that all James' debts are cancelled. When I point this out to the lawyer he brightens, bases our case on this, wins. I should be a funky boy scout.

The Organization Mouse

Gordon Page
Laiken & Page, Accountants
239 Thunder Circle
Bensalem, PA 19070

Background: 35 years' experience as a CPA in local and national accounting firms, and in my own practice; installing and consulting regarding the usage of computers in small and medium-sized enterprises. This work included design, installation and employee training. In addition, since 1975, I have been working with micros, programming as well as consulting. In regard to the Graphics Arts, I am a well-read student and learner. Hopefully from my experience in the computer business, I can give you some information not available in the trade magazines.

I am somewhat in the position of the First Lieutenant at the Army War College who has to instruct a bunch of field-grade officers (majors, lieutenants and full colonels) a particular military specialty. In addition to the field-grade officers mentioned, there were also a few newly-appointed Brigadier Generals. "There are at least 10,000 officers in the Army who know more about this subject than I." He gazed slowly about the room at the assembled magnificence, and continued, "Fortunately none of them are present today."

Computer Selection

In the Graphic Arts Context, there are only 2 computers (or families) involved:

(1) the Macintosh, and
(2) the PC and clones

At first glance, it would seem that a choice between only two machines would be easy. Unfortunately, convincing an owner of either one of the computer types that there is some merit in the other is comparable to trying to convince the Ayatollah to become a member of the Tel Aviv synagogue. Accordingly, I shall compare the two computer types on the basis of non-controversial features only.

(1) Historically, the Macintosh used the MOTOROLA 68000 series CPU chips which had 32-bit registers. This gave the Macintosh the ability to address more memory. This was essential for DTP programs. The Apple/Macintosh has been involved in game programs form the very beginning, commonly using lots of graphics. The Mac has always had the lion's share of Desktop Publishing Programs.

(2) The PC, starting with the 386 and the 486, with the 586 and 686 now under development now have adequate memory. The future 586 and 686 computers will be running at 100MHz and 250MHz respectively. Because of these improvements, and due to the large installed base of PCs (about 25 million), most of the software used in DTP is being ported to the PC. The PC, since it had not been devoted to graphics programs , developed a huge body of other business programs, not available on the Mac, and more high-level language compilers. This made it easier to write accounting and similar programs on the PC, assisted also by the large user base.

While Desktop Publishing Programs were being ported from the Mac to the PC, certain other baggage not applicable to business programs were also ported: mice, moving bars of light, pull down menus; instead of simple menu selection, icons, all in the mane of "Standard User Interface" (or "GUI", pronounced "Gooey"). Also "Windows, by Microsoft" was touted as necessary for all computer operations.

Crocodile tears were being shed by the same dealers/ manufactures and trade magazine "experts" who collectively have been able to standardize the keyboard or the printer control codes. Arcane subjects such as standard locations for the SHIFT-KEY, SHIFT-LOCK, ESCAPE and CONTROL KEY were beyond their comprehension—in spite of the fact that the typewriter keyboard has been around for some 90 years. Nor the shape of the human hand, for that matter, which has been around a little longer. Also, of course "Windows" was a requirement for everything, whether graphics-based. Character-based programs commonly place the cursor automatically, leaving the mouse redundant.

GUI or FUI: It has been argued by these public-spirited gurus that a graphic interface was more "intuitive" and much easier to learn than a character-based interface. Even if these statements were true, the conclusion would be false. And learning one interface is, naturally easier than learning two? Wrong. Virtually everyone in an office already knows how to type, including the boss, using two fingers. Graphic user interfaces are completely foreign in business use. They are difficult to learn. The GUI had to be learned, or more aptly commonly-used procedures had to be unlearned. Rather than having one commonly-understood interface, we now have an interface useable by the graphics people but completely antagonistic for accounting or general business usage.

Windows: Microsoft has been trying to foist off its Windows programs for several years on anyone who could be induced to put up the cost. It will no doubt be surprising to learn that the equivalent of windows has been around for some 50 years in good accounting programs. These programs had automatic windows, dedicated to the program requirements which required no input whatsoever to produce a lookup in another file. The user merely entered the required data, and the computer displayed the needed record data: The user entered the Customer Number, and the customer data was displayed without any mouse clicking, or any menu operations whatsoever. This procedure was not limited to customers but could be applied to any accounting operation: inventory, vendors, etc. What "Windows" did was produce a non-dedicated lookup, which could jump into any file from any program regardless of the program requirements. **Try selling that to corporate security?**

A procedure, however simple, takes more effort than no procedure at all.

Revolutionary or self-serving?: For the last seven or eight months the user has been bombarded with the cries of "GUI is good!" and "Windows are good!". These cries have been shouted in unison form all the rooftops. Not a dissenting voice is ever heard. It is no secret that computer business for stores and distributors has been slow. No doubt advertising in the computer magazines have also declined. The trade magazines, perhaps with a push from their advertisers, believe if they all shouted the same Little Lie. the Little Lie would become the Big Lie; and, eventually the Big Lie would be perceived by the public as the Truth. If a few extra hundred dollars' worth of unnecessary mice, memory and software could be sold to a small fraction of the 25-30 million PC users, the potential sales could run into the billions. Is this "Computergate?"

Whatever happened to "user-friendly"?

User Priorities: The selection of a micro depends upon your evaluation of your priorities, and how much money is involved. There is the story of the club pro, about to tee off on the first hole when the clubhouse door opened and a woman in a wedding gown accompanied by her bridesmaids came out. "No, Diane, I said **'if it rained'.** Different users will have different needs:

Multi-user requirements: even the smallest business tends to require more than one input terminal. This means a networking system, including a multi-user operating system. This system must provide for record-locking. Lacking proper record-locking about 15 minutes, , when more than one user accesses the same file during the same period. Multi-tasking, long the press agent's panacea, is worthless as a substitute for proper record locking. Multi-tasking is of negligible value anyway, and **dangerous** to use as a substitute for a true multi-user program.

Some of the useable multi-user systems are listed below:
(1) Novell
(2) Unix & Xenix—requires considerable training
(3) PC-MOS
(4) DIGITAL RESEARCH'S Multi-user system. Since Digital Research has be taken over by Novell, the future of this system may be in doubt.

Conclusion: Assuming that the computer system and system software has been selected with the overall good in mind, there are still a few areas where problems may arise:

(1) Graphic Arts Software Selection: Many of you are already well informed in this area. In addition, the leading trade magazines have well-rounded analyses of this type of software. It should not be difficult to make a selection.

(2)Accounting Programs: Accounting programs are more difficult to analyze. You will not get any help of any value from the sellers or distributors. They will tell you to get a nationally recognized, "reputable" program, usually with an 800 number. Since "reputation" in this industry is largely based on blind advertising and blind recommendations based thereon, do not place too much reliance on a dealer's word. Does he use it in his store? Never. Well, hardly ever. Trade magazine articles on business software is entirely based on advertising commitments, including those prepared by national accounting firms which were bought and paid for. These are the same national accounting firms who audited the 1100 or so S & L's and found nothing. Most published accounting program evaluations are either self-serving or incompetent or both. Representing the "iron" interests, they will usually wax enthusiastic over low-priced software, leaving more money for iron buying. To quote Voltaire:

> **"They know the price of everything and the value of nothing."**

(1) Those whose sole interest is in graphics prefer the Macintosh. In truth, they need manual cursor control and the ability to hop around among the sub-routines. This need is served by "Windows" and some type of GUI.

(2) Those whose sole interest is in accounting prefer the PC type of interface. They are only confused and greatly slowed by icons, moving bars of light controlled by the often hidden cursor control keys, and mice. Automatic cursor movement is far preferable to mouse movement. This is true even if the GUI were as intuitive, natural, and easy to learn as the proponents argue—which it is not!

Some time ago a small computer magazine printed a survey of 200 or so users. The results, of those who indicated their usage, showed unanimity with the conclusions above.

However, in most enterprises, accounting is one area which the user would like to serve. In this situation the user must choose between having only one of the computer types and perhaps not being able to serve the other area satisfactorily. The cost of machines and software, additional training and perhaps networking problems may make use of two types of computers impracticable. If only one can be chosen, the PC seems to be the most liveable solution.

Will the merger of IBM and APPLE interests affect the user's decision? Possibly. It should be noted that both of these parties are notorious advocates of proprietary incompatibility, and neither has been noted for business software development or promotion. The user should not expect much. I am sure they will come out with some incompatible system, with only minor software support. Remember IBM still sells its minis with RPG, a database management program, asking the user to implement his own needs. Macintosh, also relies on voluntary support from others. Since they will be starting from zero, and competing with an installed base of some 25 million PC's already supported, it will be an uphill battle. The present market is now led by Intel with minor competition from Motorola. It no longer is being controlled by the iron manufacturer.

Program Value: evaluations are made either on the basis of sales or advertising commitments. Many surveys have indicated that the majority of these selected programs are never in fact used. Surveys should be based not on advertising commitments and on sales. Accounting suitability and what happens after the sale are never considered. Nor does any dealer want quibbles about software, or any possible liability, therefore slow the rapid movement of iron.

Program Price: in the sixties, the glorious old days of 5 million dollar computers, the rule of thumb for software costs — are you sitting down? —was two-and-a-half times the cost of the computer. That ratio should perhaps be increased for special **custom** program suites, since programming costs have gone up, while computer costs decreased. Fortunately, programmers have written, speculatively, a large number of business programs. They relied on the large number of PC's in use to spread the cost around. Unfortunately none of the iron mongers want competition in the software market. They want cookie-cutter pre-selected software with a national name and a phone number so they do not have to waste valuable time on considerations of suitability. What happens after the customer's check is cashed is not their problem. The magazines, with a keen eye to the revenue budget, have allowed themselves to become mere prostitutes, forgetting their loudly-announced dedication to their subscribers.

It is PERFORMANCE not PRICE which should be the criterion used for any purchase decision.

SCREEN PRINTING VIA COMPUTER

Judith Oak Andraka
Art Department Chair
Prince George's Community College
301 Largo Road
Largo, Maryland 20772
301-322-0963

Abstract:

Screen Printing, a form of printing with stencils, is the oldest and newest technique in the history of printmaking. With the computer to help create the stencils, it becomes the most modern and easy to use technique of printmaking for the individual artist. This paper deals with a method of producing the stencils on a personal computer.

Background

Stencil work is the oldest type of printmaking, with experts claiming that the earliest examples come from prehistoric man on the Fiji Islands. In the guise of the modern method of screen-process printing (or Silkscreen or Serigraphy)[1] it is also the newest form of printing that we know today, with the inception of its commercial use in the 1920's.[2]

Screen-Printing or Serigraphy, as it is known in the Fine Art world, is a stencil system of printing multiple images. Using a tautly stretched fabric mesh on a frame, the artist blocks part of the mesh with various substances to create a stencil. The stencil is composed of positive pieces adhering to and held together by the fabric and of negative open spaces through which the ink or paint will pass. The screen with the stencil on it is placed over the stratum to be printed and ink is forced through the open areas of the mesh (those areas not blocked by the stencil); thus printing the color on the receiving surface.

Although called a photographic stencil, a photo-silkscreen stencil does not have to be an actual photograph. Rather, the term photo-silkscreen refers to the fact that light (photo) is used to make the stencil. However, because of the detail obtainable with this method and its fidelity to the positive, many of the images used in the photo-silkscreen method are derived from actual photographs or film positives. Positives for the stencil may also be created by drawing or in some manner blocking the passage of light through a translucent surface placed next to the photo-stencil material.

The creation of the photo-stencil is based on a positive-to-positive setup. In other words, what is positive or present on the drawing or film is what will print from the screen. The screen is produced by placing a drawing, positive transparency or other material in contact with the screen which has been coated with an ultra-violet light-sensitive solution; or in tight contact with a special screen film such as Ulano's RX200 which is also sensitive to ultraviolet light. The package is then exposed to a high-actinic (ultraviolet) light source. Wherever the film or screen is protected from the light, no reaction will occur. But where the light hits the film (through the negative/clear areas of the positive or drawing) the ultraviolet light will react with the light sensitive substance. After exposure, the screen or film is developed and rinsed with water. (Some films and coatings are self-developing and do not require a developer.) The areas that have been exposed to the light harden and remain, while those areas that were protected wash away. Remaining is a stencil that is open in the protected areas (the positive of the drawing or photograph) and is closed in the areas that were not protected (the negative or empty areas). When ink is squeegeed or forced through the screen, it is able to pass only through the open areas of the stencil, those areas that were black in the drawing or photograph. Each color in a print requires a separate stencil for that color.

As an artist/printmaker, I have experimented with many types of stencils, papers, and inks to achieve the art of my "inner eye". As a teacher of printmaking, photography and computer art, I have worked with many media, processes, ideas and images. Throughout my work, there has run a trend of what photographers call "graininess". While other photographers are trying to eliminate the traces of the omnipresent "grain", I have sought ways to increase it in my photographs and images and to translate it into the printing process. In the past, I had airbrushed glue and blockout onto the screen for the desired effect, but since many of my images require or incorporate photographic images or photographic quality images, the photo-stencil together with hand drawn positives became my primary choice for creating stencils for serigraphs. I developed a system of using very grainy film such as Kodak Recording Film 2475 pushed to the limit or Kodak Tri-X, terribly mishandled, in order to get the desired grainy appearance. The graininess of the film created a random half-tone effect unlike the regularly placed dots of a commercial half-tone screen. The images were then created from a collage of photo imagery and hand-drawn work done on mylar or directly onto the photo-positives. Many bits and pieces of positives and drawings were taped together to get the final positive image for the stencil. Although screen-printing has been traditionally opaque, my colors were (and are) very transparent and the finished prints consist of numerous overlayerings to achieve the effect of looking through the colors, creating depth within the color and the image; creating a complex interweaving of image, surfaces, and color.

Experimentation using the computer to do printmaking began by using a color ink-jet printer and collaging the images together. In this way large images have been achieved on archival printing papers, but there was always the concern of dealing with the paper's seams and with the archival or non-archival quality of the inks and adherents. This concern led to more experimentation and the development of a method of making silkscreen positives on the computer.

Making the Image

The method explained here uses an Amiga computer and software, but it is possible to make stencils by this method on any computer with paint and image processing programs.

The image for the print is created in the computer using a paint program. The best method for achieving well blended images is to use either HAM mode with 4096 colors or 24 bit-plane color to create the work. Having large numbers of colors available works very well when several images and digitized photos or slides are incorporated into the work. Images may be taken from numerous sources including still video cameras (such as the Canon Zapshot), video camcorders, VCRS, still photographs, 3d objects and drawings. The images are input to the computer using a number of different methods. Digiview, a video digitizer comprised of a Panasonic CCTV camera and the Digiview hardware and software by NewTek, is most often used to capture images from photographs and slides. The Panasonic CCTV camera is a closed circuit black and white video camera which captures the image and colors using a color wheel in three separate passes of filtered red, green and blue. Still video images from the ZapShot, a color camcorder, or paused VCR can be captured using an electronic color splitter attached to the Digiview or DCTV. Incoming live video can be captured using the FrameGrabber or one of the video boards. Once the image is digitized, it can be incorporated into the work. One of the best ways to ensure that the pictures will collage together and be palette compatible is to take them into The Art Department Pro (ADPro) or some other image processing program which changes all images to 24 bit-plane color. Within these programs the palettes can be adjusted to be mutually compatible. In addition, many other image processing functions can be performed.

When all the elements to be included in the piece are gathered, work begins in a paint program that is either HAM (Hold-and-Modify, 4096 colors) or 24 bit-plane color. Within the paint program, the elements of the picture can be brought in and converted to brushes, trimmed, sized etc. and can be combined along with hand drawing to form the image. The concept behind this method is very similar to working with photographs, hand drawing on the positives and combining them with drawn images on mylar. Among the advantages of working on the computer are that the resolution and the appearance of the various images can have the same look and feel to them, and that the artist is not dealing with photographs of different graininess and diverse textures from ink and pencil. Thus use of the computer results in exerting much less time and effort to get a more cohesive image. In addition, the proportions and aspects of photographs may be manipulated much easier in the computer than is possible in a conventional darkroom.

Once the image is executed and saved in the finished state, it is loaded into ADPro and converted to a 32 color image. (There is no way I am about to print several thousand or even 100 colors.) While the image is in ADPro, possibilities of various dithering combinations can be viewed, as well as other levels of contrast, brightness, etc. When the final choice is made, the image is saved as a 32 color IFF file. It is also converted into gray scale using the "color-to-gray" command and is saved as a 16 color gray-scale image file. Importing a picture of any palette or resolution into ADPro and changing it into gray scale there seems to create a better range and resolution of grays than any other method or software conversion to gray scale. Even digitizing or scanning it in sixteen-color gray-scale does not seem to produce as good results as converting it from 24 bit-plane color in ADPro.

With the image complete, it is necessary to convert it into stencils to use on the screen. Although this could be done in AdPro, the method here converts the stencils using the "COLOR" program from Deluxe Photo Lab by Electronic Arts. (This was the program I used to develop the method, and familiarity and laziness hinder me from converting to another program.) In "COLOR", the image can quickly be converted into four-color separation for the commercial printing process. However, I am not interested in that method, preferring instead to use overlays of transparent color to create the depth and richness I desire. With the image loaded into the program, the first step is to do a pixel count on the colors in the image and arrange the palette by value. After this is accomplished, two registers can be merged together to create one register of a color based on either an average of the two colors or a weighted amount of the two colors. Registers which have colors that are very close to each other in hue and value can be merged to reduce the number of colors. The program will do this automatically but by doing one's own choosing, the colors, amounts, and selection of mergings can be further controlled. The choice may be to change one of the two colors that are close in value/color and keep their entities rather than reduce them to a single color. The height of the bars for each register in the palette graph will also indicate the pixel amount of that particular register to the image as a whole. By clicking the mouse on any color register, all the pixels of that color register in the image will flash, enabling the artist to make other reduction choices based on the number of pixels present in a given color register. Often a color register will have only a few pixels of that register scattered throughout the picture. By identifying such colors/registers, they may be merged with another register rather than create a stencil for just a few pixels.

After the registers have been reduced, a press of the Right Amiga Key and "b" turn the image into a black and white picture (gray) based on the value of the colors in the palette. (This is not the same type of separation as when <u>ADPro</u> turns it into a gray-scale picture. Each color in the palette is turned into its own register of gray, whereas <u>AdPro</u> changes any color of a given value to one value of gray.) Following this conversion, each register is isolated to make a stencil. To accomplish this task, all the registers but one are set (locked) and that color register is reduced to black. The other registers are then cleared and moved to white, resulting in an image of black on white; the black being all occurrences of that particular color in the picture. To reduce the size of the file the image is then changed to a one bit-plane image (just black and white) and saved. (This is not necessary, but makes a much smaller file on the disk.) Each of the remaining colors in the palette is converted to black in the same way. This sounds much more complicated than it is and is accomplished within a short amount of time. In addition to making stencils from the 32-color IFF image, I also use some of the 16-color gray-scale image registers to create stencils that overlay the color-separated stencils. These stencils are not printed with gray ink, but with pale neutral tones that add to the depth of color in some areas (especially where shadow areas are required.)

The next step is to provide each of the separations with registration marks. The computer is a great help with this task. Putting those little cross-haired circles on positives by hand in the same exact position on each positive is a very difficult task, but by using the computer the marks are in perfect registration with each other. Using the <u>Deluxe Paint</u> program three registration marks are set on a page outside the image area. (This assumes that the page area is larger than the finished image area.) On the Swap screen each color separation is brought up in turn and combined with the registration marks using the "merge-in-back" command. This action causes the registration marks to be placed in the exact same place on each stencil image. Each image with its registration marks is saved for a positive stencil as a one bit-plane file .

Depending on the available equipment, there are two easy ways of getting the positive completed as a stencil.

#1. Print each image out onto paper. Any kind of printer will work since the image will be in black and white. An ink-jet printer gives a superior image but if the dot-matrix ribbon is new and dark, results will be good. These printouts are then transferred to "transparency film for plain paper copiers" using a plain black and white copier (Xerox, Kodak etc.).[3] Each of these printed transparency films becomes a film positive to be used with the stencil film or emulsion for the screen. The resulting transparency does not look dense enough, but it is. The carbon content of the toner in the copy machine seems to make it denser than it appears to the eye.

#2. If there is access to a laser printer, a step can be skipped in the printing. The laser printer is loaded with the same type transparency film used above and the stencils are printed directly to the film.
The laser printer does a much better job on solid areas than the copier does. When the copier is used to make the stencil, large areas of black often have to be touched up with a negative touchup.

On the Amiga the "POSTERS" program from Deluxe Photo Lab is recommended to print the stencils. An advantage of this program is that any size up to ten foot by ten foot can be printed in whatever page size and increments desired. For large prints the stencils can be taped together with Scotch Magic tape to create a full size positive stencil for the screen. The stencils can then be exposed to the photo-screen emulsion to create individual screens for the printing of each color.

Conclusion

The computer has developed into the ideal method of making stencils for the photographic screen process. In addition the screen printing process provides an archival Fine Art "hard copy" of computer work with minimum restriction on size.

MATERIALS and EQUIPMENT FOR COMPUTER/SERIGRAPHY

Computer	Amiga Computer
Software	***Deluxe Paint III; Deluxe Photo Lab - Colors*** -Electronic Arts
	Art Department Professional (ADPro) - ASDG
	Digipaint III; ToasterPaint - NewTek
	DCTV
Materials	Print on regular printer paper of any kind for that printer
	Copy (Xerox) onto: **C-Line Transparency Film** for plain paper copiers using dry toner #60727 ($7.50 box of 50 at Pace)
Silkscreen	Photo Sensitive Film: **Ulano RX200** -(Screen supply store)
	Developer - A/B developer for Silkscreen or Hydrogen Peroxide from drug store 1:1 with water
	Screen - whatever you use. I use 200 mesh Monofilament polyester
	Inks - **Nazdar - #5530 - Transparent Base**
	Nazdar - #6500 - Halftone Base
	Shiva - Permasol Oils for Artists

(I mix about 2/3 transparent base, 1/3 halftone base with a very small amount of artist oils for coloring. Thin as necessary with **Nazdar** Retarder Thinner.)

Paper: **Lenox 100** or other printmaking paper - (Art Supply Store)

Pencils for drawing stencils - **Stabilo #8008** and **Stabilo #8046**

Plastic for drawing on - 3 mil Frosted acetate (Art supply store)

BIBLIOGRAPHY

Auvil, Kenneth. Serigraphy - SilkScreen Techniques for the Artist. Englewood Cliffs, New Jersey: Prentice-Hall, 1965.

Biegeleisen, J.I. Screen Printing. New York: Watson-Guptill Publications, 1971.

Eichenberg, Fritz. Lithography and Silkscreen - Art and Technique. New York: Harry N. Abrams, Inc., 1978.

Fossett, R.O. Screen Printing Photographic Techniques. Cincinnati, Ohio: Signs of the Times Publishing Co., 1973.

Russ, Stephen. Practical Screen Printing. London: Studio Vista Limited. 1969.

Schwalbach, Mathilda V. & James A. Silk-screen Printing for Artists and Craftsmen. New York: Dover Publications, Inc., 1980.

Notes

1. Silkscreen, Serigraphy, Screen-Process Printing are different terms for the same process. Originally called Silkscreen because the mesh for the screen was made of silk, today's screens are made from polyester, nylon and metal as well as other materials. For this reason, commercial printers prefer the term Screen-Process Printing or Screen Printing. In 1952, the term Serigraphy was coined by Carl Zigrosser, curator of prints at the Philadelphia Museum of Fine Art to distinguish Fine Art Screen Printing from the commercial printing. The name is based on the Greek words Seri and graphos meaning silk drawing.

2. For history, see Mathilda V. Schwalbach & James A. Schwalbach, **Silkscreen Printmaking for Artists and Craftsmen,** Dover Publications (New York, 1980); Fritz Eichenberg, **Lithography and Silkscreen,** Harry Abrams, Inc. (New York, 1978).

3. C-Line film (a brand of transparency film) can be purchased at a discount store for about $8.00 for 50 sheets.

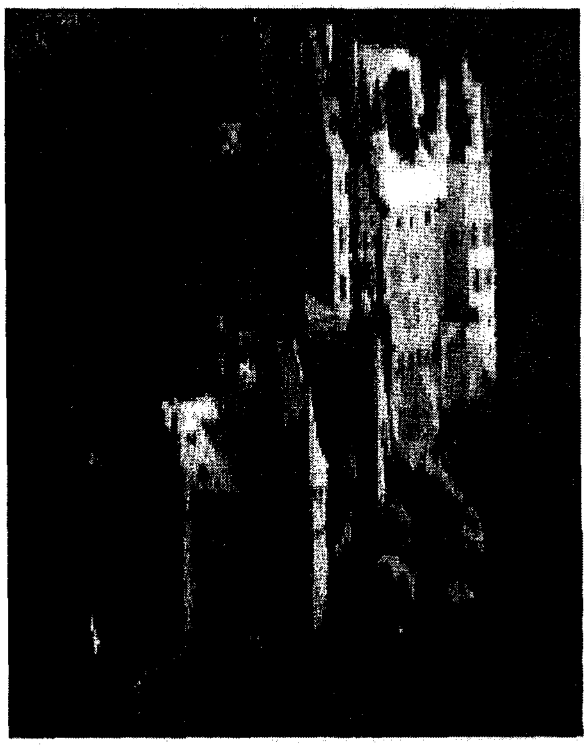


Fig.1. 16-color gray-scale printout of the finished computer picture.

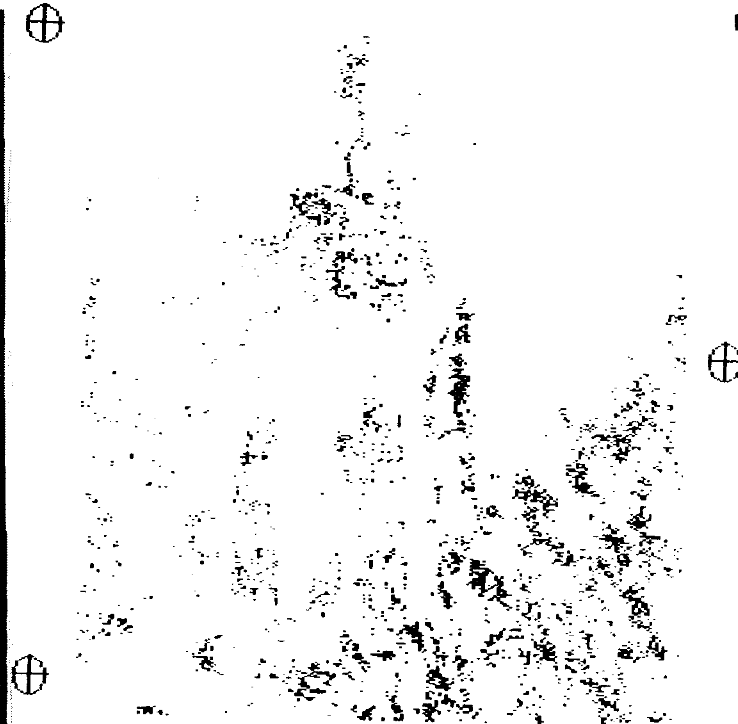


Fig.2. Separation stencil with registration marks for color register #6 - a dark, grayish green.

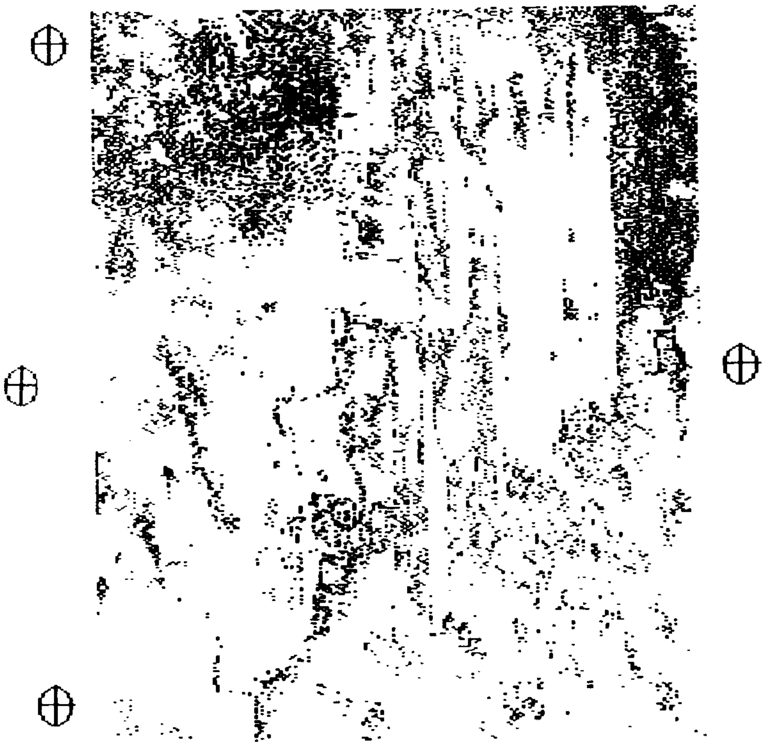


Fig.3. Separation stencil with registration marks for <u>gray-scale</u> register #6 - a dark middle value. Notice the difference between the two stencils. Fig.2 is the stencil for a specific color, whereas Fig.3 is the stencil for a value - which goes across several colors of the same value.

Hidden Geometry Stereograms

Peter Moos
Moosealogix
301 N. Harrison St. #449
Princeton, NJ 08540

Random dot stereograms (RDS) are a pair of apparently random dot patterns which reveal a hidden, coherent image under stereoscopic viewing (see Fig. 1) [1]. RDS are a striking challenge to the understanding of visual perception—how are are two monocularly meaningless images matched in stereopsis? Prior to Julesz's demonstration of RDS in 1961, most theories of stereopsis relied on matching of image features such as edges and corners. Such features are not present in RDS, and a number of new theories of visual information processing have appeared as a result (e.g. [2]).

As recently as 1978, the computer equipment required to produce RDS was beyond the means of many researchers [3]. Today the computers and laser printers required to produce RDS plots are commonplace. Postscript laser printers in particular allows us to easily explore some obvious (and rather trivial) extensions of the RDS paradigm. In what might be called "hidden geometry stereograms", the monocularly viewed RDS pairs are not random but contain obvious images which are different from the image geometry revealed under stereopsis. Figs. 2 and 3 show two such stereograms.

Learning to See Random Dot Stereograms

Viewers are invariably amazed by their first experience of successfully fusing a RDS, but achieving this first viewing typically involves 5-10 minutes of failed attempts and visually induced headaches.

To view a RDS, first find the closest distance at which you can focus on your finger (perhaps five to ten inches). Then hold Fig. 1 at about twice this distance. Relax and cross the eyes slightly so that three (rather than two) out-of-focus figures are visible. Adjust the eyes until the width of the middle figure can be kept the same as the two peripheral squares (so it also appears as a square). Once the middle figure is a steady square, be 'aware' of it for several minutes until it comes into sharp focus. You will see a smaller figure hovering about half an inch above the background. Relax and try again another time if it does not work—the experience is worthwhile.

Stereopsis requires different perspective views of a scene to be presented to each eye. In crossing the eyes so as to produce three equally sized images, the left eye is directed at the right figure and sees the left figure peripherally on the left, and vice versa for the right eye. Stereopsis occurs when the right figure/left eye and the left figure/right eye images are perceived as being the same virtual object.

Since the eyes are crossed, this virtual object is located not on the page but somewhere between the eyes and the page. As a result, initial attempts to bring the middle figure into focus often fail as the eyes shift to reconverge on the page.

Hidden Geometry Stereograms

Fig. 2 shows a stereogram consisting of rectangles at different depths. This figure should be easy to see once Fig. 1. has been successfully fused.

Fig. 3 is a random dog stereogram. Stare cross-eyed at this figure until tiny dogs leap out at you.

Conspiracy Stereogram

In Figs. 4 and 4a, the flag of the Republic of Durundi (a translucent square) is hidden in a portrait of Oliver North. In order to encode both the portrait and the hidden flag, there must be two different types of visual information: luminance information to encode the portrait, and positional information to encode the shift which creates the stereo effect. The dual encoding is achieved using a custom halftone scheme which has several alternative halftone cells for each desired grey level. The halftone cell for each pixel is selected at random from among the cells corresponding to the pixel's grey level. Portions of the resulting random halftone pattern (not the image) are then moved to create the stereo pair image. Since the original portrait is clearly visible along with the stereo image, the stereo image appears as a translucent figure floating over Mr. North. It is somewhat more difficult to see than the previous figures because of its translucency.

Implementation

The figures were created by several Macintosh Common Lisp programs which generated encapsulated Postscript files as their output. The outline of the dog in Fig. 3 was obtained by parsing an Adobe Illustrator88 drawing. The scan of Oliver North was downloaded from Compuserve.

References

1. B. Julesz, *Foundations of Cyclopean Perception.* Chicago, U. of Chicago Press, 1978.

2. B. Julesz, Global Stereopsis: Cooperative Phenomena in Stereoscopic Depth Perception. In R. Held, H. W. Leibowitz and H.L.Teuber (eds.), *Handbook of Sensory Physiology,* vol. VII Perception, Berlin, Springer Verlag, 1978, pp. 215-256.

3. S.M. Anstis, I.P. Howard, and B. Rogers, A Craik-O'Brien-Corrnsweet Illusion for Visual Depth. *Vision Research* 18:213-217 (1978).

Fig. 1. A random dot stereogram.

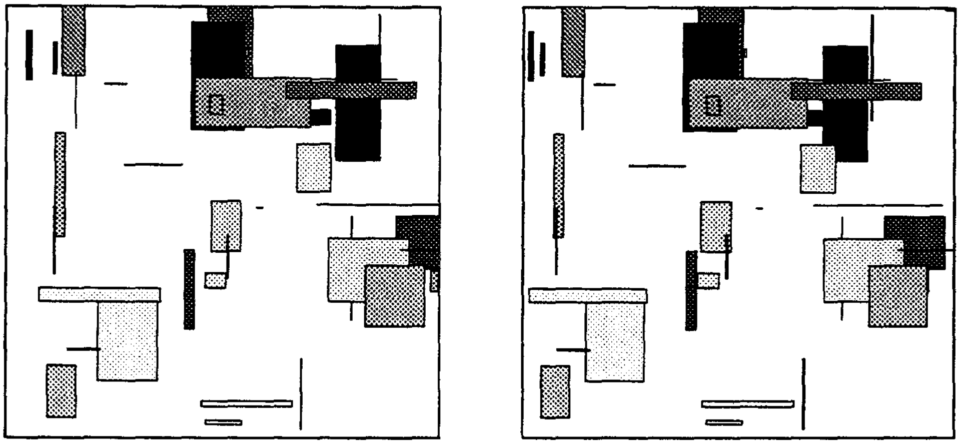


Fig. 2. A random rectangle stereogram.

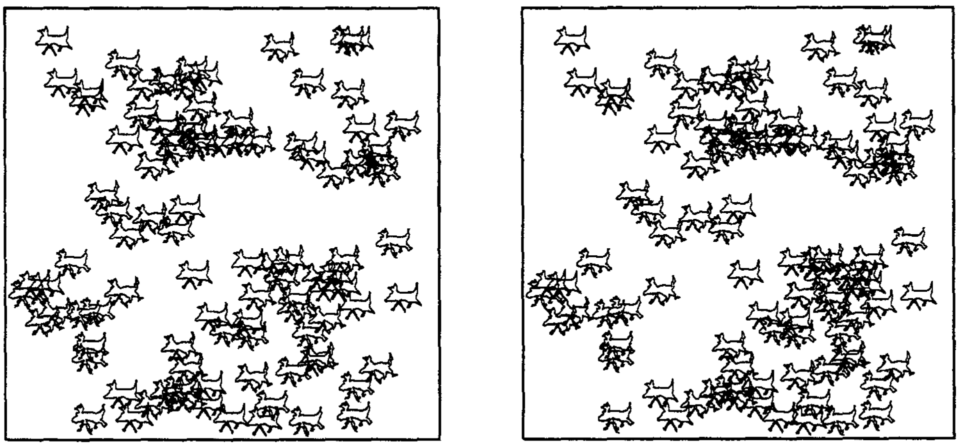


Fig. 3. A random dog stereogram.

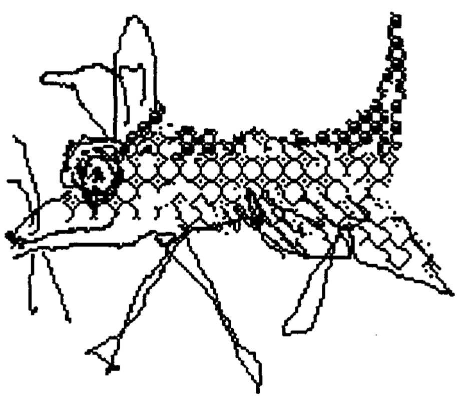


Fig. 3a. Detail of Fig. 3.

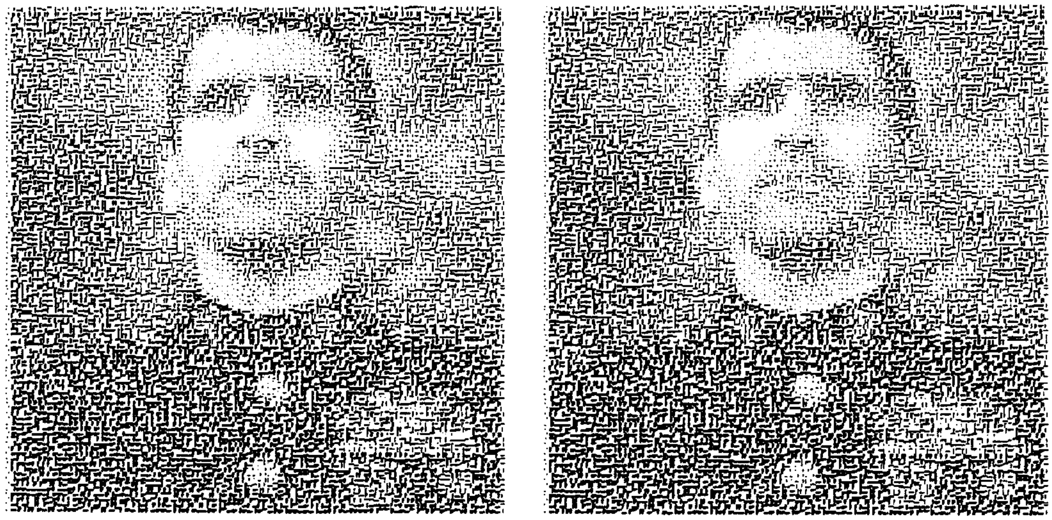


Fig. 4. A conspiracy stereogram.

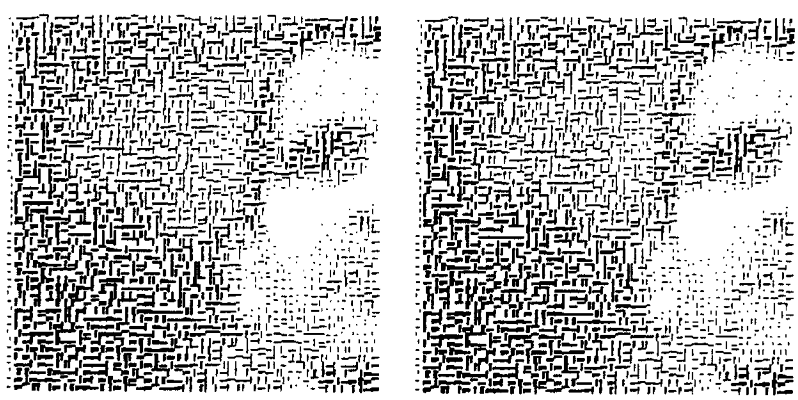


Fig. 4a.

An Interactive Laser Fibre-Optic Light Sculpture for The Carnegie Science Center, Pittsburgh

Rob Fisher
228 N. Allegheny St.
Bellefonte, PA 16823

INTRODUCTION:
In this paper I present a brief overview of five major sculpture projects which utilized three-dimensional computer imaging. Each project is site-specific, created within the context of architectural spaces and elements, what I call Environmental Sculpture. These are all very large scale works 30'-120' in height, 25' -800' in length. All but one are interior works. They are all aerial sculptures: three are suspended, one is cantilevered, and the last is mounted on top of a building. All the projects made use of architectural software on an HOK System, and an Evans and Sutherland PS300 color vector graphics engine. An Amiga 2500 was used for preliminary studies. The last project employed a Silicon Graphics 4D/220GTX. Each sculpture required the development of its own special program depending upon how I interpreted the artwork and the process by which it was to be realized.

1) OSAKA-SKYHARP 1986
Located in a very contemporary hotel in Osaka, Japan, this first example illustrates the use of walk-throughs, viewing the sculpture in its site from various vantage points. The site was simulated first, eliminating all but the most rudimentary of lines that define the volumes and planes of the environment. The sculpture was created by continuously adding, subtracting and relocating the small geometric shapes and delicate vertical lines until the form of the final sculpture emerged. This process took advantage of the computers capacity to store the x,y,z coordinates of each shape and line. I could make global changes in density and composition using a special control screen that revealed both a flattened layout and three-dimensional rotating image. At the time of construction, a printout gave me every key dimension and location of each piece.

The sculpture suggests origami, a thousand cranes, cherry blossoms falling. In fact during its conception I was conscious only of the rich variety of densities and shapes I could create as a result of computer visualization and some special software. The transparency of the sculpture creates a constantly changing sense of form and space as the viewer walks around it. This effect would have been nearly impossible to model by conventional means.

2) A PAGE FROM THE BOOK OF SKIES 1989
Site factors affecting the composition of this work for a new medical center in Saudi Arabia were studied using a wire frame 3-D model. I was inspired by the calligraphy of the Middle East, and the imaginary vision of an oasis of green vines and water. I also studied the geometry of mosaics and woodwork seen in Saudi architecture. I used the project to further explore the sense of organic complexity when geometric shape is piled upon geometric shape.

I used the experience of travelling to Saudi Arabia to supervise the installation as the source for a new multi-media interactive jazz-video performance. "Flying Carpet" premiered at the Knitting Factory in New York in June 1991. Video projected images-sculpture, crafts, landscapes and architecture-from two laser discs and a live camera were mixed improvisationally in real time, while three jazz musicians performed the images as a score. Amiga computers, a Video Toaster, and DSP time base correctors were the major hardware components.

3) FANDANGO 1990
Fandango was inspired in part by the swirling Spanish dance of that name and by the structure of wings. I wanted to create forms that floated like wings in the highly sculptural space but found little

to attach them to except the 60' high steel trusses supporting the tensed fabric roof of this very unusual mall in Florida. Then I saw their relationship to shelf fungus that cantilevers from tree trunks.

These delicate fan shapes range in size from 25' to 60' across and are up to 40' in length. Working closely with a consulting engineer, the architects and roof manufacturers, an architectural simulation of the roof was combined with a CAD sculpture software in which deflection, balance, movement and span were all specially programmed into the image. Each object was built and tested first in the computer before proceeding into fabrication. Color studies and preliminary 3-D simulations were executed on an Amiga computed. combining all *twelve* objects with the roof required the Evans and Sutherland PS300 color vector graphics system.

4) SYMPHONY OF THE AIR 1991

This was an instance when I found it easier not to use the computer or at least restrict its use to only that which I could not achieve by other means. The problem became modelling a sculpture whose combined length was over 800'. After I discovered the symphonic nature of the work, I quickly broke it down into separate but related movements.

In my studio I constructed a 1/5 scale model of a typical skylight bay whose real dimensions were 32' wide x 35' long x 46' high. Then I fabricated out of smaller scale chain a vocabulary of linear elements of varying length that I could select and combine to create a variety of aurora-like veils. Treating the skylight as a grid, it was easy, more direct and clearer for me to create each of the 13 "movements" in this miniature space than to attempt the same in the computer. Experiencing the forms directly with lighting and from many angles and being able to quickly change length and composition would have required a special program and stereo imaging. In my estimation the computer simulation would have been far less satisfying as a sculptural experience.

However, the computer was used to store these maquettes as three-dimensional information so that I could come back later and compose the entire work, giving it sequence and dynamics. The location and lengths of the chain were entered by my assistant into a 3-D model of the space and the sculpture. Each of the 13 movements were compiled in this simulation which output a perspective view of the entire 800' long skylight. I could make global changes very quickly and get a sense of the entire composition.

5) AN INTERACTIVE LASER/FIBRE OPTIC LIGHT SCULPTURE FOR CARNEGIE SCIENCE CENTER, PITTSBURGH, PA

This project, a monumental work-in-progress for the roof of a new science center in Pittsburgh,, inspired an unusual approach to three-dimensional form building. What if I could create sculpture out of science, growing forms based on scientific principles. From dialogues with eminent scientists at Penn State's Materials Research Lab and the Alcoa Tech Center, I envisioned a concept of polycrystalline forms, the growth of which could be simulated in the computer.

"GROWING" POLYCRYSTALLINE SCULPTURE

a) seeds are arranged in a cube of space. The seeds are the centers of growth of each crystal. Seed sets can vary dramatically from random to ordered, from less dense to more dense, all governed by mathematical conditions. The character of the original seed composition influences everything that follows so that it becomes one of several means of sculpting the form.

b) architectural volumes are floated among the seeds. These first examples are very simple wedge shapes. I am now developing concave—convex forms, and multiple complex forms. The seeds are instructed to grow and form crystals only if they lie inside the volume. The volume can be scaled and rotated in the sea of seeds capturing the crystals in a wide variety of ways.

&) When crystalline forms emerge that show sculptural attributes or have a particularly evocative character, they are saved and stored for the next phase. In the final stages where the forms will be integrated with the architecture I will shape the crystals further, eliminating some, stretching and pulling the form in any direction, scaling and adding more facets or nodes as I feel.

The computer, programmed with special software that simulates crystal growth, generates generic forms that are more organic and lifelike than any I could achieve. I am taking advantage of the artificial intelligence of the program to produce something beyond my capabilities.

SPACEFRAME TECHNOLOGY AS A MEDIUM FOR SCULPTURE

I saw a relationship between the computer generated wireframe images of the crystals and spaceframe structural systems. These spaceframe systems, made up of nodes and tubes are normally rather rigid and repetitious in their design. But on further study, I discovered that they could be made as organic as I wished. It appeared as though no one had written a program before now that could take full advantage of the flexibility of computer-aided-manufacturing which allowed the nodes to be drilled at virtually any angle.

The crystal growing program allows me to create the most imaginary and beautiful of forms. The data from these forms is dumped to a TK50 tape and sent to the spaceframe manufacturer where it is restored on their CAD system. The engineering, economic and structural aspects of the spaceframe crystal can be calculated using existing programs developed by the manufacturer.

THE CREATION OF AN INTERACTIVE LIGHT INSTRUMENT

I recommended that the sculpture take advantage of two advanced technologies for lighting—lasers and fibre optics. I proposed creating a light instrument to drive the laser beam onto arrays of fibreoptics that would branch throughout the crystalline sculpture in a metaphor of the communications industry.

I have invited several companies and the Electronic Visualization Lab at the University of Illinois to collaborate in the creation of an instrument that utilizes current and future man/machine interfaces— sound, music, voice, movement, special programs, MIDI, neural nets, and virtual reality techniques—to vary the color, intensity, movement and animation effects on the sculpture. It is to be an instrument that can be updated throughout the years as new interface ideas are invented. It will be designed so that it can be used by artist/composers from many fields. It could be linked via satellite to centers throughout the world where international artist could perform a work in realtime for the center.

For example, it will be possible for the Pittsburgh Symphony to perform live at Point Park, send their sound as digital data via laser, satellite, or phone lines to the sculpture across the river to the Carnegie Science Center, whereupon it can be manipulated as light sculpture on top of the Omnimax theater. Or, people could tune into a special broadcast stereo concert, park their cars across the river and watch a "son et lumiere" spectacle.

Imagine as a small child approaches the huge 150' tall object and speaks to it through a megaphone; whereupon the sculpture reacts with a burst of light. A dancer could be projected into the middle of the sculpture using virtual reality technology and climbs around and through the crystals creating sounds as a function of movement or of which part of the sculpture is touched.

Or an artist in Eastern Europe or Russia creates a program on a disc, mails it to Pittsburgh and it is performed by inserting the disc into a home computer.

CONCLUSION

From the composition of the initial seed set, to the shape of the architectural volumes, to the positioning and scaling of the volume in the seeds, to the recognition of potentially dynamic crystalline forms, to their final shaping as sculpture that is both architecturally and structurally sound. From the application of lasers and fibre optics as lighting, to the control panel of the future, from music to dance as interfaces—this project represents a profoundly different approach to art. Is it art as science? Science and technology as art? More likely it is a new, as yet uncharted, symbiotic relationship between artist and machine joining the artificial intelligence of the computer, the products of high technology, and the intuitions of the artist to produce forms which would otherwise remain as dreams.

Driving Machinery with Computers

John Goodman
Joint Ventures
9 Montague Street
Cambridge, MA 02139

Work done by computer is often ephemeral and disembodied. In some instances this may be the intention. I am more interested in making things; real solid things that people can handle and use. I am also interested in making computers useful for manufacturing as well as designing objects.

For the past ten years I have been building custom furniture, often other peoples' designs. The making is what interests me, more than the designing and certainly more than the appearance. I have been using computers to preview designs for clients. Generally, when an architect or designer presents me with a plan for a table, desk or cabinet, I will make a simple 3-D computer model and do a few basic renderings of the furniture-to-be to help the client visualize the design.

About three years ago, I began thinking about how helpful it would be to use the drawings I was making on the computer as templates for the production of the actual furniture. If the computer already had a method for making these shapes visible, I should be able to use the same information to make the shapes tangible. It seemed foolish to discard all the effort that went into the drawing when it came time to begin cutting wood. After all, writers could pass their computer-edited manuscripts on to book editors, who could work with these files and use them to set type on the ultimate printed page.

I was familiar enough with computers then to know how they were programmed, even though I did not have a lot of programming experience. My idea was to direct a machine along the same X-Y path in space that I saw in pixels on my Macintosh. It seemed like a simple enough idea. I am surprised and a little disappointed at how long it has taken to make the idea useful, and I am stretching a bit to even consider the project now as being useful. But it is a straightforward plan.

The fundamental link between the computer screen and a mechanical device is a type of motor called a stepper. A stepper motor is, in a sense, a digital motor that moves in discrete "steps" rather than running continuously at a predetermined speed. My plan was to "map" the pixels of the screen to the steps of the motor. If a horizontal line was 200 pixels long, I would have my stepper motor take 200 steps, (or some multiple of this number) and drive a mechanical arm a similar distance. If the line was diagonal, 30 pixels across and 12 pixels up, the horizontal and vertical arms of my machine would move similarly and in conjunction. A series of small straight lines could approximate a curve.

This is not a novel idea; it is exactly the way a traditional pen plotter operates, and is the basis of many existing industrial (read "$$$$$$$$") machines. My only contributions to the art were the desire to use PostScript, the prevailing file format for saving computer screen images, as the input file for controlling my machine, and my determination to keep the project simple and cheap.

My motivations for economy of concept and cost should need no further explanation, my interest in PostScript deserves a little. PostScript came to acclaim as a "page description language" for desktop publishing.

The success of the desktop publishing concept has made PostScript a near universal file standard, because PostScript, or a cloned version of it is the lingua franca of the laser printer industry. If you want to speak to a laser printer, you speak PostScript.

Much of PostScript concerns itself with page layout. But a simple and remarkably flexible subset of the language addresses drawing. It is not only the word processors that need to save PostScript files. Drawing and illustration programs save their files in PostScript as well. And within the universe of PostScript, there are only three basic commands needed to create any drawing. They are "moveto","lineto" and "curveto."

So the plan came down to learning enough PostScript to understand when my machine needed to either move from point to point, or similarly cut a line, or likewise follow a curved path. Ultimately, a curve is reduced to a series of arbitrarily short and numerous lines. And the difference between a move and a cut is only whether or not the cutter is touching the work, so my only challenge was how to make lines. Since my Macintosh obviously had that down pat, it seemed to be a reasonable challenge to figure out the rest. It is then humbling to admit that after three years, work still remains...

What I have done so far is:

> Write a software program that reads PostScript drawing files. (They are ordinary text files which can be read even more easily by people.)
>
> Convert the sweepingly general "curveto" command into a specific chain of short lines.
>
> Assemble these now faceted curves and any associated bona fide lines and moves into a long string of connect-the-dots instruction.
>
> Further reduce each line into trivially brief stepper motor commands. When two motors are being driven (the common case for a diagonal line) the choices are either a step for only one motor (clockwise or counter-clockwise) or a step for both motors.
>
> Get this information out of the computer and into the machine.

I prototyped my machine by attaching two small stepper motors to the knobs of an Etch A Sketch. When used to recreate Macintosh screen drawings, it very quickly reveals whether the motors are turning in the right direction, or moving properly relative to one another.

I also have a piece of surplus electronic assembly equipment which moves like an X-Y plotter within an open frame of approximately 20 inches by 40 inches. I attached a small woodworking router to the movable head of the machine and added a short Z-axis that allows the router to be raised and lowered approximately one inch. This allows me to comfortably cut into 3/4 inch material, which is a fairly standard thickness for solid wood and plywood as well.

I have also been developing the software towards the specific uses and needs of the sign-making industry. Many manufacturers produce what amount to heavy-duty plotters which carry a knife in place of the pen and cut into adhesive-backed vinyl. While there is much hardware available, there is little software, and nothing on the Macintosh which looks or works like Mac software. I am working with a company that will market my software for these machines.

I am also working towards simplifying the electronic hardware link between my Mac and my machine, and looking into mechanical equipment more hardy and more suited to the tasks I demand of it.

I welcome any communications with like-minded hackers who may be similarly motivated to put machines under computer control. I am trying to figure out the best uses for this computer/machine hybrid and encourage all suggestions. Much can be done, but the ideas must be raised before they can be implemented.

APPENDIX: More About Stepper Motors

Unlike common motors which begin spinning when you hook them up to some voltage and keep going at some speed until you shut them off, a stepper motor needs a lot more guidance if you expect it to do any work.

As the name suggests, a stepper motor moves in a series of steps; it may require 200 or more "steps" for the motor to complete one rotation. Like an extremely aggressive dance partner, the designer must force the motor to follow the proper sequence of steps if the motor is to be kept from hopelessly stumbling in confusion.

When a stepper motor is spinning, it is a lot like a horse being teased with carrot on a stick held continually out of its reach. When you power up a stepper motor, the rotor is irresistibly pulled into the nearest stable location. You begin its rotation by pulling the rotor towards the next stable position. But before it gets there, you change the target and draw the motor on to the <u>next</u> stable location. This can go on forever, and the motor will go round and round.

To continue the horse/carrot analogy, at any time you may stop the stepper motor by letting the horse get the carrot. Freezing the sequence of step patterns by which the motor is "motivated" will freeze the rotation. The motor will hold fixed in the last position it reached. When you resume the sequence, the motor will resume the rotation. Reversing the sequence reverses the rotation. Increasing or decreasing the rate at which the sequence is presented will speed or slow the rotation.

The greater the number of steps in one rotation, the smoother the movement. Motors are designed with a fixed number of whole steps, but can be electronically persuaded to take smaller partial steps. The basic sequence is known as "full-stepping." Each step can be written as a four digit binary number (four places which have a value of either one or zero.) This style of representation makes it easy for the computer and also gives a visual simulation of how the rotor moves. If you follows the passage of the "ones" through the number, you get a sense of how the rotor spins. A full-step sequence looks like this:

1100
0110
0011
1001
1100
0110 etc...

with only the first four steps being unique.

A slightly more sophisticated step sequence moves a bit more gradually and needs eight steps, not four, to complete one cycle. It is known as "half-stepping" and looks like this:

1100
0100
0110
0010
0011
0001
1001
1000
1100 etc...

There are increasingly subtle adjustments to the sequence that are known as "micro-stepping" which leave the digital binary world and enter into the murky gray analog domain. Basically, rather than a given value in the step being simply on or off (one or zero), a step value can be <u>partially</u> on which moves the rotor <u>partially</u> to the next step. My understanding of the technique does not go much beyond that crude explanation so I will leave it at that.

Robotic Painting: A Little Manifesto

Tim Anderson
MIT Electronic Research Society
MIT Branch P.O.Box 202
Cambridge, MA 02139-0902
email: tim@media.mit.edu

Abstract:
Robotic painting and its potential as an art medium are explored in this paper. Two specific robots built by the author are described. Both use ordinary brushes to apply paint to a canvas.

Keywords: robotic painting, robotics, art, painting, oil painting, artificial intelligence.

WHAT IS THE POTENTIAL OF ROBOTIC PAINTING?

New Types of Images, New Textures

Why build a robot to paint? By arming a robot with paint and brush, it is possible to achieve new effects: sweeping strokes longer than the reach of a human arm; precise detail too fine for human fingers, or even the naked eye; complex images made up of thousands of strokes, painstaking detail that would take years for a human artist to complete.

The precision of the device can be used to lay up layers of transparent and tinted paint in registration, producing hologram-like illusions of depth by presenting each eye with a slightly different view.

Actual three-dimensional scenes with great detail can be built from paint in extreme relief. Transparent and opaque paints can be daubed together into a mass several feet thick, the depths of which contain myriad objects built of paint by the robot's brush, intertwined, viewable from any angle.

The robot is capable of moving the brush at speeds and in patterns that are not possible for a human. This permits the use of fast drying materials, which can be worked in different ways at the various stages of the drying process. For example, in order to combine two images in a unique way, a thick layer of paint could be allowed to develop a skin, which is then tugged, punctured and indented, superimposing the second image on the first in relief. These two images, one of pigment and the other of texture, could be of great complexity and still be completed before the materials lose the necessary working properties.

No human need be present as the robot works, so toxic and explosive materials can be utilized without endangering anyone.

Beyond Reproduction

The computer that controls the robot could be a "Word-Processor" for paint. An artist could grasp the robot's brush and commence painting as normal. Each movement of the brush would be recorded by the robot's motion sensors. This information could then be edited, manipulated, and "played back". Like the musician's sequencer, the best "takes" could be edited, saved, and "cut-and-pasted" into a "multitrack painting". Even images which would ordinarily require laborious, time-consuming techniques could be committed to canvas before the inspiration is lost. In experimentation with variations in color, shading and composition, the repetitive work could be delegated to the robot, like an assistant able to exactly mimic the brush-strokes of the master. The "stroke-recordings" of a human painter could be saved as a form of immortality for the artist, motion caught and held for posterity. A student could grasp the brush as the robot is playing back these stroke-recordings, and be guided by the feel of a master's techniques.

Water-paintings and Buried Images

Rather than simply viewing the process as a means to some static end, the choreography of brush and paint can itself be the work. For example, the robot's brush could apply water to a wooden panel, and as the pattern of dark, wet wood dried and disappeared, more strokes would be added, to make a continous series of changing images, every instant being a final image, the work itself being the sum of these ephemera. The same process could be used with permanent pigments, each stage being buried and obscured by the next, but not lost, because by saving it in the robot's memory, it can be performed again and again.

Give the Tin Man a Heart

Through simple accelerated mimicry, a robot could certainly out-perform the humans currently painting on third world assembly lines for U. S. mall galleries. But why waste the potential of such a powerful tool on so banal a task? The robot is no mere mechanical device. Moving under the control of a computer program, the robot can be instructed to behave in ways limited only by the ingenuity of the programmer. By making programmed decisions based on stochastic processes[1], real-world sensors, or disk files which store memories of previous paintings, the "robot artist" can produce works that surprise even the robot's own creator, the tracks of the brush leaving a record of each twist and turn in the program that drives it.

Allowing the machine to decide what to paint raises a number of challenging questions. Who is the author of a given work, the machine or the maker of the machine? Can a machine be endowed with creativity? What is creativity?

VAN GOGO™: A ROBOT WHO PAINTS

In order to test and explore the concepts of robotic painting, I have built two systems. The first is fairly straightforward, and could be built inexpensively by a reasonably skilled technician. The second is more elaborate and has greatly extended capabilities.

Van GoGo™ Jr.

Van Gogo Jr. is intended to make robotic painting accessible to anyone. The prototype of this device was built from a cast-off teleprinter and an 8" floppy drive at a cost of One Dollar. An ordinary paintbrush was attached to the head positioner from the floppy drive. This was in turn bolted to the printer's carriage.

A PC running programs written in Turbo C is used to control this device. The computer's printer port is plugged into the circuitry of the butchered drive's controller board. Characters sent to the port turn the various windings of the head positioner's steppping actuator on and off[2], and move the brush in steps toward or away from the canvas at about a half millimeter per step. The printer's electronics are left intact, except for the removal of the print head. The PC sends ASCII characters to the ex-printer through its serial port. This induces it to move the carriage left or right and "paper feed" a fiberglass drum with the canvas attached to it a "line up" or a "line down". Thus, 3-axis control of the brush relative to the canvas was achieved. As it paints, the brush is first moved to a paint container, dipped in paint, then moved to some location on the canvas, and dabbed, stroked or shaken upon the surface.

Van GoGo™ Sr.

Van Gogo Sr. looks like a wall about 12ft. square, with pulleys at the corners. A canvas is mounted on the wall, equidistant from the pulleys. A brush is suspended over the canvas by threads running to the four corners. A fifth thread holds the brush away from the canvas and runs to a spool of thread and electric motor on a stand a few feet away. The paint is below the canvas in a row of small containers. Mounted to the other side of the wall are the other two motors and their associated electronics boards. Ribbon cables run from these boards to a PC, which contains the servo-control board.

This arrangement of tensioned threads for the control of the brush has the lowest possible mass and the greatest possible speed for any given set of motors. The brush can be moved at a speed of one hundred and fifty inches per second. High speed is necessary due to the large number of strokes and the repeated trips to the paint pots the brush must make.

The motors have optical encoders mounted to their shafts that can discern a brush motion of a few thousandths of an inch. By sensing the friction of the brush, it is possible to determine how much paint is between the brush and the canvas. The servo-control board uses one LM629 dedicated motor control chip for each axis. These integrated circuits allow very close control of position, speed, and acceleration through the three axes of motion, without bogging down the host processor.

How Van GoGo™ Goes

In the design of a program to control Van GoGo, I have placed an emphasis on modularity, flexibility, and independance in the control of each aspect of the painting process. There are distinct program modules for the motion of the brush on the canvas, dipping and mixing the paint, and the selection of the colors for any given image area.

The style of the "on-canvas" motions and the resulting textures are controlled as one aspect of the painting process. Likewise, the module that requests a given color is distinct from the one that is charged with actually mixing colors and getting them onto the brush. These programs interact, resulting in any number of distinct styles.

It is possible to "plug in" the system in various ways with minimal modification, thereby radically altering the architecture of the "robotic artist" at will. For example, in a given configuration the brushwork could be modulated by a piece of music and color selection could be in response to the digitized signal from a video camera. In another, brushwork could be generated by a separate computer, the paint mixed by small motors squeezing paint tubes, and the colors selected based on a design the computer has generated through a scheme for evaluating color harmony and geometric balance.

References:

1. Harold Cohen. Implementing an Expert Artmaking System. 1st International Symposium on Artificial Intelligence and Expert Systems. Berlin. May 1987.

2. Anita Flynn, Editor. Olympic Robot Building Manual. MIT AI Lab memo #1230. December 1988.

"Van GoGo" is a trademark of Timothy C. Anderson.

"The Game"
Teaching Programming in an Interactive Classroom

Byron Grush
Professor of Art, Media Arts/Electronic Media
School of Art
Northern Illinois University
DeKalb, IL 60115
1-815-753-1567
email: byron@art.niu.edu
compuserve: 70670,1156

Abstract
In the Spring of 1991 I taught a course in the Electronic Media program of the School of Art at Northern Illinois University entitled "Electronic Media Processes." I previously described some of the concepts of this course in the proceedings of the 1989 SCAN conference. This intermediate graphics programming course explored user interface design from the artist/programmer's point of view. I wanted my students to become involved in the development and testing of a graphical user interface in a dynamic and interesting way. What follows is the story of the evolution of an interactive game written in part by the students in the class.

Acknowledgment
My teaching assistant for this class, Jeff Byrne, is one of our graduate students in Electronic Media. Not only did Jeff help with planning and development of The Game, he did the actual programming. Many of the ideas embodied in the The Game are the results of his creative and often ingenious problem solving abilities. My role as teacher, while it extended to directing Jeff as well as the undergraduate students enrolled in the course, became as much learner as it did mentor. At the end of this paper I have included Jeff's notes from the interface section of the game unit.

Some background on our curriculum
The Electronic Media curriculum at NIU introduces design students to computing through two 200 level courses: Fundamentals of Electronic Media, an applications course taught on Macintoshs, and Fundamentals of Computer Graphics, a programming course taught on IBMs. After completing these courses the students may enroll in 300 level courses such as Motion Graphics (animation) and Media Graphics (design for raster) but must pass a portfolio review before beginning work at the 400 (senior status) level. Advanced courses include Desktop Publishing, Media Synthesis (video output from computer "paint" programs) and two more programming courses.

Often we are asked, "why all that programming?" Indeed, we are not trying to produce programmers. However, we feel that the core of computer art is process: the creative artist at work with a "meta-tool." The understanding of that tool is best taught through the experience of computer programming. Merely teaching students the use of an application program cheats them of the full potential of the computer and builds a reliance on the work of others. We have tried to make the content of our programming courses relevant to the computer graphic design field. One of our advanced courses explores three-dimensional modeling and animation. The other, "Electronic Media Processes", the subject of this paper, explores the design of graphical user interfaces.

The Electronic Media Processes class
Students entering this class will have completed the first programming course and the introductory "applications" course. Their knowledge of the Pascal language will be limited, but they will have a basic understanding of the computer as a graphics system. They will know how to write a program, how to use data and procedures to draw images, and how to use existing graphical user interfaces. They will

be familiar with necessary MS Dos commands and be comfortable in the Turbo Pascal programming environment.

The Pascal Programming Language

I have been teaching graphics programming to art students since about 1983. My language of choice for this is Pascal. In the beginning, we used Apple II computers and UCSD Pascal with "TurtleGraphics." Pascal is an eloquent but highly structured language written by Niklaus Wirth as a teaching language. It encourages disciplined thinking and the use of predefined data types. It is readable and easy to learn even for those art students who claim to "not like math." The Borland company has a product called Turbo Pascal for the MS Dos environment. For the most applications it is "standard" enough to be called Pascal, although it is really assembly language in disguise. It makes use of "libraries" of precompiled code called units and includes a graphics unit called "Graph" with many useful primatives.

The Network

We taught the class "Electronic Media Processes" in one of the networked labs run by the university's Academy Computing Services. This was a lab originally configured for students in English and Journalism who needed word processing capabilities and was only beginning to be upgraded for graphics. The lab had IBM PC computers with VGA displays but no mice. It was interconnected with Novell networking software. The students were limited to directory space equal to 360K, the size of the largest floppy disk the PCs could accommodate. They had read/write permission only in their own directory on the network but had read-only permission in the teacher directory. In addition, we established a write-only directory called "homework" which could be used as a "deposit" space for finished projects.

We wanted to take full advantage of the connectivity of the network in designing some kind of interactive project. The limited permissions didn't allow the students to read and write into each other's directories. The impact of this was to conjure up the idea of an elaborate game they would play to take possession of a common space. The game would need to be monitored by a "master" program launched from the teacher directory.

For the style of the game we initially chose a gameboard, an 8 by 8 rectangular grid of squares which would fill the VGA screen vertically but leave a blank area along one side. The students' programs could draw this board and add any "pieces" that had been played by accessing a game database located in the read-only directory. In order to add a game piece to the database, the students' programs could send information to the write-only directory where a master program would find the move, evaluate it for legality and then update the database. Thus the master program was both a sort of post office and police station.

Planning the game

With the game board and logistics decided, we opened up the discussion for the formulation of rules for the game. What would be the object? How would the playing take place? How could all 15 students play at the same time? This phase of the process created a lot of energy and enthusiasm for the project, but it also generated too many conflicting ideas. Eventually we settled for perhaps the simplest of the ideas, but one which was practical and which allowed the students a great deal of interpretation for designing their own interfaces.

The object of the game was to fill every square on the board with a game piece or "icon" belonging to the player. This was done by placing a piece on the board according to the grid coordinates. If two or more players chose the same square for their pieces, a battle would occur which only one could win. The winning piece would remain on the board until another player fought for and won the square.

The battle for the square consisted of the comparison of attributes which were boolean values associated with the game board square and the attacking game piece. Each piece had six "attack" values and six "defend" values. Only three of the six booleans could be true. The attack attributes of the challenging piece were compared to the defend attributes of the resident piece so that a "true" attribute would win over a "false" and two trues or two falses would cancel each other out. If there was a tie, no change would occur, otherwise the winner took the square and the loser "died."

The students' programs consisted of a main "loop" which would redraw the current game board each time there was a change in the database. They could interrupt the loop with a keypress and enter a new move into the game. This could be to change the attributes of an existing game piece, or to add a new piece to the board. To make things more interesting, we restricted sequential moves to adjacent squares. If a player managed to surround one of their pieces on all eight adjacent squares, that piece became a "castle" and could not be killed by the first battle it lost. If a castle lost a battle it became an ordinary piece and could be killed or restored to castlehood at the next battle.

Designing Game Pieces

A special program named Makepce was written to allow the students to interactively design bitmapped images to be used as game pieces. These were used from the game program by calling Turbo's Getimage and Putimage procedures. The program Makepce displayed the bitmap, a 60 by 60 pixel square, as an enlarged image. A simple cursor movement routine let the student set any of the pixels in the bitmap to one of 16 possible colors.

The Game Unit

We wrote a Pascal Unit which contained types and procedures which the Master program and the student programs would use. The interface listing of this unit was available to the students to read, but they were not allowed to change or recompile the code. The unit handled calls to DOS and defined many data types which were needed to be consistent throughout all the programs.

The Pascal language allows the programmer to define enumerated types with identifiers that make sense when you read through the program listing. For example, Jeff defined a type he called "keytype" which consisted of most of the non- alphanumeric keys on the keyboard such as the function keys, the arrow keys, the delete key, and so forth. Once defined, the student could refer to the space bar as spacekey or the sequence, "control key + home key," as ctrlhome. A procedure called inkey was written which scanned the keyboard for input, then checked any input for a match against one of the keytypes.

A Game Program example

The students were given a "bare bones" program which would run the game but which had very little in terms of user interface. The main loop of this program looked like this:

```
begin  (* main *)
  initgame ('mydirectory', pathtodriver);
  entergame ('castle.PCE', inpath);
 if not error then
  begin
    repeat
      ScanForQuitters (outpath);
      ScanForPlayers (outpath);
       repeat
         ScanForNewBoard (outpath);
       until not error;
     resurrection;
      Showboard (20,1, getpiecesize);
      inkey (ch, key, false);
     case key of
      f1 : makeamove;
       f2 : invade (inpath);
     end; (* case *)
```

```
    until (key = AltX);
    exitgame (inpath);
  end;
end.
```

In this example, the loop draws the gameboard, including any pieces which have been played by other players on the network, then tests the keyboard for input. A case statement checks to see if function key one or two have been pressed. If f1 has been pressed, the procedure "makeamove" is called which changes the attack or defend attributes for the move. If f2 has been pressed, the procedure "invade" enters the move into the game by writing data containing the coordinates of the square to attack, and the battle attributes. Some of the types defined for this process look like this:

```
battle_attribute = record
            a1, d1,
            a2, d2,
            a3, d3,
            a4, d4,
            a5, d5,
            a6, d6,
             castle : boolean;
           end;


 messengertype = record
            batat : battle_attribute;
          x, y : byte;
         end;

playertype = record
          name : namestr;
          image : pointer;
        end;

territory = record
          owner : playertype;
           batat : battle_attribute;
        end;
```

Student User Interfaces
Given the interface listing of the game unit annotated with comments and the sample program which ran the game, the students were challenged to create a graphical user interface. This included designing game piece "icons" which would be drawn on the gameboard during play. They had to be able to enter the game, select a square to attack, change their battle attributes, attack, and quit. They could also provide information about the progress of the game

The master program ran during their development time so that they could try out their interfaces. They were actually able to play the game using the example program and begin to add their own "bells and whistles." It became obvious that speed of use was a desirable goal for the interfaces. One student designed an elaborate system of menus and submenus with many a "are you sure?" Although the system worked well, a more streamlined interface could play three or four times faster.

The students were stuck with the 8 by 8 gameboard grid which filled the screen vertically, but they could position it horizontally to leave some space on one or both sides. In some cases, this area was used for "buttons" or menus which allowed the user to implement necessary functions of the interface. Other approaches included the opening of "windows" in which menus or information would appear. Nearly everyone started their program with a title screen. Some allowed the selection of different icons for use as the playing piece. Few got as far as including a game update window showing the status of the game and so forth.

Thematic interpretations of the game were varied. One student began to write a history of his main character, "Bloodwolf," who gave the game a "Dungeons and Dragons" flavor. Others interpreted the battle attributes in terms of weapons and armor. Another student produced a slick, businesslike interface which might translate well into a database manager or word processor. Many redesigned the color scheme of the gameboard to match their interface's special pallet.

Conclusion
At the end of the four or five weeks the students spent on this project everyone had a working version of their interface, though some were more finished than others. We had a few sessions of the game with nearly the whole class playing at one time. No one actually won the game by filling the game board with their piece. Everyone was enthusiastic and work hard on the project so we might say, since "it's not whether you win or loose that counts, it's how you play the game," that eveyone won.

Explannation of the Game Unit by Jeff Byrne

procedure inkey (var ch : char; var key : keytype; wait : boolean);
This procedure scans for keyboard input. If wait is true, then the procedure will wait for a key to be pressed, otherwise, the procedure will exit withvalue of key as nullkey. Keytype is a user defined type with names for commonly pressed keys. (cariagereturn, escapekey, leftarrow, etc.) Ch is the character pressed if key = textkey.

procedure initgame (whoareu, pathtodriver : pathstr);
This procedure tells the computer who you are and initalizes graphics and other variables used by the program. This should be one of the first procedures called.

procedure loadpce (var memory : pointer; path : pathstr);
This procedure allows one to load an image that was created in makepce. Memory is a pointer that will point to the image bitmap. Path is the name of the file to load. If the file is not there then error is true and the procedure exits. Example:

```
var
 count, xpos : word;
 button [1..5] of pointer;

begin
  loadpce (button[1], 'club.pce');
  loadpce (button[2], 'spear.pce');
  loadpce (button[3], 'rock.pce');
  loadpce (button[4], 'gun.pce');
  loadpce (button[5], 'sword.pce');
 xpos := 10;
 for count := 1 to 5 do
  begin
    putimage (xpos, 40, button[count]^, copyput);
    inc (xpos, 10);
  end;
end;
```

This example uses an array of pointers to keep track of the images. The array is filled with images using loadpce. These 'buttons' are displayed on the screen using putimage.

procedure showboard (xpos, ypos : word; piecesize : byte; custom : freespace);
This procedure will display the gameboard with the upper left corner at (xpos, ypos). Piecesize controls the size of the grid and should be 59 for this game. Custom is a procedure that draws the empty squares. If custom = defaultproc then the original (bright colored) board is drawn. Otherwise, custom is used to draw the empty spaces.
Example:

```
{$F+} (* use far calling conventions *)
procedure myboard (x1, y1, piecesize : word); (* formal parameters MUST be the same *)
begin
  setfillstyle (solidfill, blue);
 bar (x1, y1, x1 + piecesize, y1 + piecesize);
end; (* myboard *)
{$F-} (* let computer decide calling conventions *)
showboard (1, 1, 59, myboard);
```

procedure territory2messenger (xpos, ypos : byte);
This procedure does several different things. First, it checks to see if (xpos, ypos) is a valid move for you to make. If not, the procedure will buzz and exit with error true. Otherwise, the variable messenger is updated to the current battle attributes of that space. Messenger is used to communicate between your program and the master. This procedure must be called before setting the battle attributes with toggleattack and toggledefense and calling invade.

procedure toggleattack (attack : byte);
This procedure will toggle the attack attribute attack in the variable messenger. Attack is the number of the attack attribute (1 - 6).

procedure toggledefense (defense : byte);
Same as toggleattack except for defense.

procedure invade (inpath : pathstr);
Invade sends your messenger to the master program. Inpath is the path to the input directory (the write-only directory). Invade checks to see that the messenger is valid (not more than 3 attack or defense values can be true). If valid, then the messenger is sent and processed by the master program. Otherwise, the procedure exits with error true. Territory2messenger must have been called before calling invade.

procedure ScanForPlayers (outpath : pathstr);
This procedure will scan for new players and add them to the game. Outpath is the outgoing path (instructor's read-only directory).

procedure EnterGame (pname, inpath : pathstr);
This procedure sends a message to the master program telling it that you want to play the game. Pname is the name of the piece that you want to use during the game. You can only use one piece at a time. Inpath is the ingoing path (students' common write-only directory). If your piece can't be found, you are already playing, or you did not exit properly last time (while the master is still running) - then error will be set to true and the procedure will exit.

procedure ScanForNewBoard (outpath : pathstr);
This procedure will look to see if a new board has been created by the master program. If a new board is found then the local gameboard is updated. Otherwise, error is set true and the program exits. This is why the program game.pas has a repeat until not error loop around this line. Without this loop the program will continue even if the a new board had not been made. You can remove the loop if this is what you want. Outpath is the outgoing path (the read-only directory).

procedure resurrection;
This procedure checks to see if you are dead. If you are dead, you are given another chance and brought to the same condition as if you had just started.

procedure exitgame (inpath : pathstr);
This procedure tells the master program that you want to quit. The master, in turn, tells all the other players that you are a loser and want to quit. All of your territories are now free spaces. Use exitgame to end the game. Otherwise, you will not be able to reenter unless the master is restarted. Inpath is the ingoing path (write-only directory).

procedure scanforquitters (outpath : pathstr);
This procedure looks for a message from the master program telling it that a player has quit the game. The player is then removed from the local game as well. Outpath is the outgoing path (the read-only directory).

function validmessenger : boolean;
This function is false if you have tried to set more than three attack or defense attributes.

error : boolean;
This is a global variable used to detect an error during a procedure.

ch : char;
This is a global variable used with inkey.

key : keytype;
This is a global variable used with inkey.

RenderMan™ and Shaders

Mitch Prater
Bill Kolomyjec
The VALIS Group
P.O. Box 422, Point Richmond, CA 94807-0422

Abstract

In 3-D computer graphics, shaders provide appearances to geometric entities and are analogous to fonts in 2-D picture making. Pixar's RenderMan Interface Specification describes an open architecture for renderers and a procedural shading language as a means for producing high quality synthetic pictures allowing shaders to be written outside of a RenderMan-compatible renderer. A discussion of salient issues involving RenderMan and Shaders, shader development, and shader functionality is presented.

Introduction

Pixar introduced the RenderMan Interface Specification[1] to organize modeling and rendering activities. The RenderMan Interface is a paradigm and a methodology for the exchange of 3-D scene description information between modeling and animation software and rendering software. The RenderMan Interface Specification is both open (the details are published, not proprietary) and industry defined. Because of this, the RenderMan Interface Specification has become an important standard in the computer graphics industry. Any manufacturer's renderer that fully implements a prescribed set of 3-D information routines can qualify as a RenderMan-compatible renderer. As of the SIGGRAPH '91 conference in Las Vegas, Nevada there were five announced RenderMan-compatible renderers either on the market or in progress. Pixar's PhotoRealistic RenderMan is but one RenderMan-compatible renderer, albeit the first. It is distinguished by its incorporation of the Reyes rendering algorithm[2] also developed by Pixar.

The combination of the RenderMan Interface Specification and a RenderMan-compatible renderer such as PhotoRealistic RenderMan aims to remove the "computer graphic" look in synthetic imagery through:

- properly applying the concepts of sampling and filtering throughout the image generation process.
- facilitating continuous surface geometry that truly represents an object rather than using polygonal approximation.

• providing control over the image quality through filtering of the geometric data in space and time.
• providing extensive lighting, atmospheric, and camera effects.

No other 3-D standard as comprehensively addresses quality issues. Historically, PhotoRealistic RenderMan and the RenderMan Interface Specification were developed to integrate computer graphic animation with live action; quality was an issue from their inception. Many awards and accolades speak to Pixar's accomplishment. Images made with PhotoRealistic RenderMan do have an inherent film-like "photorealistic" quality about them. This is no fluke! The success of RenderMan is attributable to both the design of the renderer and the shading language that allows aesthetic effects to be developed as shaders.

Shaders Empower RenderMan

Within the RenderMan paradigm, the RenderMan Interface is the vehicle for conveying all the geometry and other 3D scene and 2D image composition and filtering information from a modeling or animation application to the renderer. However, there is more to generating an image than geometry, scene composition, and filtering. Most important to realistic image generation is the additional ability to control the other visual aspects of a scene, namely, the surfaces of objects, the lighting, and atmosphere. The surfaces of objects can appear to be made of any conceivable material or effect from wood grain to clouds to polished chrome. Lights can be of any conceivable light source from sunlight to ambient light to light emanating from a film projector. And, if the image maker chooses, the atmosphere could be hazy, clear, or filled with colored smoke.

Additionally, in a sophisticated renderer, the surfaces of objects should be able to be altered or perturbed slightly in a procedural way. This surface displacement capability relieves the modeler from having to incorporate fine visual features such as the threads of a bolt, the treads on a tire, or the texture of tree bark, into the geometric model. Instead, the renderer can add these visual features during rendering, relieving the modeler of this considerable (and often insurmountable) burden.

In the RenderMan scheme, all this additional visual information is contained in small programs called "shaders". Virtually every visual aspect of a RenderMan image is controlled by a shader, from the atmosphere, to lights, to surfaces. Without shaders, a RenderMan image is uninteresting in the extreme. All objects have the same flat appearance and there are no lighting effects whatsoever. The edges of objects will be nice and smooth, but without shaders, a RenderMan renderer knows only about the scene geometry and can't make anything look real, let alone photo-real.

How Shaders Work

Shading is often confused with texture mapping. Texture mapping is the process of applying a 2-D image to a 3-D surface, similar to applying a decal to a model. A "shader" is a historical term that originates from attempts to "shade" or color surfaces, typically polygons, with a computer procedure or algorithm. Gouraud and Phong shading are shading algorithms of this ilk. Surface shading with shading algorithms began in the early 1980's and, in a decade, shading has significantly evolved into sophisticated procedural shading languages. Pixar must be given credit for being the major developer of this technology through its introduction of RenderMan and the RenderMan Shading Language.

The RenderMan Shading Language is the key to understanding RenderMan's power. Other rendering systems, such as those included in software packages by Alias and Wavefront as well as Macintosh and PC based systems, have built into them the ability to make certain kinds of visual effects; users cannot add custom external effects (this is not to say that the user cannot import textures). Users of such closed rendering systems are restricted to only those kinds of effects that were thought of when the rendering software was written.

In RenderMan, shaders provide all the required visual effects. For every visual aspect of the scene, the surfaces, the lights, and the atmosphere, the renderer calls the shader program assigned to that entity to determine the effect. This allows the renderer to provide any kind of effect, not just what was initially implemented in the renderer. It also lets anyone write shaders because the shader programming language, the RenderMan Shading Language, is a standard part of RenderMan. So the capabilities of a RenderMan renderer are far greater than other rendering systems because RenderMan allows anyone to expand the renderer's capabilities - not just those who initially wrote the rendering software. RenderMan separates the shading processes from the rendering process - opening the system - so that creative minds can add their own special shader effects as they deem necessary.

Shaders in Conjunction with RenderMan Interface and Renderer

The renderer knows what shader to use because the RenderMan scene description contained in a RenderMan Interface Bytestream (RIB) file also contains RenderMan function calls that specify what shader to use and when to use it for any given purpose. In this way, when the time comes for the renderer to render any part of the scene, it knows what shaders to use and executes them.

As stated previously, the shaders themselves are written in the RenderMan Shading Language, a C-like language with special functionality built in just for its specific purpose. RenderMan defines several different categories of shaders depending on the intended use:

- Surface shaders are used to describe the appearances of the surfaces of objects.
- Displacement shaders are used to alter the surface geometry to add more detail without the need to model it explicitly.
- Light shaders define the way light sources behave.
- Volume shaders describe the way the atmosphere alters light passing through it.

There are other types of shaders defined by RenderMan but these are either not implemented or unavailable in the currently available renderers.

Once written, the shader code is compiled before it can be used by the renderer. The compiled shader works like a function called by the renderer. The renderer passes to the shader a certain set of values pertinent to the the particular point being rendered and the type of shader being called. This data represents the information needed to determine the shader's effect at a particular point in the image. The shader uses these values in its calculations to determine what values to return back to the renderer. The renderer then uses the values it gets back from the shaders in producing the image.

In addition to the values that the renderer passes to the shader there are other parameters that the shader gets directly from the RIB scene description file. These parameters are used to control the general behavior of the shader and are user determined or specified. For example, a light shader that implements a spotlight effect would need to know what direction the light was facing, how bright the light is, the size of the spot, and the light color. This sort of information stays constant throughout the generation of a given image. Parameters are specified with their values in the same RenderMan function call as the shader. The renderer then passes on this information along with the other rendering information to the shader when the shader is called.

How Users Access Shaders

To the users of RenderMan-compatible application programs the RIB file scene description is intentionally hidden, as hidden as Postscript is to users of a laser printer. Control of the shaders in these applications is handled through a shader interface. In many of the current software packages that use RenderMan-compatible renderers, the amount of information about shader parameters available to the application is adequate but minimal. A great shader interface is still a software design problem which, when elegantly solved, will allow users to examine, alter, and save shader parameter values so as to produce custom shader variations for use in their 3-D scenes. Discussion is underway for planned extensions of the RenderMan shader implementation. The ultimate goal is to enrich and standardize a methodology for interfacing to shaders.

Current Shader Access Issues

For the moment, all a modeling or rendering application can know about a shader is what the shader is called (the name used in a RIB file to specify that shader), the shader type, what its parameters are, their type, and their default values. All this information is contained in the compiled shader object code file or .slo file (.slo is the filename extension used for compiled shaders). An application may access or extract this information from a compiled shader through a set of library calls known as the sloargs library. Given the name of the shader, each sloargs function returns a different piece of information about that shader. An application may then display this information to the user and, hopefully via some interface, accept new parameter values as specified by the user.

Unfortunately, the amount of information currently available from the shader .slo files is very minimal. In particular, it is impossible for the shader to relay any bounds information to the renderer. Also, should limits be placed on a parameter's value, such that an appropriate slider control could be used to adjust the parameter, there is no method to exchange these data. Currently, it is impossible to specify a list of meaningful strings for string parameters. Moreover, help information regarding descriptions of what the parameters do, the name of the shader (not just the file name), as well as an example image of the shader would significantly enhance the user's ability to make effective use of shaders. These are but a few examples of information, that if provided, would significantly improve the usability of shaders and shader interfaces.

Issues for Future Access to Shaders

To address these and other deficiencies, new shader concepts are being discussed. The shader file will be extended to include a large variety of additional information about a shader, its parameters, and how they are to be used. With such additional information available to applications, the user interface to shaders will be greatly enhanced. As well as containing a greater amount of information, the future shaders will allow a shader instance. For example, all shaders come with a default set of parameter values, but Autodesk's AutoShade Version 2 product allows a user to customize a shader by providing the means to save a set of preset values for a particular shader. Hence, these presets are a form of shader instancing. In future, all users will be able to save their own parameter settings, in essence, creating their own version or instance of a shader.

Pixar's Showplace has developed a variation of a shader known as a "Look." Currently, Looks are user friendly alternatives to full featured shaders and they only work with a Librarian application in conjunction with Showplace. Still developing, the idea of a Look has some promise. What needs to happen is that the role of the Librarian needs to be expanded to allow access to all the parameters of the shader (not just a subset) and once a set of parameters is chosen, instances should be rendered for preview. If the desired appearance or Look is acceptable to the user it could then be saved as an instance and made available on demand to the modeling (or scene arranging) application.

With Windows, UNIX, or on the Macintosh, there is also the possibility of a separate Look browser application that could provide the complete user interface to Looks that would communicate with the primary application while running with it. This would provide a consistent Looks interface to users as well as minimize application development time. A universal shader interface with such features is yet to be determined.

In keeping with an effort to make RenderMan more popular and pervasive, Pixar is encouraging third-party developers, such as The VALIS Group, to develop and provide shader libraries. Although the theory of an open architecture renderer is sound, in practice, writing shaders for general purpose application requires esoteric and specialized skills. In the sections that follow, several issues related to providing quality shaders will be discussed.

Specific Shader Functionality Issues

While the Look paradigm addresses some of the usability issues surrounding RenderMan shaders, it doesn't address them all. If a shader is to be applied to a scene by an application's user, it must meet the basic requirements of general applicability and quality. The applicability issues are met by ensuring that all shaders provide a standard set of functionality and that shader parameters follow certain conventions. The quality issue is more difficult to specify and to implement. It deals with the need for shaders to operate properly over a wide range of conditions and accurately represent the intended effect.

Units

In order for a surface shader to produce an intended effect accurately, the units used in modeling the object to which it is applied must be known. Otherwise, the shader would have no way of knowing whether a square with sides of 1 unit long is 1 foot, 1 meter, 1 mile, 1 millimeter, or 1 whatever. So a surface shader designed to produce a tile floor using 1 foot square tiles would be lost without knowing the physical size of the unit square. Unfortunately, RenderMan currently has no mechanism for specifying or conveying this information to the shader by itself. To work around this, the modeling units used must be passed to all surface shaders as a string parameter that specifies which units to which the object is applied is modeled. This puts the burden of unit conversion onto the shader rather than with the renderer. Until RenderMan incorporates a means of tagging its coordinate systems with a corresponding system of units, it will remain the shader's responsibility to implement this functionality.

Projections

Surface shaders must be aware of the fact that the objects they are applied to are three dimensional entities. RenderMan allows shaders to take full advantage of the inherent three dimensionality of the objects they're applied to as well as the fact that certain materials are themselves three dimensional. Examples are stones of any kind, solid wood, and Styrofoam. However, there are a great many surface materials and effects that are inherently two dimensional. Examples of these are fabrics, tiles, wood veneers, and wall paper. In order to apply an inherently two dimensional effect onto a three dimensional surface, the shader must "project" the two dimensional effect into three dimensions. This can be accomplished by simply moving the flat material through space or by wrapping the flat material around a roughly spherical or cylindrical object. Shaders designed for general use must make available to the user a standard set of projections for surface shaders of inherently two dimensional effects. The type of projection is selected through a shader string parameter. Currently, the RenderMan Shading Language has no predefined projection function to implement standard projections. This is another bit of functionality the shader itself (and shader writer) must implement.

Anti-aliasing

Anti-aliasing is the primary quality issue the RenderMan renderers are designed to address and why they achieve such a dramatically photographic look. Anti-aliasing is simply the proper application of filtering to the image generation process to remove or suppress frequencies in the image that the output medium cannot accurately represent. Pixar's PhotoRealistic RenderMan compatible renderer can apply just this kind of filtering to the geometry of the scene, and to the resulting image. However, as was mentioned before, there is much more to a realistic scene than the geometry of objects.

The true realism of an image comes from the shaders used in the scene and on the objects. So the effects produced by shaders must also be anti-aliased if a high quality image is to be the result. Without anti-aliasing, shaders applied to objects that are either too small or too large in the image will produce very objectionable results. Inaccurate results commonly known as the jaggies, unwanted moire patterns, or the complimentary problem of loss of detail will appear in the final image. Also, as an object's surface curves away from the viewer or shrinks with perspective, these problems can occur.

Renderers are hard pressed to properly anti-alias the effects produced by shaders. The reason is that it's impossible for the renderer to have any information about what the shaders are doing to the surfaces of the objects and other scene elements. Shaders are independent programs that the renderer can only call to get a result. Having information about the specific frequency content of a shader effect is essential to properly filter that frequency content and produce a high quality, anti-aliased image.

Since the information regarding the specific frequency content of a shader effect is known within the shader (or more precisely, by the shader writer), the shader itself must perform any anti-aliasing filtering of the effect rather than the renderer. This requires of the shader writer an intimate understanding of all the possible frequency sources in the shader code and of how to properly filter those sources given the sample size the renderer is using. Understanding of the rendering process, the values the renderer uses, and those it provides to the shader is critical to this process as well as knowledge of filtering, sampling theory, and the frequency content of the shader code being written. A shader writer must know exactly what detail to represent in the shader and how to smoothly transition from the most detailed view of the shader's effect to the most gross overview. The Shading Language does not provide any real set of filtering functions suitable for this task so these must also be implemented by the shader writer.

Proper anti-aliasing of the shader is the most important and difficult aspect of shader writing. While almost any shader will work under a specific range of conditions, it will certainly fail in general use if it is not properly anti-aliased. This is an essential element of any shader written for general application to images by end users. The renderer itself cannot deal with this issue since it can't be made aware of all the complexity contained within a shader. Any attempt to make shader anti-aliasing automatic in the renderer would, of necessity, be far too general, resulting in very long rendering times. But without shader anti-aliasing, the high quality and realism of RenderMan images is almost completely lost due to the artifacts caused by shader aliasing.

Conclusion

The RenderMan Interface Specification is designed to exist at the boundary between modeling and rendering to separate those activities and organize the way information is exchanged between those activities. The Shading Language was developed to allow maximum flexibility in 3-D computer graphic scenes. Shaders are written outside of the renderer to control surface appearance, lighting, and atmospheric effects. RenderMan-compatible renderers rely on shaders to add visual quality to synthetic imagery.

Shaders are procedural and users access the appearance characteristics of a shader through parameter lists. Currently, the interface to shaders is minimal. The concept of a Look has been introduced to improve access and interface to shaders as well as to allow for shader instancing. A shader instance is a customized parameter set defined by a user that can be saved and used again.

While interface issues are important for accessing the power of existing shaders, functionality issues are important to development of shaders and are harder to address. The three most critical functionality issues are units, projection, and anti-aliasing. Shader developers must contend with these issues in order to make shaders that have general application: that do the right thing under a variety of conditions. These issues are not easily generalizable and make writing marketable shaders a difficult task.

At present there are over thirty 3-D design and application products that output RenderMan Interface Bytestream (RIB) files on the market. In order to make rendering and RenderMan more pervasive these products need quality shaders and improved shader interfaces. The VALIS Group is a shader development and marketing organization established to take advantage of the open architecture proposed by the RenderMan Interface Specification. As additional RenderMan-compatible renderers appear, the demand for shaders will most certainly increase. Shaders allow us to add variety and interest in photorealistic images. For the more creative individuals, shaders allow 3-D imagery to go into new directions beyond realism.

References

1. The RenderMan Interface. Version 3.1 September, 1989. Pixar. 1001 W. Cutting Blvd. Richmond, CA. 98404.

2. Cook, Carpenter, Catmull. "The Reyes Image Rendering Algorithm." 1987 SIGGRAPH Proceedings. Computer Graphics. Volume 21, Number 4. ACM SIGGRAPH.

AUTHOR INDEX