PROCEEDINGS

5th symposium on small computers in the arts

OCTOBER 5-8, 1985

PHILADELPHIA, PENNSYLVANIA

SPONSORED BY:

IEEE Computer Society
Delaware Valley Chapter ACM/SIGGRAPH
Small Computers in the Arts Network
and in cooperation with ACM/SIGGRAPH

ISBN 0-8186-0689-4
IEEE CATALOG NO. 85CH2218-6
LIBRARY OF CONGRESS NO. 85-62328
IEEE COMPUTER SOCIETY ORDER NO. 689

IEEE COMPUTER SOCIETY

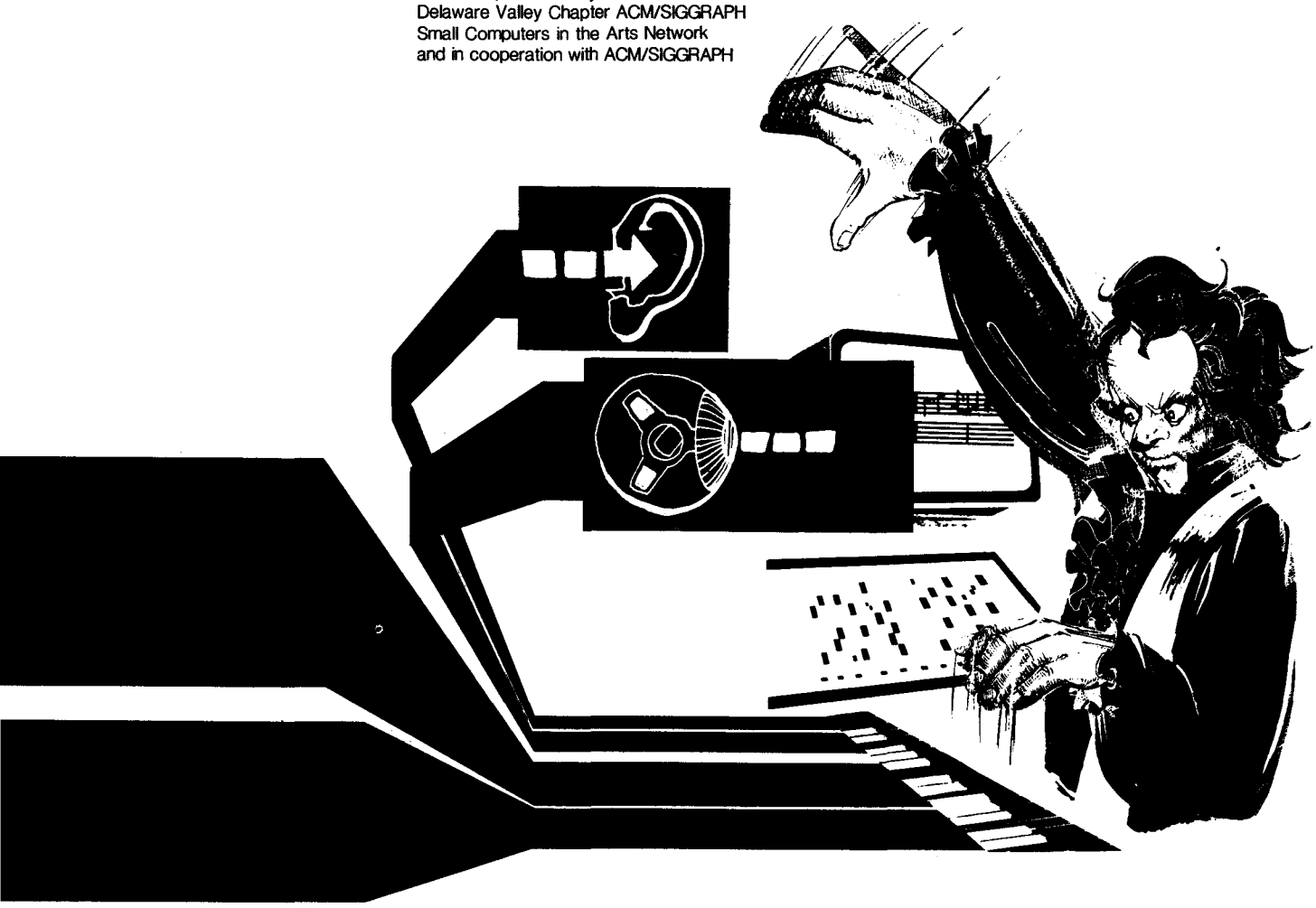
IEEE COMPUTER SOCIETY PRESS

IEEE
THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

PROCEEDINGS
5th symposium on small computers in the arts

OCTOBER 5-8, 1985
PHILADELPHIA, PENNSYLVANIA

SPONSORED BY:
IEEE Computer Society
Delaware Valley Chapter ACM/SIGGRAPH
Small Computers in the Arts Network
and in cooperation with ACM/SIGGRAPH



ISBN 0-8186-0689-4
IEEE CATALOG NO. 85CH2218-6
LIBRARY OF CONGRESS NO. 85-62328
IEEE COMPUTER SOCIETY ORDER NO. 689

IEEE COMPUTER SOCIETY
COMPUTER SOCIETY PRESS
IEEE
THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

Published by IEEE Computer Society Press
1730 Massachusetts Avenue, N.W.
Washington, D.C. 20036-1903

IEEE Catalog Number 85CH2218-6
Library of Congress Number 85-62328
IEEE Computer Society Order Number 689
ISBN 0-8186-0689-4 (Paper)
ISBN 0-8186-4689-6 (Microfiche)
ISBN 0-8186-8689-8 (Casebound)

Order from: IEEE Computer Society
Post Office Box 80452
Worldway Postal Center
Los Angeles, CA 90080

IEEE Service Center
445 Hoes Lane
Piscataway, NJ 08854

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

1985
Proceedings of the
Fifth Annual
Symposium on
Small Computers
in the Arts

October 5-8, 1985
Philadelphia

Sponsored by:
IEEE Computer Society
Small Computers in the Arts Network
Delaware Valley Chapter ACM/SIGGRAPH
and in cooperation with ACM/SIGGRAPH

Organized and Produced by:
Small Computers in the Arts Network

Symposium Staff

Symposium Chairperson:	Cathy Del Tito
Proceedings Editor:	Marc Scott
Organizing Committee:	Eric Podietz
	Bill Mauchly
	John Senior
	Alan Datri
	Tom Rudolph
	Tom Porett
	Dick Moberg
	Harvey Weinreich
	Trip Denton
	Ann Hitchins
	Ben Gollotti
	Maurice Wright
	Myra Messing
	Bob Grenetz
	John Blackford

And help from many other volunteers.

INTRODUCTION

We would like to introduce these Proceedings of the Fifth Annual Symposium on Small Computers in the Arts by relating just a bit of the history of the group that organizes it, the Small Computers in the Arts Network, Inc. This group is committed to the promotion of small in the arts through its newsletter, SCAN, concerts, the Symposium, and various informal meetings.

This current group emerged out of an earlier one, the Personal Computer Arts Group which held its first computer arts activity, a computer music concert, in 1978. The group hoped to provide a forum for creative people interested in using computers in the arts.

Art and music created with the aid of computer technology is more affordable than ever and the results become more "aesthetic" as every year goes by. As creative equipment and software tools become more mainstream, the commercial, fine arts, and music commmunities are bound to feel the positive impact. This Proceedings reflects the universal position of computers in the creative arts today.

Articles this year include trials and triumphs in the areas of aesthetic judgement, teaching, and evaluating artistic equipment as well as reports on project, products, conferences, networks, and expereiences.

The range of interests represented here and the degree to which they have been developoed by the authors shows us as organizers of the Symposium that a forum where computer artists can share knowledge with their colleagues is as worthy a project today, if not more so, than five years ago. We hope you are pleased with our selection of works.

Table of Contents

Symposium Staff..... v

Introduction.....vii

Artists' Issues at SIGGRAPH: Aesthetics, Access and Interfaces..... 1
W.S. Moldauer

Electronic Media Laboratory Proposal..... 3
W.J. Kolomyjec

Effective Visual Representation of Computer Generated Images..... 9
M. Holynski and E. Lewis

Art, Microcomputers and the Mona Lisa..... 13
G.K. Shortess

Image Enhancement Techniques for Microcomputer Art..... 19
C. Glassmire

Philosophy and Capability Issues in Computer Graphics Software Evaluation: What Do You Mean I Can't ...? 29
C. Spiaggia and C. Beebe

Teaching "C" Programming to Artists..... 39
D.M. Palyka

Digital Textile Construction: Using LOGO..... 47
M.M. Ozmon

Computer-Aided Printmaking..... 57
I.V. Kerlow

Computers Viewing Artists at Work..... 65
J.L. Kirsch and R.A. Kirsch

Music and Data Structures: The Application of Music Theory in Programming Computer Assisted Instruction..... 69
G.S. Karpinski

The Yamaha DX-7 Synthesizer: A New Tool for Teachers..... 73
D. Righter and R. Mercuri

Applications for Computers in the Arts..... 77
D.A. Butler

New Page: The Changing Face of Composition..... 91
J.K. Shafran

New Aesthetics for Musical Variation..... 9
S. Berkowitz

Paint Programs: An Artist's Evaluation and Wishes.....10
T. Denton

Conceptual Guidelines for Choosing a Graphic Workstation.....10
T. Porett

Algorithmic Composition.....11
R. Kram

Aesthetics and Computer Graphics.....12
W. Wright

Author Index.....13

ARTISTS' ISSUES AT SIGGRAPH: AESTHETICS, ACCESS AND INTERFACES

Wendy S. Moldauer

There are several important issues facing artists working with computers. As the primary organization addressing the computer graphics community, SIGGRAPH has shown a high degree of responsiveness to the needs of artists. In 1985, courses and panels engaging a wide variety of speakers on the subjects of user interface issues and aesthetics were well attended. The next step is to involve the artistic community even more, by developing a new format for discussion and evaluation of these issues as they relate to the artist/user.

AESTHETICS

There seem to be a couple of theatres in which the aesthetics issue is being played out. The first is internal, a discussion between artists already familiar with the medium. The second involves dialog between computer graphics artists and the art community at large.

An indication of the gelling of aesthetics as a credible issue within the computer graphics community was the panel at SIGGRAPH '85, chaired by Mihai Nadin, of the Rhode Island School of Design with spirited discussion among Naidin and panelists Chuck Csuri, Frank Dietrich, Hiroshi Kawano and Tom Linehan. Those attending the panel were more than ready to participate--both of the microphones provided for questions sported long lines, and most of the questions offered generated more debate than could be covered in the time allotted to a panel.

1985 also saw a new format for the presentation of visual work. Along with the traditional art show and film and video evenings, a separate screening room was set up to provide an overview of works from a variety of areas.

The art show itself met with little criticism. It comprised several elements: a gallery-type space with two-dimensional work and sculpture; several installation areas for stereoscopic pieces, light sculptures and interactive environments; a Mackintosh environment for displaying many artists' work; and screening rooms for new high-resolution images and animation.

An innovation this year was the screening of film and video pieces old and new, which made up a survey of past, present and near-future developments in computer graphics. There was an impressive overview of work, creative and otherwise, divided into categories such as "Classics," "Education," "Art" and "Demo." Frankly, the telling segment was "Classics." It was impossible to deny the power of some very early work, such as that of John Whitney, Sr. and Peter Foldes. Other, more technical pieces also showed grace and humor, sometimes even truly inspired use of the computer, but even the "Art" segment was sometimes a bit stale.

As for the film and video presentation, there continues to be, though in smaller quantities, redundant "glitz" in the form of gratuitous streaks and flashes, swooping and diving cameras, all six million available colors co-displaying less than peacefully. Available resolution goes up exponentially every year, but it does not appear that artists are getting much more input regarding its effective use. Where is the gap? Why is the glitz being adopted so willingly by artists, when there are endless opportunities to explore unimagined possibilities? Is the "art" in computer graphics being guided mainly by technical wizards? Or, have the artists who gained access to commercial systems lost their device independence?

A type of chauvinism etches a great many discussions of computer art. It is characterized by heated debate over computing power, available resolution and colors, display and output technologies, software vs programming. Often this misses the point: does the piece work? The agony of creation is not enough. Facts of resolution, output device, unavailablity of desirable technology are, for the most part, irrelevant. The reaction of the art community as a whole has little to do with technological considerations. Unfortunately, the world at large has to make do with little more than a slow trickle of pieces from the computerized community, and often these are pieces in which there was very little input from artists.

CH2218-6/85/0000/0001$01.00 © 1985 IEEE

ACCESS

Is access an issue? Well, yes and no. A persistent individual who is willing to be flexible can usually find a computer to work with. The available tools, resolution and work environment may not be optimal, but there are systems out there. Adequate distribution is a far more persistent headache.

Still, there are many problems with this entrepreneural system of access. First of all, the artist must usually make do with whatever is available, without much hope of influencing the machine's actual usefulness. That is unfortunate, because it perpetuates existing inadequacies. Also, when the interface is complex, it may require more time than is available to learn ones' way around it. The artist must be willing to negotiate blocks like awkward time schedules, inadequate training and support, learning more than s/he ever intended to about hardware, operating systems, the hazards of inelegant programming.

So what are art schools doing? Good computer graphics departments are still hard to find, and often even harder to get into. One option is to become part of a fledgling program, which often means a lot of pioneering and diplomacy, and a little time to make art.

Another consideration is that some programs combining art and computers, such as Ohio State's, involve the student in a great deal of programming and development, areas which do not appeal to all artists, while others, such as the undergraduate program at Rochester Institute of Technology, put emphasis on using existing turnkey systems--finding ways of getting around the limitations instead of trying to optimize them through redesign. These are clearly very different approaches to the role of the artist in development of his/her own computing environment. Though the situation is improving, it still seems that very few artists are making a direct impact on the computer graphics systems available to artists.

INTERFACE

For those who feel that the interface is not a central issue, it is especially important to talk to artists who have been turned off by their first encounter with a computer graphics system. Common frustrations are the inflexibility of the tools; the complexity of their use; a general unwieldiness which often requires incredible amounts of time in order to simply mimic traditional media. Another curse is the the incompatible nature of not only operations but also language--sometimes it seems that each system's designer decides to reinvent the languages of both computers and art.

In the discussion of aesthetics is some potential guidance in interface design. In working to develop long range visions of the potential of computers to enable, facilitate, perhaps even assist in the creative process, we will learn what is needed to bring the machine the rest of the way into the studio.

An exploration of aesthetics in computer graphics needs to be continuous, and interconnected with dialogue relating to all art forms. The Symposium is one of many suitable forums, perhaps better suited than most due to its flexibility. It is up to each individual involved in computer graphics to urge that his/her community begin to consider aesthetics along with technical wizardry.

Electronic Media Laboratory Proposal

by
William J. Kolomyjec, PhD, MFA
Associate Professor of Art
College of Visual and Performing Arts
Northern Illinois University
DeKalb, Illinois 60115

Abstract

This paper addresses issues relevant to coordinating a college-wide effort to provide access to media and technology in the areas of visual art, music and theatre. At the heart of this strategy is a design for an Electronic Media Laboratory consisting primarilly of a network of small computers. Using enhanced and off-the-shelf microcomputers is the first phase of a plan to achieve state-of-the art capabilities.

Introduction

This paper represents a strategy for facilitating the instruction of media and technology at the post secondary level. Such a project is taking place in the College of Visual and Performing Arts, Northern Illinois University. NIU already has a microcomputer facility of approximately 20 Apple II class machines based in the Department of Art, Visual Communication and Design area. The challenge presented to the author is to coordinate an effort to go beyond this dated technology. Moreover, to develop a college-wide facility open to all the Visual and Performing Arts faculty and students.

The term "media and technology" will be used in a visual context to mean imagery or objects that are produced using electronic technology. Very loosely this includes film. More appropriately it means; slide and movie projection systems, equipment to control slide and movie projection systems, analog and digital video, analog and digital video editing and projection equipment, computer graphics hardware and software, computer image editing and projection systems, and any combination or permutation of these. (Media and technology in the context of music, theatre or dance would be slightly different.) Hence the term Electronic Media has a broader connotation, it represents the application of media and technology to the Arts.

Electronic Media Laboratory (EML)

Recently, this author had the pleasure of working in a state-of-the-art environment called the Computer Graphics Research Group (CGRG) at The Ohio State University, Columbus, while on the faculty of the Department of Art Education. It is his opinion that state-of-the-art technology is centered around a UNIX operating system/ C programming environment and that this operating system/ programming environment is the model for advanced work in the computer arts. The CGRG effort is geared toward three dimensional computer graphics animation and their facilities reflect this effort[1]. Our intention is broader-based. This proposal represents the first phase of a two phase plan we are implementing to do advanced work in Electronic Media in the Arts. It represents a reasonable transition between where we presently are and the direction in which we intend to proceed.

Figure 1 schematically represents the Electronic Media Laboratory (EML) that we have begun to establish as a College facility in the College of Visual and Performing Arts, at Northern Illinois University. In the conceptual phase of the laboratory design there were three major considerations: 1) To achieve a level at or near state-of-the-art technology. 2) To maximize versatility. And, 3) To be as cost effective as possible. We believe our configuration succeeds in reflecting these considerations.

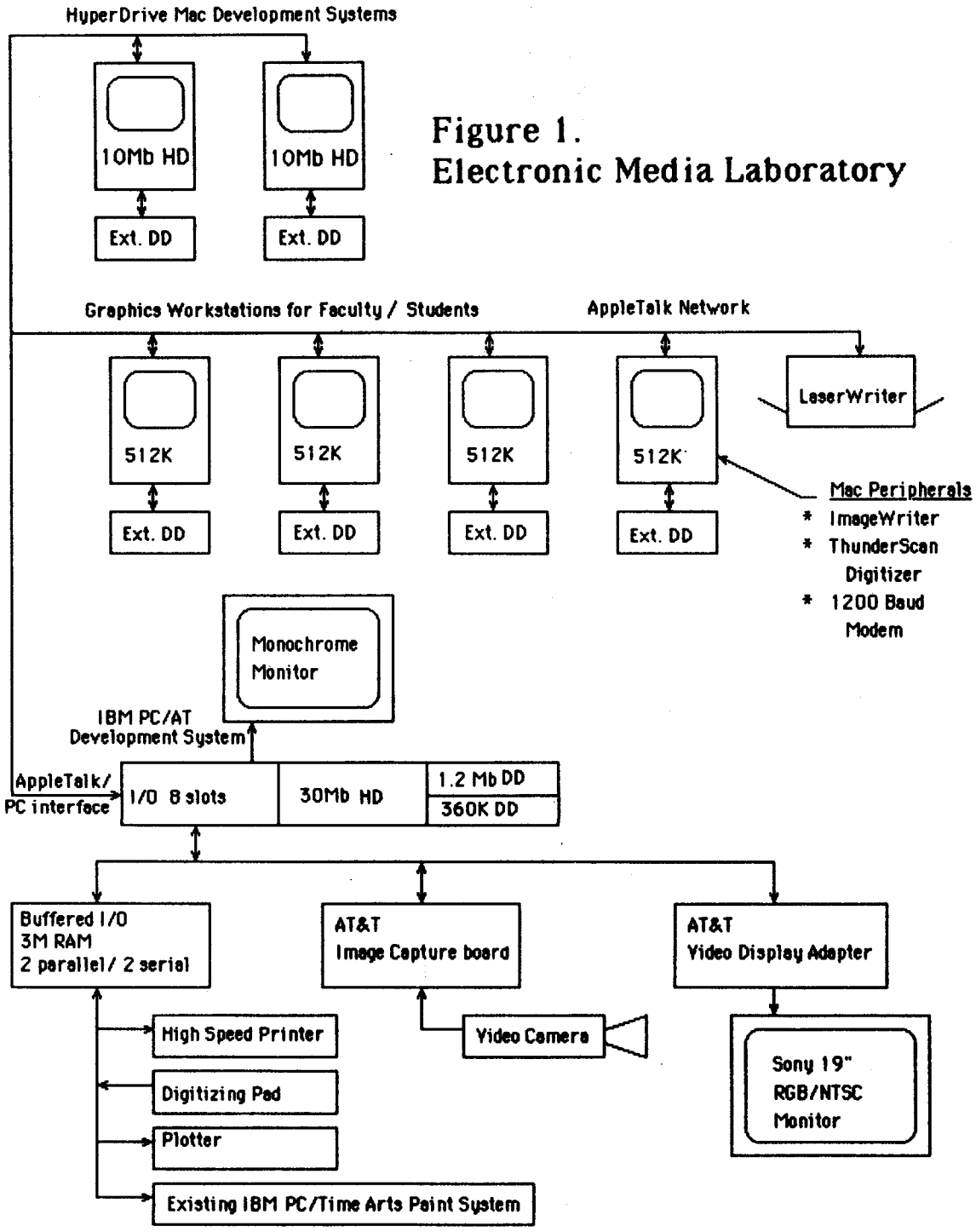


Figure 1.
Electronic Media Laboratory

A Two Phase Plan

In phase one, we have chosen to work with both Apple Macintosh and IBM PC/AT microcomputers and appropriate software. By themselves each machine is extremely versatile. However, by taking both enhanced and off-the-shelf versions of these machines and integrating them into a local area network (LAN) using AppleTalk we feel we can provide a more powerful environment with greater capabilities than individual workstations. With this strategy we can also achieve cost

effectiveness. We will not be using exotic hardware and software. Necessary components, such as AT&T frame buffer and frame grabber, will plug directly into the IBM PC/AT (or other devices into the Apple Macintosh) all of which can be accessed through the AppleTalk LAN.

This initial phase will allow us to begin to explore and develop technology-based applications in the Visual and Performing Arts, to support a multitude of advanced aesthetic applications for the entire college. Pragmatically, the emphasis will be placed on exploring computer graphics and media and technology applications in the Visual Communications and Design area. In this respect the Electronic Media Laboratory would seem to be a logical extension of the existing teaching facility. However, we feel strongly that the Electronic Media Laboratory has potential to facilitate research and development, i.e., a graduate program.

Administratively, the Laboratory is attached to the college Dean's office. This is logical, practical and advantageous. Structured in this manner it will allow any faculty person or student in Visual Arts as well as, Music, Theatre or Dance to have access to the facility.

When we begin to demonstrate what is possible with the resources we have obtained in the first phase of this plan then we will have a foundation to discuss and justify phase two. Phase two is imagined to be comprised of hardware for advanced video, electronic music composition and several graphics workstation configurations: minicomputer or mainframe system, very large disk storage capacity, 24 bit frame buffer and very high resolution display capabilities. A true UNIX/ C environment. We are just beginning the first phase of our plan. When phase one is completed, phase two will be an easy transition.

Strategy for Implementation

The time frame for developing phase one is predicted to be one year. Completing phase one, establishing the EML, is our first major goal. As such it can be divided into a series of objectives, called short, medium and long term objectives. Of course, each objective can be subdivided as well. Broadly stated, these are given below:

Short term objectives.

Acquire key personnel. It's people that make programs not vice versa.

Acquire the hardware and software required by the design.

Check out each component of the system. Check out each individual system. Fill in the warrenty cards and send them in.

Learn how to thoroughly use each system and subsystem. This includes the physical operation of the system, the operating system and the programming languages used in conjunction with the hardware.

Acquire and learn how to use other relevant application software.

Begin development work on the enhanced systems.

Begin digitizing imagery using scanners and digital video equipment.

Develop a file handling strategy.

Medium term objectives.

Network the Macintosh's and LaserWriter with AppleTalk.

Network the IBM PC/AT to AppleTalk.

Establish a network controller using either the PC/AT or the enhanced Macintosh as a fileserver. Begin transferring ASCII files between devices.

Develop and implement data generation, data manipulation, and data display algorithms.

Long term objectives.

Develop and implement an optimal method of information interchange within the network.

Begin transferring binary files (imagery) between devices.

Make each peripheral, especially the frame buffer, a device on the network, available from any CPU.

General Discussion of Components

Without going into specifics it may be appropriate to discuss the components of the EML configuration represented by Figure 1. These components can be grouped by component and type, i.e., hardware or software.

Apple related hardware.

Six Macintosh workstations have been purchased. Four 512K "Fat" Mac's will be designated as faculty/ student workstations. Two HyperDrive Mac development systems have been put together from 128K machines upgraded to 512K when 10Mbyte hard drives were installed. Each system will have an external drive. All six systems have an Apple ImageWriter assigned to them; four will be used for output, two will have ThunderScan scanner devices adapted to them for image input capability. Two 1200 baud Apple Modems have been aquired for telecommunication purposes. An Apple LaserWriter will be networked into this configuration via AppleTalk. The LaserWriter will produce high quality output that can be used directly by design students in their traditional assignments. Note: This hardware was obtained at significantly reduced prices through the Illinois Educational Consortium.

Apple related software.

The Macintosh systems come with MacWrite and MacPaint. From Apple we have or will obtain MacTerminal and MacDraw. From Microsoft we have obtained the powerful BASIC interpreter Basic 2.0. For advanced work two C compilers have been obtained: Megamax C and Manx C, the latter having a near UNIX shell. Other Macintosh software has been obtained and more is becoming available on a daily basis.

IBM related hardware.

A major component in our scheme is the IBM PC/AT. We began with a stripped chassis model with a 1.2 Mbyte and 360K disk drive and installed 512K RAM and a 30Mbyte Maynard Hard drive. Input/Output will be directed through an AST Advantage board with the Advantage-pac and a full complement of RAM. A monochrome monitor is assigned to this machine and a mouse was added.

IBM related software.

From IBM we obtained DOS 3.1, Topview End User and Topview Tool Kit. However, the majority of the IBM development software was obtained from Lifeboat, namely a Lattice C compiler, libraries, utilities and Multi Halo graphics.

AT&T hardware.

Due to a technological breakthrough AT&T has marketed two affordable products of particular interest to us; two boards which plug directly into the PC/AT known as the Image Capture Board (ICB) and the Video Display Adapter (VDA.) These two boards are essentially a video frame grabber and a 256 by 256 by 8 bit frame buffer. Both RGB and NTSC video are supported by these devices. Software for the IBM comes with these boards. Both boards are very reasonably priced. With these devices we will have digital video imaging capabilities.

Video.

Video is a major component in the larger scheme of media and technology. Our second major goal is to develop the video component along with the computer imaging effort from the onset. The rudaments of a video program are in place at NIU. Ideally, we can direct our efforts to the point where video and computer become unified into a single program.

To demonstrate our commitment to this end we have used a significant proportion of our resources to purchase a 3/4" Sony U-Matic Video Editing System for the Electronic Media Design Laboratory. This system consists of 2 video decks, 1 editing controller, 2 RGB monitors and all cables. By placing this equipment under the umbrella of the Laboratory we have provided a valuable resource within reach of all faculty and students in the college. In return we have gained access to existing video equipment and, more importantly, gained the support of existing faculty in this area.

Miscellaneous peripherals.

Video will interface to the laboratory network via the AT&T devices and the IBM PC/AT. To display video output a Sony RGB Direct Drive 19" Color monitor has been purchased (Model PVM1910.) Presently imagery can be filmed directly off the CRT screens using 16 or 35mm film. This is not desirable. We will be looking at ways of interfacing existing hardcopy devices and an existing filmrecorder into the system.

Other peripherals that have been purchased are: a high speed dot matrix printer and quality digitizing pad for the IBM PC/AT. Also, an IBM PC/ Time Arts Lumena Paints system exists in the college. An attempt may be made to interface this existing paint system to the EML configuration.

Other important considerations.

We are looking at the present curriculum in our Comprehensive Design, Electronic Imaging program (a BFA Studio Art Emphasis) and suggesting the modification of existing courses or the establishment of new courses to make the best use of the EML facility. We must address the issue of the appropriateness of this curriculum in a single area or as a separate area. We are also looking at areas within the college that use technology outside of the Visual Arts to determine how these areas might best be served by our effort. Our third and ultimate major goal is to integrate the individual efforts of media and technology within the Visual and Performing Arts into a single unified effort.

Conclusion

Media and technology is affecting all facets of the Arts. In both performance and recording electronic media is playing an ever increasing role. Moreover, imagery and sound have been reduced to the common denominator of zero's and one's. This requires those who work in Arts and modern media to be familiar with binary devices and this means computers. Electronic media issues must be addressed by institutions who proport to provide instruction in these areas.

State-of-the-art facilities are a major institutional investment that require ongoing expenditures[1]. Thus, they will always be few and far between. Small computers are a reasonable alternative. At Northern Illinois University we feel that with the increasing capabilities of small computers we can deliver relevant and contemporary instruction in the Arts. In this spirit we have proposed the Electronic Media Laboratory and we have begun to work on making it a reality.

References

[1] Kolomyjec, W.J. "Animating at Ohio State: Thoughts on a Graduate Program in Computer Graphics", Small Computers in the Arts Symposium. Proceedings of the 4th Annual Meeting. Philadelphia, PA. 1984.

EFFECTIVE VISUAL REPRESENTATION OF COMPUTER GENERATED IMAGES

Marek Holynski and Elaine Lewis

Boston University

Abstract

This paper presents an approach for incorporating aesthetic criteria into a computer graphics system to yield a valuable tool for artists. Through empirical evaluation design standards are discovered. In this experiment, seven picture variables were used to generate a series of visual stimuli. Testing revealed optimal levels variables in terms of viewer preference.

Introduction

There is an almost infinite number of ways that given picture elements can be visualized on a computer screen. Only a few of these, however, can suitably represent a specific meaning within the desired aesthetic criteria of a particular artist. Selection of the appropriate representation from a large number of all possible representations can be cumbersome for human viewers. The burden of this task, a chore shared by the artists of more traditional media, can be lightened through the development of formal criteria for effectiveness and aesthetic quality. We argue that this task should be performed by the computer where decisions are based on aesthetic criteria given in the form of algorithms. To develop a computer system incorporating such aesthetic criteria we propose:

- to discover the optimal levels of picture variables through testing a series of computer images created with a menu-driven program,
- to apply advanced machine learning methods (inductive learning techniques) for the development of criteria for perceptual judgement.

The criteria developed can guide the computer graphics system in determining the perceptually optimal graphical representation of displayed picture variables. These variables include number, size, position of picture elements and other factors such as rotational or perspective transformations, color, complexity, variety, and regularity. We correlated these with perceptual and cognitive factors such as attention and preference. By determining the most effective visual formats for specific purposes and certain classes of individuals, and integrating these with artificial intelligence techniques, we are able to analyze, create, and modify graphics from the standpoint of contextual understanding.

Evaluation Techniques

An intelligent graphics system should contain empirically determined standards used for default values and rules that reflect users' preferences. In more technical areas, some of these display standards have already been defined. For instance, in many CAD/CAM systems, principles of representing an engineering drawing are well established and these production rules are used to guide the system. In order to apply these principles for more artistic purposes, we need to define them in a more generally applicable way. First we must define relevant visual variables which can be used for evaluating user reactions. Then we will have to test these variables for viewer response in order to find optimal levels and establish standards. Finally, these standards can be integrated into the graphics system.

Previous work has begun to address this area by combining traditional design principles with computer generated imaging techniques to yield descriptive variables like regularity, complexity, order, and balance (1,2,3). In some cases, these variables were found to significantly predict over sixty percent of the variance in preference (4,5).

Modular Patterns with Design Variables

In order to explore more sophisticated variables relevant to techniques for generation of computer graphics, we illustrated image form variables through a series of pictures created with a menu-driven program and a graphics tablet.

Design variables such as complexity, regularity, symmetry, balance, variety, busyness, and color were illustrated through a set of sixty-eight stimuli. Stimuli were generated as abstract display matrices comprised of controlled combinations of thirty-six basic elements.

The elements are shown in Figure 1. All elements have specific values of both complexity and regularity. There are three levels for each variable. Complexity is defined as the number of curves or line segments outlining each shape. Regularity refers to local symmetry where very symmetrical shapes have high regularity and asymmetrical shapes have low regularity. For example, a circle has low complexity and high regularity whereas an elaborate shape with asymmetric curves would have high complexity and low regularity.

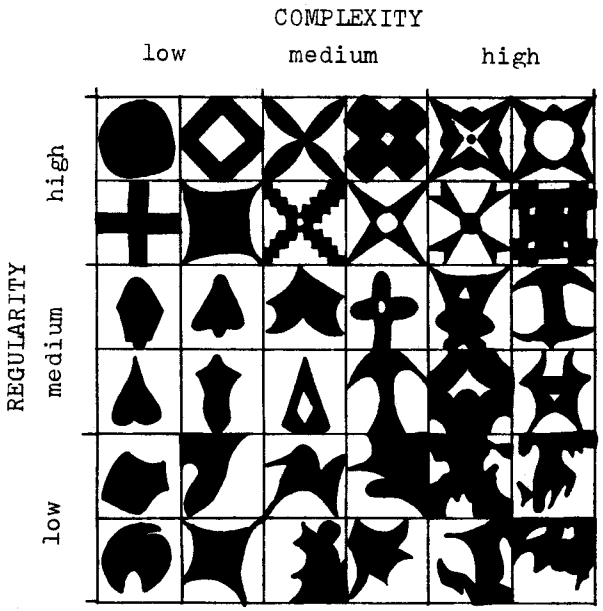


Figure 1: Basic Elements for Stimulus Patterns

Transformations of these original elements produced the other variables. Symmetry considers the kind of reflection that was used to create the display matrix. Pictures with low symmetry have no reflection, those with medium symmetry have reflection along one axis, those with high symmetry have reflection along two axes.

Busyness is the number of objects in each quadrant, low busyness shows one or two object in each quadrant, medium has three, and high busyness has four objects in the quadrant. Variety considers the number of different kinds of elements used to produce the stimuli. In low variety the entire matrix is made from the same type of basic element. Medium variety has two kinds of shapes, and high variety has three or four different types of shapes.

Balance deals with the level of rotation of the quadrant used to produce the display matrix. Low balance has random placement of elements. Medium balance shows incremental rotation where the quadrant is rotated ninety degrees for each placement. High balance is a straightforward translation pattern. Examples of research stimuli can be seen in Figures 2 and 3.

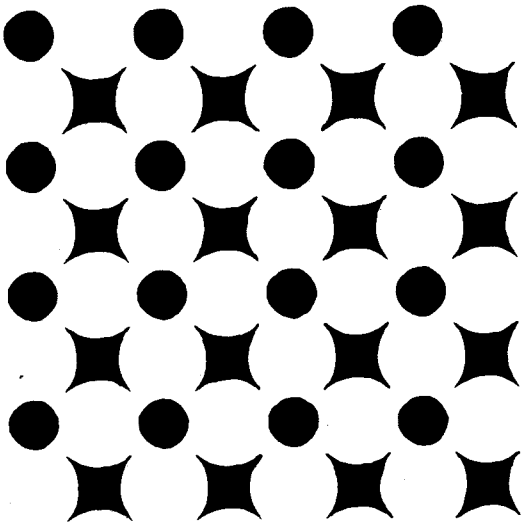


Figure 2: High Balance; Medium Busyness, Regularity, and Variety; Low Complexity and Symmetry

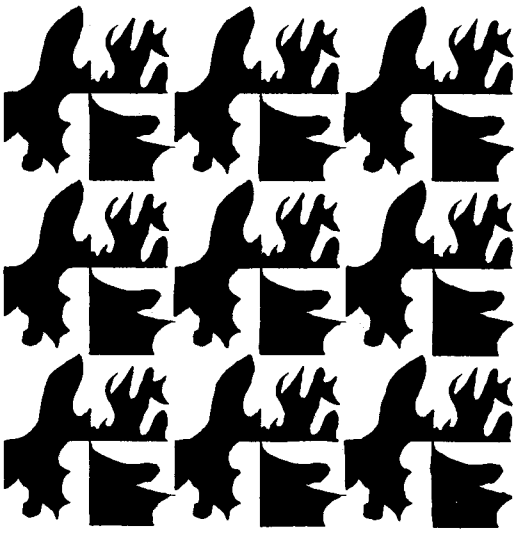


Figure 3: High Balance, Busyness, and Variety; Medium Complexity; Low Regularity and Symmetry

Analysis and Results

Eighty subjects from three different student groups were shown sixty-eight slides and asked to evaluate them using a 100 point preference scale. Average values for each slide were calculated and analysis of variance revealed no significant differences according to group or sex. Average scores were regressed on a set of dummy variables which represented the stimulus levels for each slide.

When all the variables (15 dummies-seven conceptual variables) were entered into the regression equation, the R-square value was 0.788, but few of the individual variables showed a significant effect. This combination of high overall significance but little individual significance indicates a substantial multicollinearity among variables. This was due to the unfortunate fact that the experimental design lacked balance for all the conceptual variables. After careful examination of these results, the researcher identified variables which seemed to contribute most to the collinearity. These variables, variety, balance, and regularity, were deleted and the model was reanalyzed. This model is indicated in the table below by the "Reduced Model" column.

	FULL MODEL		REDUCED MODEL	
	b	F	b	F
Complexity				
Medium	7.66	6.30	8.49	23.72
High	6.81	1.07	6.84	12.57
Symmetry				
Medium	-0.50	0.01	-4.26	6.47
High	7.41	1.53	8.66	23.67
Balance				
Medium	-1.16	0.19		
High	-2.73	2.45		
Variety				
Medium	-0.44	0.01		
High	-5.58	1.08		
Regularity				
Medium	3.49	1.82		
High	4.00	0.43		
Busyness				
Medium	-2.07	0.17		
High	4.36	1.08		
Color				
Blue	9.82	34.05	9.76	40.25
Red	4.60	7.32	4.41	8.07
Green	0.12	0.01		
Constant	31.44		35.05	
R-square	0.788		0.722	

As the table shows, complexity increased the mean preference score by about seven points on the 100 point scale. The effects of medium vs. high symmetry were strikingly different. High symmetry accounted for about a nine point increase, while medium symmetry actually reduced the average preference level by about four points. Colors also showed a significant effect with blue accounting for an average gain in preference of eight points and red showing an effect of about half that value. R-square for the reduced model was 0.722. The constant term represents the mean score for slides that were white or green in color and low on complexity and symmetry.

While these findings are quite interesting, especially in terms of the predictive effect of symmetry and color, problems with the research design and the small number of observations limit practical inference. The high R-square suggests the worth of this general approach for defining visual information.

Further research

Future studies should have more balanced designs so that variable effects are not collinear. This will allow more direct measurement on the effect of each of the individual variables on preference. More precise quantification of the independent variables would also improve the design. Continuous values would eliminate the need for dummy values and they would allow more sophisticated modelling techniques.

We also plan to define and explore three dimensional layout variables for number, distance, and angle of viewpoint. These will relate to solid modelling techniques for computer imagery. In a similar manner, 3D variables can be manipulated to generate image stimuli differing in various aspects of depth depiction. Previous research has identified as important for display efficiency the distance of the viewpoint ("camera"), its height and angle, the number of viewpoints and the size of the visual angle it subtends (7,8,9).

The precise visual characteristics of these image stimuli will be quantified using an automatic rule acquisition program (10,11). The rule acquisition program will interact directly with the graphic generation routines and continually refine its rules as new viewer variables are optimized. Using this information interactively the system will generate images which more accurately measure the user's likes and dislikes. The system will then run through a series of images choosing those which are most effective.

Future theoretical work should more fully justify the choice of variables. Studies where the form variables are used to structure content would allow the use of comprehension and other more complex measures of viewer reaction.

References

1. Marek Holynski, Art and Computers, Warsaw, Wiedza Powszechna, 1976.

2. Marek Holynski and Elaine Lewis, "Experimental Visual Evaluation for Computer Graphics", IEEE Proceedings, 3rd Symposium on Small Computers in the Arts, 1983, pp. 21-24.

3. Marek Holynski and Elaine Lewis, "Effectivness Standards for Computer Graphics", IEEE Proceedings, 4th Symposium on Small Computers in the Arts, 1984, pp. 23-28.

4. Elaine Lewis, "An Effectiveness Measure for Visual Communication Media: Toward Definition of Visual Principles", Doctorial Dissertation, Department of Language, Literature and Communication, Rensselaer Polytechnic Institute, 1981.

5. Elaine Lewis and Brian Keith, "The Addition of Content and the Consistency of Preference Ratings for Visual Structures", paper presented at the 1983 annual meeting of the International Communication Association.

6. Marek Holynski, Intelligent Machines: Computer Graphics and Vision, Warsaw, Iskry, 1984.

7. Margaret A. Hagen, Generative theory: A descriptively adequate perceptual theory of pictorial representation, in M.A. Hagen (Ed.), The Perception of Pictures, Vol. II, New York, Academic Press, 1980.

7. Margaret A. Hagen, Generative theory: A descriptively adequate perceptual theory of pictorial representation, in M.A. Hagen (Ed.), The Perception of Pictures, Vol. II, New York, Academic Press, 1980.

8. Margaret A. Hagen, Varieties of Realism: Geomtries of Representational Art, New York, Cambridge University Press, 1985 (in press).

9. M.H. Hagen, H.B. Elliott and R.K. Jones, A distinctive characteristic of pictorial perception: The Zoom effect, Perception, 1978, 625-633.

10. R.S. Michalski and R.L.Chilausky, Learning by Being Told and Learning from Examples: An Experimental Comparison of the Two Methods of Knowledge Acquisition on the Context of Developing an Expert System for Soybean Disease Diagnosis, International Journal of Policy Analysis and Information Systems, Vol.4, No.2, 1980.

11. Richard S. Michalski, Inductive Learning as Rule-Guided Generalization of Symbolic Descriptions: A Theory and Implementation. Chapter 24 in Automatic Program Construction Techniques, Macmillan Publishing Company, New York, 1984.

Acknowledgement

The authors gratefully acknowledge Robert Garneau's contribution to this research.

Art, Microcomputers and the Mona Lisa

George K. Shortess

Department of Psychology, CU #17
Lehigh University, Bethlehem, PA 18015

Abstract

In this paper I briefly discuss several aspects of art using the descriptive language of cognitive psychology and show how some examples of art by microcomputer can express these characteristics. One of the functions of art in general is to expand the ways in which we perceive the world. Art works using computers have a unique role to play in developing this kind of expansion through alternative experiences, alternatives produced by art objects with a high density of meaning. In contemporary cognitive psychology, these ideas can be considered changes in schemas of art and computers that occur within a structured network of meaning. In addition, the multifunctional nature of computers requires that art generated by computers be time and circumstance dependent.

Introduction

There is a classic demonstration in psychology that involves two light-weight strings dangling from the ceiling of a room. The lengths and spacings are such that while it is possible to hold the end of either string, it is not possible to touch one string while holding onto the end of the other. It is just out of reach. The task of the subject in this demonstration is to tie the ends of the two strings together. The first attempts of most subjects involve simply holding one string and reaching for the other which, of course, does not work. The problem can be solved by recognizing that any of several common objects lying about the room (ball point pen, coffee cup, etc. which are not ordinarily used as weights) can be tied to the end of one string creating a swinging pendulum. While holding the other string, the subject can grab the swinging pendulum, pull the strings together and tie them. The object used as a weight can then be removed. In order to solve this problem, however, the subject must change the usual set of assumptions about the functions of pens and coffee cups and recognize that they can be used as weights on a string. In somewhat the same way, art works should function to provide us with new modes of perceiving, new ways of experiencing the world. While all art need not do this, I would argue that most art, in fact, does this to some degree.

The idea that art can change the viewers' assumptions about the world by uncovering true reality has, of course, been an avowed goal of most new art movements of the past century. For example, Piet Mondrian, the Dutch painter who was a pioneer in abstract art in the first half of this century, said, "Because it is free of all utilitarian limitations, plastic art must move not only parallel with human progress, but must advance ahead of it. It is the task of art to express a clear vision of reality[1] ."

Herbert Read[2] , among others, points to Cezanne's work as pivotal to the fundamental direction of much modern art. Cezanne was consciously concerned with generating a new art that was objective and was to provide for the senses a new structured order that differed from the prevailing norms. This same idea of changing the way reality is represented, and by doing so changing the viewer's experience, has been at least implicit in major stylistic revolutions from the Classical period of Greek art through the Renaissance to the modern era.

But it would seem that it is not enough simply to be new and different. There should be substance, multilayered meaning, or what Nelson Goodman refers to as a symptom of art, multiple and complex references[3] . Many successful works seem to have levels of meaning that interrelate and allow the person viewing the work to see various relationships emerge as he or she experiences it at different times. While this is obviously not a definition of art, it does point to several aspects of art for which the computer has special significance.

Cognitive Structures

In contemporary cognitive psychology there has been a good deal of research on the nature of memory or information structures; that is, how we remember definitions, people, events, places, relationships and all the other items in our memories. Within an information processing approach the question is phrased in terms of structures, often using a computer metaphor. Good introductions to this literature can be found in Donald Norman's book, Learning and Memory[4] or John Anderson's Cognitive Psychology and its Implications[5] .

The typical information processing models suggest that there are semantic networks that

link items together, so that when we think of the Mona Lisa, for example, we would think of it as included under the idea of a painting which in turn is included under the idea of a work of art. We might also think of it as included in a popular song's lyrics, if we are over 35. In this way we link the term Mona Lisa in various ways to larger, more inclusive ideas by various relational terms, and these linkages give different meanings and levels of meaning to the Mona Lisa. This allows for individual differences, since each of us has a different network, yet there is also room for shared meaning by having common elements in the networks of different people. The important aspect of these semantic networks is that they store meaningful relationships rather than any particular statement of that relationship. For example, what seems to be stored in our example is the relationship that the Mona Lisa belongs to the group of objects called paintings, not any particular statement that the Mona Lisa is a painting.

In addition to the meaningful relationships among individual terms as expressed in these propositions, we may also think about groups of propositions which some researchers have labeled schemas. These refer to overall representations of complex ideas, impressions or experiences. The term, work of art, might be thought of as represented by a schema, in which there is a wide variety of relationships among music, painting, dance, sculpture, literature, etc. In addition, there will be various relationships among subclasses of kinds of art works. Within schemas there often seems to be hierarchical organizations such that the general term, works of art, might be at the top with the kinds of art under that and with individual works at the bottom. Further, it seems clear that there are specific cases that epitomize particular concepts. Turning our example around, I can ask you to think of a work of art. You have some idea of what I mean, and if I ask you to provide examples, the Mona Lisa may be one of them. For most of us the Mona Lisa is a prototypical example of an art work. This may account in part for its widespread use as an early computer graphics image.

The extent to which we actually represent natural categories of objects like works of art by single or multiple prototypes is not yet clear from the research. Some investigators[6] would argue that the term prototype should be used only for categories in which a single set of attributes applies, such as color names. There is a prototypical green that corresponds to particular wavelengths of light and other greens relate to this prototype along the dimension of wavelength composition. For a concept like works of art, there is no simple, identifiable dimension underlying the concept to which all cases relate. In fact, we may have a number of typical cases of art works which are not easily related to each other.

However, when I ask you if the computer graphics image of the Mona Lisa is a work of art, and your schema "works of art" includes only objects created before 1940 that are in art museums, you will say that it is not. If your schema of art objects is more inclusive, you may say that this Mona Lisa image is a work of art. Furthermore, the particular object may link together several schemas and provide various semantic links, which become the basis for multiple layers of meaning. The role of mental images, as opposed to words or semantic structures, in all of this is unclear, but there is some evidence that images may be involved in the process of generating these schemas and networks so that we might talk about image networks as well as semantic networks.

Our question then is about how the computer as a tool, a medium, or an art object fits into the existing schemas of art and how computers may change our art related schemas. In one sense, there are two general schemas involved, one for art and one for the computer. For some individuals the schema of art is represented by art objects that are the result of human intuition and emotional processes, far from the schema of a computer, in which the examples are cold machines operating in logical and calculating ways. From this point of view the idea of art as a result of computer operations is a complete contradiction. A number of years ago I gave a questionnaire to a group of undergraduates at Lehigh University, asking them if they felt it was possible to make art with a computer. Twenty-five percent felt that it was not possible. I suspect that this percentage may be lower than that of the general population. The task for those of us who believe that computers can be involved in the creation of art objects is to change the schemas by enlarging the range of cases that art related schemas cover.

Computers in the Visual Arts

Within the possibilities of art using computers there are two general levels of computer involvement, as a means to produce art and as an art object in itself. The first, as means, allows artists to develop ideas that might be expressed in more traditional ways, but with the computer, the artist can be more flexible and more efficient. All of the computer paint systems and graphics workstations provide obvious examples of the ways in which graphics artists can use the computer as an aid to design and as a means for creating visual art. At this level, the artistic output is usually expressed as a print, a photograph, a video or a film. The second level, of computer as art object, is reached when the output is the video monitor operating in real time with the computer itself. In this way the computer is part of the object and the boundary between the two levels, of means and art object, begins to blur.

But what are the characteristics of these artistic productions in whatever form? Multiple

images, rotations and distortions are relatively easy to accomplish and can provide some stunning and exciting compositions. In addition, the "natural" ability of computer systems to produce sharp contours tends to produce hard edged images with a mechanical look. Furthermore, there seems to be a strong desire within the computer graphics field to produce realistic images, and "better" seems often to be equated with the ability to produce more realistic images by including shadows, color gradients, reflections, etc. that mimic the physical properties of the objects portrayed. By default the computer artist too often seems to adopt the aesthetics of photorealism or a kind of super realism. For some this is typical art by computer, often reinforced by television commercials and science fiction sequences.

These more "natural" directions in computer aided visual arts do not have to be there, of course, since computers are flexible enough to be used in other ways. To use the computer for soft edged graphics is, in a sense, to work against many existing system properties. Too often, however, artistic creation using computers is simply the result of "Well let's see what the system will do." And then when the system produces a result there is a kind of "Wow-isn't-that-neat!" response. In this way the computer generated image may be new and different but it lacks the depth of meaning and multiple relationships of the more successful work. On the other hand, there are examples of graphic work in which the computer medium is an appropriate vehicle for the ideas of the artist. By understanding the medium well enough to know what it can do and by having ideas that properly mesh with it, the computer artist can produce innovative work which has a depth equal to work in any other medium. There have been a number of presentations at previous Symposiums where this fit has been accomplished in exceptionally successful ways.

Also using the graphics capabilities as an aid to sculptural design as Frank Smullin[7] was doing, clearly provides the sculptor with a useful tool to accomplish the goals of the work. In my own interactive sculpture[8], I have also tried to use microcomputers to develop artistic ideas about neural function and perception. Again, it is using the properties of the systems to express the ideas. Because microcomputers are built into my sculptures, computers become parts of the works themselves. This use moves into the second category of computer aided art, where the computers are the art works. In a similar way performance pieces using computers in real time provide examples of the computer as the art work[9].

Computers and Artists' Books

Another area of computer aided art is in book publishing. With digital typesetting and computer aided composing, the publishing industry has been at the forefront in the use of computers in commercial applications.

However, within the publishing field there has been a small but highly significant development over the past fifteen to twenty years, in the publication of artists' books. While the current notion of artists' books dates back at least to the various art movements of the 19th Century, the recent development seems to have gotten its start in the social and political atmosphere of the sixties. It has also been fueled by the availability of relatively inexpensive means of reproduction from Xeroxing to the offset press. A number of independent artists' presses have been established, such as the Visual Studies Workshop in Rochester, N.Y. and the Coach House Press in Toronto, Canada. In addition, a distribution network has been established which includes, among others, Printed Matter in N.Y.C. and the Washington Project for the Arts bookstore in D.C. Good reviews of the growth of artists' books have appeared in recent issues of Afterimage published by the Visual Studies Workshop. As an indication of this growth, Printed Matter's 1976 catalogue had 450 books while the current catalogue has over 3000 entries.

But what is it that these presses publish and these stores distribute? The definition of an artist's book, as distinct from other books, is by definition vague. It probably means at least a book created by an artist, often a visual artist, explicitly as an artistic statement that is not so much about art as it is art. When Matthew Hogan, the archivist of the artists' book collection at Franklin Furance, was asked for a definition he responded by saying "Sometimes I don't know. Often an object is considered a book if it is bound and made up of sequential elements"[10].

To put this into the descriptive language of schema theory, we all have some idea or mental structure associated with the schema of a book. The characteristics that we might use generally include a hard or soft cover and sequentially numbered pages of text and/or pictures. Beyond that we might begin to disagree about what a book is; that is, what our schema for a book is. How does a book differ from a magazine or a pamphlet or a newspaper? Are these Proceedings a book? Within this context an artist's book can be seen as a way of enlarging the meaning of the schema associated with the term book, although most book artists would not state their objectives in this way.

Artists' books are often more concerned with the printed page as an object in itself rather than the meanings of the descriptive prose of the text as in a regular novel, biography or other book. In fact, a number of artists' books have no prose but are a series of pictures. For example Sol LeWitt's book, "FIVE CUBES ON TWENTY-FIVE SQUARES," is a series of photographs of five cubes on twenty-five squares in various ordered configurations. In a way this concern

for the book as art object is parallel to the development of modern painting in which the painted, often shaped, canvas is an object in itself rather than, or in addition to, a surface depicting a still life or a portrait.

In other cases of artists' books the text format is chosen to provide visual images that amplify the textual meaning. In some instances the books have been associated with social and political causes, attempting to restructure society, as well as movements devoted to art for its own sake. But in most cases of successful books there are multiple meanings provided by the text or pictures interacting with the book format to provide new linkages of meaning with our information network structures.

While many of the books that have been produced in quantity have used offset presses, a number of other methods of reproduction have been used. Some of these alternatives have included various individualized processes that have produced one-of-a-kind books.

An intermediate approach between the offset press and the individualized book is to use the microcomputer as a bookmaking machine. In some ways it borrows from both the offset process, which has become more and more computerized, and the individualized hand process. It provides a very flexible yet individualized approach to bookmaking. As an example, I have recently begun to develop a small book using a Zenith 151 microcomputer equipped with the Wordstar word processor. As I mentioned earlier I am trying to express in my work ideas about nervous systems, perception, and information structures. This led to the network metaphor and the idea of looking at the world through the grid-like structure of the information network of the nervous system, helped along by some inspiration from Mondrian. This network can become interactive, as I have done with my sculptural pieces. Applying this approach to the idea of a book suggested that the text about the nervous system and perception might be arranged in a way to create a network structure. This is relatively easy to do as I have shown in the page from the book which appears at the end of the article. This represents a very limited first approach in using a microcomputer as a book producing device: networks producing networks. Other extensions, of course, could include direct graphics displays and programs that generate text in random sequences or other formats that are not limited by the usual word processor programs that produce conventional text.

One aspect of this application is that it raises some interesting questions, one of which concerns the physical nature of the artist's book. Since the book is self-consciously an art work, the representation of it within the electronics of the machine could be considered the art work. Further, the machine itself might be thought of as the cover and binding of this electronic or computer book. Certainly the sequential nature of the book is preserved to a certain extent and page turning is simply a matter of key pressing. This is a somewhat different concept of the book than our schema usually produces. If we accept this version of the book, then we have enlarged the schema and changed the ideas to be generated by the term "book" or at least by the term "artist's book."

Concluding Comment

I would make one more point about the use of schemas in trying to understand art generated by computers. This is an idea developed by Nelson Goodman[3] . He suggests that since an object can have different functions at different times, it is necessary to define the time and circumstances under which we considered an object a work of art. The question is not, "What is art?" but rather "When is art?" This requires that our schemas change and shift to include, or not include, certain kinds of items and that the items, therefore, take on certain meanings at certain times, like the pen or coffee cup in the two string puzzle. The same computer can be used to solve engineering problems or to create art, depending on the context; this means that only some things that I create on the machine are successful works of art.

How do we know when the result is art or engineering? I'm not sure I know. But it seems to me that developing new perspectives on our experience is an essential part of much of what we mean by art and that computer related art has a unique role to play in helping to understand this aspect of the art experience by challenging and expanding our art related schemas and semantic structures. Other aspects of art are necessarily involved as well, but then we are getting into material for another time.

References

[1] G. Kepes, The Visual Arts Today. Middletown, CT: Wesleyan University Press, 1960, pp. 91.

[2] H. Read, A Concise History of Modern Painting. New York: F.A. Praeger, 1959.

[3] N. Goodman, Ways of Worldmaking. Indianapolis, IN: Hacket Publishing Co., 1978.

[4] D. Norman, Learning and Memory. San Francisco, CA: W.H. Freman, 1982.

[5] J.R. Anderson, Cognitive Psychlogy and Its Implications. San Francisco, CA: W.H. Freeman, 1985.

[6] W.A. Wickelgren, Cognitive Psychology. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1979.

[7] F.M. Smullin, "Pipedreams, a complete CAD CAM system for tubular sculptures,"

Proceedings of the Third Symposium on Small Computers in the Arts, Philadelphia, PA, pp. 112-127, 1983.

[8] G.K. Shortess, "Neural art: Works based on concepts of the nervous system," Leonardo, 16, pp. 306-309, 1983.

[9] M. Krueger, Artificial Reality. Reading, MA: Addison-Wesley, 1983.

[10] A. Edgar, "A conversation with Franklin Furnace," Afterimage, Summer, pp. 28-30, 1985.

Page sample with network structure.

I look at a painting and what do I see? I see an object, a sensation or a patte rn. Billowing resp onses of nerve cells give me information about what I see. But, no, I am the billowing responding nerve cells - nerv e cells tell other nerve cells. And

experience is the activity o f these ce lls communicating among themselves. But they are us, we are the m. And here I am, a mess of quasi-organized nerv e cells, w ith knowing as electrical nerve cell activi ty.I look at a painting and what do I see? I see an object a

sensation or a pattern. Bill owing responses of nerve cells give me information about wh at I see. But, no, I am the billowing responding nerve c ells - nerve ce lls tell other nerve cells. And experience is the activity of these cells communicating among themselv es. But they ar e us, we are them. And here I am, a mess of qua si-organized ne rve cells, with knowing as electrical nerve cell activity. I look at a paint ing and what do I see? I see an object, a s ensation or a

pattern. Billow ing responses of nerve cells give me informat ion about what I see. But, no, I am the billowing responding nerve cells : n erve cells tell other nerve cells. And exper ience is the ac tivity of these cells communicating among th emselves. But t hey are us, we are them. And here I am, a mess

of quasi-organized nerve cel ls, with knowin g as electrical nerve cell activity.I look a t a painting an d what do I see? I see an ob ject, a sensation or a pattern. B illowing responses of nerve cells give me infor mation about wh at I see. But, no, I am the billowing respondin g nerve cells - nerve cells tell

other nerve cells. And experienc e is the activity of these cel ls communicating among themsel ves. But they are us, we are t hem. And here I am, a mess of quasi-organized nerve cells, w ith knowing as electrical ne rve cell activity.I look at a painting and what do I see? I s ee an object, a sensation or a pattern. Billowing responses of nerve cells give me informatio n about what I see. But, no, I am the billowing responding nerve cells - nerve cells t ell other nerve cells. And exp erience is the activity of t

hese cells co mmunicating among the mselves. But they are us, we are them. And here I am, a mess of quasi-organized nerve cells, with k nowing as electrical nerve cell activity. I look at a painting and what do I see? I see an object, a sensation or a pattern. Billowing

responses of nerve cells give me inf ormation about what I see. But, no, I am the billowing respondi ng nerve cells - nerve cells tell other nerve cells. And ex perience is the activity of these cells communicating among them selves. But they are us, we are them. And here I am, a mess of q uasi-organized nerve cells, with knowing as electrical nerve cel l activity.

IMAGE ENHANCEMENT TECHNIQUES
for
MICROCOMPUTER ART

Charles Glassmire

University of Pittsburgh
Computing and Information Systems

ABSTRACT

Classical mathematical methods for digital image enhancement can be used by the artist, designer and teacher of computer art to enhance digital images. Examples of these techniques are illustrated. After video digitizing an image with a microcomputer, memory arrays of pixel brightness and color indices may be accessed. These arrays may then be manipulated to produce interesting graphic changes. Pixel-point mapping functions can modify image contrast, produce negative images, and allow other creative manipulations. Spatial filtering techniques are available.

1. Introduction

1.1 Monochrome Image Display

A black and white television image is painted on the TV screen by sweeping a varying voltage electron beam from left to right. Repeating this process paints 512 lines of grey tones (called a "raster") down the screen to form one complete image. The grey-tone video image is converted to digital form by connecting the video camera to a microcomputer through an interface card known as a video digitizer. This card converts the continuously changing analog voltage raster into an array of brightness values (called "pixels" - for "picture elements") which are stored in the computer memory as discrete numbers. Each number represents the brightness of a single pixel making-up the screen image. This brightness array is saved in the display buffer, a special piece of addressable memory, and the array is read out to refresh the screen at least thirty times/second to keep the screen image from fading. Raster scan devices have (at least) one memory address corresponding to each pixel location on the screen and containing a brightness value for that pixel. Careful programming can access the display buffer, and by changing the brightness values stored there, produce creative distortions of the image displayed on the screen.

2. Grey Scale Image Processing

2.1 Pixel Point Processes

Pixel point processes manipulate the brightness space occupied by the array of pixel points in the image. A mapping function is defined such that the brightness value of each pixel is modified in a predetermined way by some input function producing a new output image which is enhanced. The enhanced gray level of each pixel at each (x,y) position on the screen depends upon the input gray level function defined for the image process. A new output image is generated based in some defined way upon the gray levels of the old input image. If the new enhanced image is E(x,y), and the mapping function f, a pixel point operation is expressed mathematically as

$$E(x,y) = f[\, I(x,y) \,] \qquad (1)$$

where I(x,y) represents the brightness array of the input image, and E(x,y) is the brightness array for each point of the enhanced image. The operation is carried out pixel by pixel (N, the index on the brightness array in the display buffer, runs from 0 to its maximum value) and the mapping function f specifies the operation completely.

2.2 Uses of Point Processes

The uses of such techniques can provide a powerful tool the artist and designer may use for the control of the digitized image. Some mapping functions as given below are discussed in depth by Baxes (ref. 2) and should be reviewed for detail. Mapping functions exist which can lighten or darken an image. Additionally, this change in brightness may be selectively applied to small areas of the image whose boundaries are known. Mapping functions can modify image contrast by changing the number of gray tones between the darkest tone and the lightest tone in the image. This may increase the contrast

of a dull lifeless image which was photographed on a dark and overcast day, or may reduce the contrast of an image made when lighting conditions were overly bright, producing dark empty shadows and washed out highlights. Mapping functions are discussed below which can change the image from a positive to a negative without losing the detail in the original and without building the inevitable contrast which is the mark of the 2nd generation film dupe. Such a negative image is referred to as "complemented" and the complement may even be applied to a selected portion of the brightness range.

2.3 Examples of Point Processes

Consider the simplest image point process - the copy function. The copy mapping creates a simple duplicate of the original image which is of no practical use but which illustrates the techniques for mapping. This mapping requirement is that every pixel in the output image have a brightness identical to the corresponding pixel in the input image. We can represent the process graphically (curve a-figure 1) by plotting the input pixel brightness along the horizontal axis and the corresponding output brightness along the vertical axis. The mapping function for a simple copy point process becomes a straight line with slope of one. To find the new enhanced brightness of any pixel, we locate its old (input) brightness on the horizontal scale, and read off the new enhanced brightness opposite the curve intersection. We notice the line with slope one maps every input brightness to the same value of output brightness. In this fashion we can generate the enhanced image E(x,y) at every point on the screen. We see the mapping function for a copy is simply

$$f = 1 \qquad (2)$$

and the pixel point operation for copying an image becomes:

$$E(x,y) = 1 * I(x,y) \qquad (3)$$

The pseudo-code for such a mapping function to be displayed on a screen of 512x512 pixel resolution might look like this:

```
{select a row of pixels}
for Y = 0 to 511 do
     begin
       {process each pixel in the row}
       for X = 0 to 511 do
        begin
          {process the pixel at (X,Y)}
          set the graphics pointer at
               screen pixel (X,Y)
          read the brightness value of
               this pixel from Display
               Buffer
          set input brightness =
               brightness value just read
          enhanced brightness = 1 * input
               brightness
          write the enhanced brightness
               to the Display Buffer
        end
     end
```

Suppose now we wish to lighten an image which is darker than desired. This amounts to adding a fixed brightness to every pixel brightness in the image array. The enhanced mapping function is expressed as:

$$E(x,y) = I(x,y) + b \qquad (4)$$

where b is the additional brightness value added overall. In our example image, the maximum reproducable white is 255. No brighter value is displayable on the screen, and zero brightness is the level of the deepest black displayable. All integer values between these limits are allowed. Only two changes need be made to the pseudo-code above:

```
enhanced brightness = input brightness
     just read + b
If enhanced brightness is greater
     than 255 then set enhanced bright-
     ness = 255
```

The output brightness is calculated by formula (4) and a test is inserted after the calculation to insure the added brightness has not produced a value beyond the upper limit. Notice a black density of b will be added to all areas of the input image which possessed a density of 0:

The mapping is graphed as curve b figure 1. This operation is known as a "slide" because it moves the histogram of pixel population in brightness space toward the high side of the brightness axis without changing the fundamental shape of the pixel distribution. The pixel brightness population should be examined to insure that the majority of pixels do not block-up in the high brightness areas of the image, losing essential detail in these regions. An inverse operation may be used to decrease the overall density of an overexposed image simply by changing the value of b in equation (4) to a negative number.

The contrast of an image may be improved in several ways. The meaning of contrast here will be taken to be the separation between the blackest black and the whitest white in the image, as measured on the brightness histogram axis. If darker pixels are added to the image without losing pixels at the bright end, or if pixels lighter than the whitest

white are added without losing at the darkest end, contrast will be increased. Basically the slope of the mapping function changes to produce this effect. This is most easily done simply by multiplying the input image by a constant greater than one to increase contrast and by a number less than one to decrease contrast:

$$E(x,y) = C * I(x,y) \qquad (5)$$

C must be chosen to avoid losing information when increasing contrast, as the bright pixels can easily be moved above the maximum limit for display brightness, causing loss of highlight detail. It may be necessary to do a slide prior to contrast mapping, so as to move the maximum brightness low enough by properly choosing a negative b in equation (4).

Sometimes it is beneficial to add brightness selectively to the shadow areas of the image. The human eye more easily distinguishes detail in the image shadow areas.While there may be structural information in the shadow brightness array, these details may be below the perceptual limit. A mapping function which adds greater brightness to the shadows can be developed by several different techniques. Basically, the shape of the curve must be decreased below a slope of one:

$$E(x,y) = (1- b/255) * I(x,y) + b \qquad (6)$$

This function maintains the condition that the brightest pixels remain under the 255 limit for any added brightness b, while raising the shadow detail an amount proportional to b.(curve d figure 1).

A negative image is simply one which maps the original highlights into the darkest enhanced areas and vice versa. This function is shown as curve e figure 1. The mapping function is:

$$E(x,y) = 255 - I(x,y) \qquad (7)$$

3. Color Image Display

3.1 Color Pixel Display

While the attributes of the monochrome display were discussed above, the situation is more complex for color raster displays. Black and white screens are coated with monochrome phosphors, but the color screen must contain a mixture of phosphors, each representing a primary color (red, green,and blue). These are often arranged in a "triad" of three phosphor dots, which mix their hues to produce any color (see figure 2). Each triad is perceived by the eye as a single pixel. The apparent color of the pixel is determined by the relative brightness of the three primaries in the triad, and a seperate brightness value is stored for each of the three color guns for each pixel. Thus to produce software modifications to the image color palette, the mixture of primaries must be controlled by software to produce the correct color and intensity. These R G and B color values may even be modified from a high level language to produce changes in the display image, providing the addressing scheme is known. Packages such as Multi-Halo do this automatically and work with BASIC.

The color pixel has several attributes associated with it. The x and y screen coordinates of its location, the red, green and blue brightness all must be accessed. In this case the display buffer usually contains a memory location for each x,y pair, containing the address of the appropriate color for that pixel location with brightness values saved in a "color table" located elsewhere in memory. (The actual value stored is an offset which is added to the starting address of the red, green and blue portions of the color table - see figure 3).

3.2 Color Display-Board Architecture

The architecture of a display controller board is shown in figure 4. The Number Nine Revolution Board will be mentioned as typical. The board comes in several models, but 512 x 512 pixels with 256 simultaneous colors is sufficient for professional graphics work. Smooth shading is possible; one may use anti-aliasing techniques etc. The board will display 256 values of red, green and blue, yielding a full color palette of 16.8 million different hues (256 x 256 x 256). Each of the R G and B intensities is saved in a color table, requiring 8 bits per color (24 bits per pixel). The display buffer stores 512 x 512 locations for the address of the color table containing each particular color for the pixel at each x,y screen position. A graphics controller chip on the board controls the display, making sure that the buffer is scanned out at proper intervals, that color values are updated when instructed by the computer CPU and that certain fundamental graphics operations are performed when called by the controlling program. The NNGB graphics controller may be the NEC 7220 or Intel 82720 chip.

3.3 Creating the Color Palette

Graphics controller software is often supplied with a default color palette. This defines the relationship between a

visual hue on the display screen and an identifying color index which the software associates with that hue. The palette provided with most commercial products is modeled after the RGB system and seems to be dictated by logical rules of bit manipulation which show no insight into the needs of artists and designers. The IBM Personal Computer default color palette zero contains green, red and brown (which actually displays as yellow) and one additional background color. Palette one contains Cyan, Magenta and White. These hue groupings strain the talents of even the best designer. The default palette provided with the Number Nine software has its own eccentricities, and it is clear the artist must mix his own personalized palette prior to doing any creative image enhancement in color.

There are more interesting palette systems than the RGB system, but RGB is the one preferred by most programmers. Foley (ref.4) discusses several others more useable by artists, however commercial micro software usually appears in RGB format.

The following pseudo-code shows an example process and the steps needed to replace the default color palette in the NNGB with a simple palette composed of three overlapping sine waves, one for each of the three primaries (fig. 5). The colors intermix to produce a smoothly varying palette starting at red and moving through yellow to green, cyan and finally blue. It is a palette which groups color indices of similar hues together in sequence and makes for simple selection of continuously changing hues. It is weak in the yellow band and of interest only for illustration of the technique of creating a palette.

The pseudo-code assumes a package of high level subroutines which are linkable to call certain graphics functions such as "return the color of the current pixel". Such packages are commercially available and work with a variety of languages:

```
{create a table of sine values in 128
steps from 0 to 180 degrees}
pi=3.14...
anglestep = pi/128 {radian increment}
angle = 0          {initialize the angle}
{load the sine table}
for i = 1 to 128 do
     begin
          angle = angle + anglestep
          sintable(i) = sin(angle)
     end
{load red palette from index 0 dark red
     to index 64 bright red}
set grnvalue=0 and bluvalue = 0
for index = 0 to 63 do
     begin
          red = 255 * sintabl(index)
          redvalue = integer value of red
          {replace default color with
           new computed color}
          call setcolorpalette(index,
             redvalue,grnvalue,bluvalue)
     end
{load yellow to green palette from
     index 64 to 127}
set bluvalue = 0
for index = 64 to 127 do
     begin
          red = 255 * sintabl(index)
          redvalue = integer value of red
          {green phaselag is 64 in sintabl}
          grn index=index-64
          green=255 * sintabl(grn index)
          grnvalue = integer value of green
          call setcolorpalette(index,
               redvalue,grnvalue,bluvalue)
     end
{load cyan palette values from 128 to 191}
          .
          .
          .
{load blue values from 192 to 255}
          .
          .
          .
{plot color spectrum across the display
 screen using 1 vertical line per color
 index}
set x =0 and y=0
set current color index = 0
set xincrement=1 and yincrement=500
for i=0 to 255 do
     begin
          move cursor to (x,y)
          set current color index=i
          {draw line from last point
           to (x,yincrement) in the
           current color index}
          call line(x,yincrement)
          x = x + xincrement
     end
```

When plotted on the screen of a high bandwidth monitor the spectrum begins on the left with deep red, changes smoothly through bright red to yellow to green, then continuously through cyan into the blues, and finally fades from bright blue to indigo black at the right edge of the screen. The color indices for red advance from 1 (very dark red) to bright red (color 64), increasing yellow-green from 64 to a bright green at 128 to a bright blue at 192 and very dark blue at 254. A more accurate approach would allow for the intensity response of the eye to each color. This response curve is gaussian rather than sine-wave shaped, and the red and green peaks occur closer together, allowing for more intense yellow. However scientific accuracy is often not the artists aim and may prove uninteresting. Other palettes may produce better results.

4.0 Color Image Processing

4.1 A Definition of Color Enhancement

Many of the enhancement processes discussed above are defined only for use on black and white images. They operate on pixel brightness, and the translation of these operations to a pixel which is defined by three brightness levels remains ambiguous. One possible substitution is to consider the color pixel to have a brightness proportional to its color index number. This "pseudo-brightness" may then be modified using methods defined above. The results, of course, will not be related to true brightness enhancement of the image unless the artist arranges the palette so that brightness is proportional to index number. While there is little scientific interest in such an approach, the artist may produce some interesting results.

4.2 Spatial Filtering

Digitized color images contain repetitive information in the form of rapidly varying brightness. These variations are referred to as spatial frequencies, and may occur in either the vertical or horizontal direction (or any arbitrary direction). Closely spaced vertical lines of bright white alternating with vertical lines of deep black produce very high spatial frequencies. A series of edges may produce this effect. Alternating wide bands of low contrast greys produce low spatial frequencies. It is possible to analyze these frequencies and remove or modify them when objectionable.

Up to now pixel point processes have been discussed. These modify the pixel gray scale without accounting for spatial position or the behavior of "neighboring" grey scale pixels. A class of image enhancements called "group processes" allow techniques which examine the neighborhood of a pixel and enhance its brightness by some function influenced by the pixels' spatial surroundings. There are many such filters. We will examine only two, hi-pass and lo-pass, to illustrate the technique.

4.3 The Spatial Convolution Technique

The mathematical method which underlies many spatial filtering techniques is called "spatial convolution" and is characterized by the ability to compensate each pixel brightness for image behavior surrounding the pixel of interest. The operation is performed as before by moving across the image point-by-point from right to left, replacing the pixel brightness at each point with a brightness based upon the calculated surrounding area.

Basically, a weighted average of the eight pixel brightnesses surrounding the pixel of interest is calculated. A simple average would be calculated by adding all nine brightnesses, dividing by nine and replacing the current pixel brightness by the average brightness. Instead,a weighted average is computed, multiplying each brightness by a weighting factor which is chosen to select out the image behavior which is of interest (i.e. pick out high frequency behavior, low frequency etc.). Some filters use more than the surrounding eight pixels but this increases the calculation time considerably. This surround, together with the ninth center pixel is called a "kernal", and comprises a square containing nine pixels. The weighting factors are contained within a "weighting mask" which is a 3 x 3 array of integers which are the weighting coefficients. The mask is placed over each kernal, centered on the pixel of interest. An enhanced brightness from a spatial convolution is calculated by:

$$
\begin{aligned}
E(x,y) = \; & (\text{brightness } 1 * \text{coeff. } 1 + \\
& \text{brightness } 2 * \text{coeff. } 2 + \\
& \text{brightness } 3 * \text{coeff } 3 + \\
& \quad \cdot \\
& \quad \cdot \\
& \quad \cdot \\
& \text{brightness } 9 * \text{coeff. } 9 \;)/9 \qquad (8)
\end{aligned}
$$

This is written more succinctly in mathematical notation as two conditions summarized in equations (9) and (10):

$$E(x,y) = \sum_{n=1}^{9} \text{coeff}_n * \text{Brightness}_n \qquad (9)$$

and,

$$\sum_{n=1}^{9} \text{coeff}_n = 1 \qquad (10)$$

4.4 The Low Pass Filter

A spatial filter removes some information from the image topology according to a defined mapping. Not only is the brightness distribution changed but the spatial appearance of pixels and brightness in the image is also changed. The display screen may be considered simply as a two-dimensional database of information with periodic variations of grey levels. These variations are the frequency components relating to how fast the greys change over a given spatial distance. The Low-Pass filter removes high frequencies and emphasizes the low frequencies present in the filtered image. Before using such a filter, it may be wise to assure that low frequency information is present,or else the filter will have no effect. Fourier analysis will detect the

spatial frequencies present (see ref.) or often visual inspection is sufficient.

A commonly used low-pass convolution mask is given by:

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Notice that the sum of the coefficients add to 1. The characteristic behavior of this mask will be to sample a small portion of the surrounding pixel brightness (specifically 1/9th of each) and average these together. The center pixel then receives this weighted average as its enhanced brightness.

The effect will be to reduce sudden changes in brightness across the screen such as might be produced by a sharp edge, whose characteristic is a rapid change in brightness along a straight line region. Edges will be broadened and the effect to the eye will be a blurring of the sharp details of the image. Pixel points of bright white on a dark field will be changed to broader much darker points with a wider squarish shape in a black field. These high frequency points have been blurred to low frequency areas.

One use of the low-pass filter is the elimination of noise in the image. Noise characteristically appears as pinpoints or "snow" in the positive image and the low-pass filter tends to merge these patterns with background surround.

4.5 The Hi-Pass Filter

One convolution mask for the hi-pass filter is given by:

-1	-1	-1
-1	9	-1
-1	-1	-1

This filter works in somewhat opposite fashion to the low-pass filter discussed above. The tendency is to emphasize rapid frequency changes in the image. The enhanced filtered image appears "snappier" to the eye because edges have become darkened and emphasized with the appearance of more obvious detail in the high frequency regions. It should be mentioned that no detail has been added which was not already present, only the frequency trends already present have been emphasized.

5.0 Example Results

A sample image is presented in figure 6a as an unfiltered reference image, and various filtered examples are shown for comparison, including the low and hi-pass filters. It should be noted that the originals were generated in color and the black and white reproductions are from internegatives which suffer severely in the printing process. In addition, all color visual cues have been eliminated, causing further perceptual difficulty. The images are presented only as an indication of the type of visual effects available to the artist when using mathematical techniques for image enhancement.

6.0 Conclusion

Mathematical techniques for microcomputer art are feasible and practical tools for the artist and graphic designer to call upon. Techniques presented here may be used to generate images with acceptable professional graphics standards and are interfacable to the modern electronic studio. What is most significant about the images and techniques presented is their display resolution of 512 x 512 pixels with 256 simultaneously displayable colors selected from a palette of 16.8 million colors. No filter required longer than 45 minutes to process the screen image and the entire system including microcomputer is available for less than $6500.

Bibliography

(1) Klapholz, Jesse,"Digital Sampling and FFT Analysis of Acoustic Sources:A Microcomputer Implememtation",Proceedings of the 3rd Sumposium on Small Computers in the Arts, IEEE Computer Society Press, Silver Springs MD,1984.

(2) Baxes, Gregory A.,"Digital Image Processing", Prentice Hall Inc.,Englewood Cliffs, NJ, 1984.

(3) Castleman, Kenneth R.,"Digital Image Processing", Prentice Hall Inc.,Englewood Cliffs, NJ, 1979.

(4) Foley, James D. and Van Dam, Andries, "Fundamentals of Interactive Computer Graphics",Addison Wesley, Reading Ma., 1982.

(5) Pratt, W.K., "Digital Image Processing", John Wiley, New York, 1978.

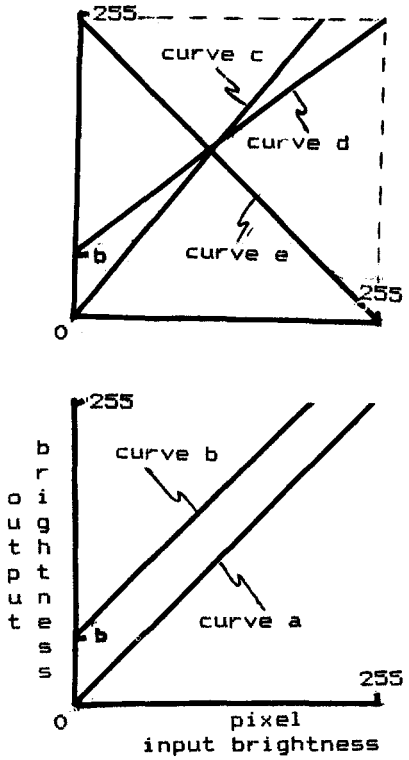


Fig. 1:
Point Mapping Functions

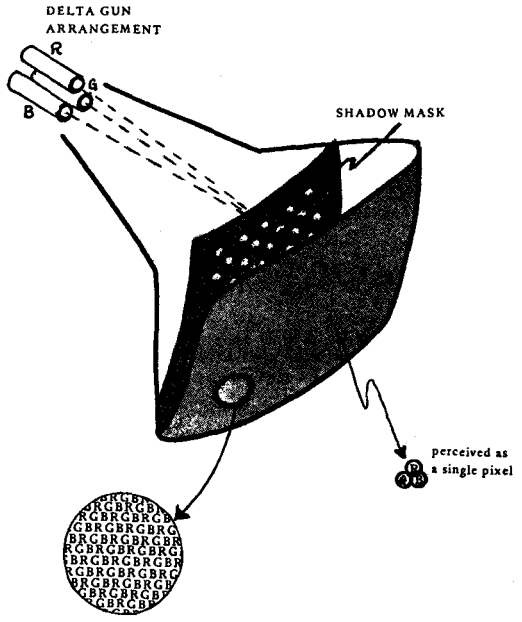


Fig. 2:
Color CRT Display

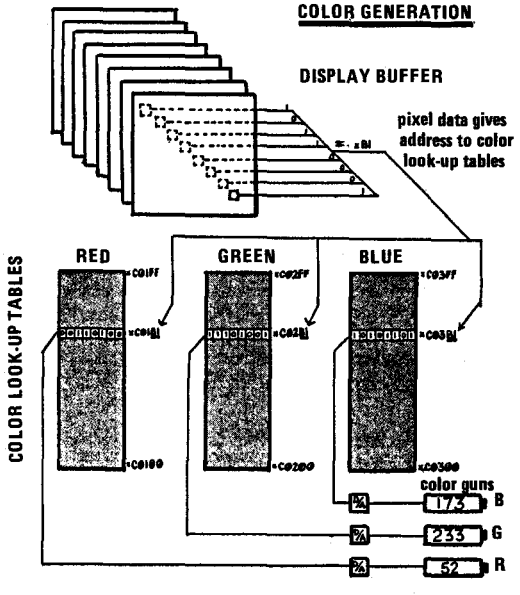


Fig. 3:
Display Buffer

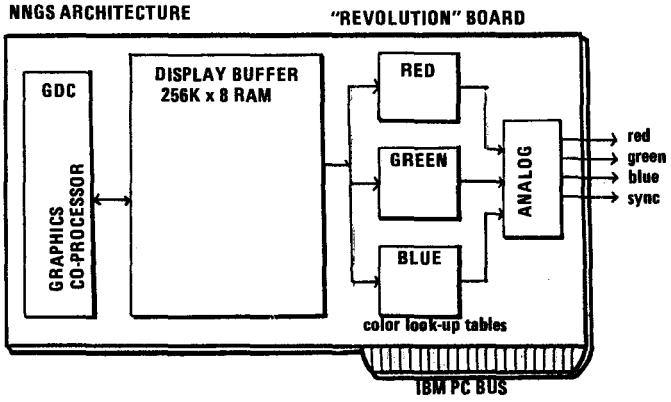


Fig. 4:
Display Card Architecture

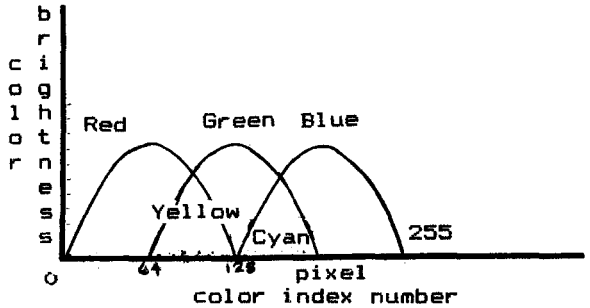


Fig. 5:
Spectrum Composition Functions

Fig. 6a:
Unfiltered Image

Fig. 6b:
Hi-Pass Filtered

Fig. 6c:
Lo-Pass Filtered

Fig. 6d:
Laplacian Filtered

Fig. 6f:
Median Filtered

Fig. 6e:
Orphan Replacement

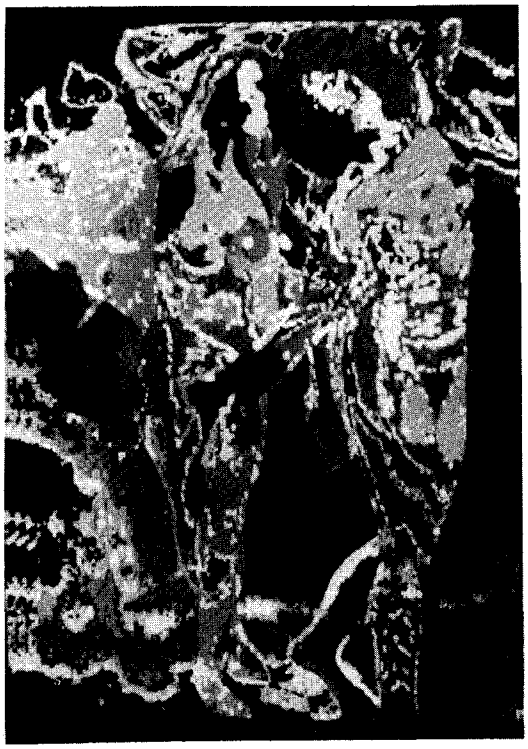


Fig. 6g:
Dilation Filtered

PHILOSOPHY AND CAPABILITY ISSUES IN COMPUTER GRAPHICS SOFTWARE EVALUATION: WHAT DO YOU MEAN I CAN'T ...?

CATHY SPIAGGIA AND CAROLINE BEEBE

Bloomington Academic Computing, Memorial Hall West
Indiana University
Bloomington, Indiana 47405

ABSTRACT

Soon, almost anyone will be able to take advantage of the qualities of a computer which make it a unique graphics tool. This paper addresses some philosophical issues resulting from this technology's impact on the graphics field and outlines some capabilities of graphics packages to consider when evaluating software. The guidelines presented should help users establish criteria for choosing software which best corresponds to their needs.

PHILOSOPHICAL ASPECTS

The human body has often been used as an analogy for the various component parts of the computer. The analogy can be extended to include the relationship between the left and right hemispheres of the brain and the emerging relationship of artists and computer scientists. Historically, humans have described themselves as being either scientific or artistic, or in more recent terms as either left or right brain dominant. Society has tended to emphasize the worth of the scientific, left brain dominant thinkers, and relegated as irrational the imagination and intuition of the right brain dominant thinkers.

As whole brain thinking becomes more popular among educators and psychologists, computer graphics emerges as an experimental arena for the application of whole brain theory to the resolution of the duality between scientist/artist, left/right brain thinking. In its most microcosmic sense, the computer exemplifies duality: on, off. Yet, when the many component parts are considered as a whole, imagination flourishes: artistic irrational expression is defined in scientific, rational terms. Just as with the human brain, the choice is not left or right, but the integration of both for mutual benefit and greater potential.

This integration illustrates a recurrent theme in the development of graphics technology. Artists and scientists are being forced to communicate and work together - to understand each others' conceptual ideas so that the fullest potential of this media can be realized. This symbiosis of artists and scientists is driving this new age with a positive energy. It is creating a need to humanize the technology.

There are ethical as well as technical issues which result from this symbiosis of artists and scientists. One is the availability of powerful artist's tools to unskilled hands and untrained eyes. The computer is conceptually capable of reproducing or creating every configuration of light/color and space - from reproducing great works of art to a thousand variations on a theme. Who will decide which image to use? How will it be decided? As incredible as many graphics software packages appear, knowing how to work through the menus will not in itself develop the user as an artist or supplant the need for a good sense of color, space, and composition. Even for chart and graph packages, the appropriate type must be chosen to convey the specific information to a particular audience. Consider the "chart junk" which results when a software designer makes a decision to translate colors into predetermined black and white patterns: virtual visual chaos.

Another ethical issue is the struggle between "tyranny" and "freedom" for the artist. Computer scientists are designing the artists' tools (hardware and software) not artists, so freedom and creativity must stay within the limitations of the process. It is as if the computer artist must constantly remind himself: "Humans are the ultimate piece of software."

Of course, the "Catch 22" is that in order to design good tools, artists must first work with what already exists and become familiar with the potential of this new technology. Some artists are becoming programmers to deal with the problem. Interestingly enough, a good portion of these artists are finding that programming is just as creative as image making.

Most artists' images, however, are confined by the specific design of another's software. And it seems Marshall McLuhan's concept of the "rear view mirror" generally applies to the way this new media is being used. Educational software developers are designing programs reminiscent of "fast, cheap books" rather than creatively exploring and experimenting with the qualities of the new media. Scientists are designing software tools based on their perceptions of the artistic tools of the past. More realism is one of their recent competitive desires that stems from the scientific/left brain/technical view of art.

CH2218-6/85/0000/0029$01.00 © 1985 IEEE

Scientific designers of graphic tools are emerging from educational backgrounds in mathematical space. They are attempting to make accurate images - forgetting that images are subjective determinations. But, the artist needs access to and understanding of this multidimensional, mathematical space in order to integrate intuitive and imaginative powers of visualization. If artists and visual designers are to begin to take an active role in determining software design, it is important for them to educate themselves to the potential capabilities of image-making with a computer.

Listed below are several inherent qualities of the computer which make it a unique graphics tool. Perhaps understanding these qualities and their implications for the artist would be a good starting place in this re-education process.

Time - There are two aspects of time which computers affect: creation time (making images can take minutes) and presentation time (especially in CAI, designers and artists are able to consider temporal factors when deciding how to bring up an image.

Ease - in transforming color and spatial locations, in making modifications without re-creating the whole image.

Flexibility - to collage images together in several ways (adding, subtracting, logical functions, edge enhancement).

Appropriateness - an excellent tool for conceptual design (e.g. choreography), thinking and editing ideas in rough form - in addition to being an adequate tool for the final product.

GRAPHICS AS VISUAL COMMUNICATION

Microcomputers are increasingly using graphics as the basis for visual communication. Buzzwords such as "icon" and "mice" are now familiar terms to micro users. When considering the capabilities of various graphics packages it may help to first consider the type of visual communication you will be creating. We have categorized four types of visual communication for the micro graphics environment:

Fine Art
CAI (Computer Assisted Instruction)
Charts and Graphs
CAD (Computer Aided Design)

The FINE ART category includes any art or graphic made with the assistance of a computer. This could refer to artists' illustrations and sketches, page layout/publicaton design, and image processing. In this category the artist's primary concern is one of aesthetics even though the ultimate use of the product may fall into the other categories. The CAI category includes any graphic whose purpose is to supplement and enhance instructional material delivered on a computer. The CHARTS AND GRAPHS category, also sometimes referred to as Business Graphics or Presentation Graphics, includes both analysis graphics (extraction of data from a database) and presentaton of information in chart or text form (viewing messages). The CAD category refers to the development of structured images to communicate environmental and structural design.

To help you prioritize the issues of concern in each of these application areas we have developed a table which rates the importance of various issues for each type of application (See Table 1). Interpret the rating scale as follows:

YES - a definite "high priority" issue. Don't consider any package without first considering it's performance in this area.
POSSIBLY - This means "it all depends." This issue may or may not be critical to your particular needs.
NO - It has a very "low priority" compared to the others.

The following list clarifies some of the terms used to describe the issues:

HARDCOPY - Paper, slides or photographic prints as the final product.

SCREEN RESOLUTION - The number of pixels (smallest addressable units) available to you. This is both hardware and software dependent. Even if the hardware capability is there, the software must "know" how to communicate with it. Generally, the larger the number of pixels, the more resolute or sharp the image is.

WYSIWYG - "What you see is what you get" refers to any changes which may occur between what you see on the screen and what you get in hardcopy. Also be aware that some software generates code which may look poor on a low resolution screen but very crisp if sent to a plotter or printer which can interpret the code. Likewise, a full color display on a CRT may give unpredictable results if sent to a black and white printer. Determine if the final product will look like what you see; if not you should check into enhancement capabilities and/or determine if the image is satisfactory for previewing purposes.

SPEED - Refers to how long an image takes to plot on the screen. This is not only an issue for any CRT presentation of the final product, but also affects previewing time during the creation of the image.

FREEHAND DRAWING - The capability of creating an image with non-geometric shapes.

COMBINING TEXT AND GRAPHICS - The ability to combine pictures and text (besides chart labels) on the same screen.

INTEGRATION - The ability to integrate an image file into a program.

TEMPLATES / PICTURE LIBRARY - Basic images or templates available in the software which can be personalized. Some packages also allow the user to create his own templates.

COMBINING PICTURES - The ability to put more than one image file on the screen at the same time - juxtaposed or overlayed.

MODIFYING AN IMAGE FILE - The main question is: Can an image file be modified after it is created? If so, the process should be an efficient one.

EVALUATING GRAPHICS SOFTWARE

It is impossible to talk about software without also considering hardware. Here are some very basic considerations. If you are unfamiliar with the terms and concepts you should find someone who can answer your questions.

One of the most important factors to consider in selecting graphics software is the presentation form you will need. Consider not only your current needs but also anticipate the future. There are three basic presentation forms: paper (printer, plotter, photographic print); slides (for slide projection); CRT (for viewing off the monitor or projection of the monitor image itself). Each software generally specializes in only one or two of these presentation options.

There are many different types of input device options. Consider those that would best suit your needs. Some options include: keyboard, joystick/game paddle, touch screen, light pen, mouse, graphic tablet, digitizing camera.

If you don't own any hardware, you have the luxury of choosing the software that best fits your needs. You can then configure the hardware system that is most appropriate for that software. If you already own some graphics hardware, you will want to limit your evaluation to software that will run on your system. However, be aware that your system may need upgrading (e.g. additional memory or graphics boards). Whether you have hardware or not, compatibility of graphics board, monitor, input-output devices, and software is essential.

Another concern relates to whether your chosen input or output devices plug into serial or parallel ports. Be sure there is a way to plug it all in at the same time. Some devices can only plug into a specific type of port. Ideally your software would give you the option of assigning devices to specific ports. And finally, be sure there are enough slots in your computer for all the specialized boards you may be adding.

As you talk to vendors and view demonstrations, ask how their hardware is specifically configured. Expensive, high resolution monitors and graphics boards can make a dramatic difference in resolution and aesthetics. What you see in a high resolution demonstration is not what you will get if your computer has low resolution hardware.

THE GRAPHICS SOFTWARE EVALUATION FORM

The attached Graphics Software Evaluation Form is an attempt to help graphics users shop for software which best fits their needs. Before using it, though, be aware of one major issue: generally, ease of learning and flexibility are inversely proportional. The easier a package is to learn and use, the less flexibility you will have. Likewise, a package with many options and capabilities may be more difficult to master at first but more flexible in the long run. Your particular situation should determine whether flexibility or ease of use is the more important quality.

In summary, keep in mind that The Graphics Software Evaluation Form is a tool designed to serve a variety of purposes. For example, you could use the form in any one or all of the following capacities:

1. Use it to help make you aware of the range of graphics capabilities. It is better to anticipate what capabilities you might like to be able to have rather than get part way through an image only to exclaim, "What do you mean I can't ....?"

2. It could help you ask the right questions of vendors who often have very slick demonstrations of their software. Vendors will readily tell you what their package CAN do but seldom tell you (unless asked) what it CAN'T do.

3. The form provides a method for scoring each package. The purpose of this scoring system is twofold. First, it will help you determine your needs for specific graphics capabilities. Second, it provides a formula for scoring each capability based on this set of self-determined needs. The final score for the package is the sum of all scores for each capability.

Take care in interpreting these final scores. They are not meant to be the "final word" but rather an aid in comparing several packages. For example, a similar score on two different packages could indicate very different sets of capabilities. If two packages both receive a score of 30, one package may have capabilities which satisfy a number of high priority, "must have" needs, while the other package may receive a similar score by satisfying a larger number of low priority needs.

In another situation, two different packages may have 13 out of 20 capabilities which you "must have." But they could be a very different set of thirteen. In this case you would need to decide which set of capabilities is most crucial to your needs.

The above words of caution are only meant to illustrate that numbers don't tell the whole story. However, the process of arriving at these scores provides you with valuable information which is readily available at all times. Use the form and the information you collect in any way that helps your decision-making process.

Selected Bibliography:

McMillan, Tom. "Special Report: Creating Computer Art." Popular Computing. November, 1984. pp. 73-82.

Palyka, Duane. "Computer/Art--De-Polarization and Unification." Proceedings 4th Symposium on Small Computers in the Arts. Philadelphia, Oct. 1984. pp. 7-16.

Tufte, Edward R. the Visual Display of Quantitative Data. Graphics Press, Cheshire, Conn. 1983.

Wilcox, David L. "The Boom in Business Graphics." PC World, August, 1984. pp. 54-61.

THE GRAPHICS SOFTWARE EVALUATION FORM

Software Name _____ Date_____

Vendor _____

_____

_____

Evaluation Procedure

1. **NEED:** Rate each capability according to your specific needs.

   0 = Not needed
   1 = Nice, but not necessary
   2 = Must have this capability

2. **AVAILABILITY:** Rate whether each capability exists in the software.

   Yes = 1
   No = 0

3. **TOTAL:** Multiply the NEED rating by AVAILABILITY rating for a total score on each capability.

   NEED x AVAILABILITY = TOTAL

4. **ADD** "total" column in each major area and enter the sum in the following table:

Major area	Score
General Considerations	_____
Text	_____
Charts & Graphs	_____
Illustration/Painting	_____
Total Score =	_____

TOTAL	AVAILABLE	NEED		COMMENTS
			GENERAL CONSIDERATIONS	
			Colors available	
			#user-available colors_____	
			Does software include routines for:	
			printer	
			plotter	
			Documentation is clear	
			On-Line Help	
			Communication to mainframe?	
			Transportability (use of images in other programs)	
			Picture Creation capability	
			templates	
			picture library	
			user-defined templates	
			preview on screen	
			Are there Picture Editing capabilities?	
			Is it easy to edit?	
			Is there a low to moderate training time investment?	
			Speed: Is it fast enough?	
			creation time	
			image plotting time on screen	
			multiple copy production	
			TEXT	
			Choice of font types	
			(no. fonts=_____)	
			User-definable fonts	
			Text placement	
			percent of screen	
			cursor/pixel movement	
			justification (left__ right__ center__)	
			Hardware fonts (fast)	
			Software fonts (professional)	
			Hardware and software (option to use either)	
			Combine text and graphics on screen	
			Preview text to check spelling	
			Word processing mode	
			CHARTS & GRAPHS	
			Types Available	
			Bar (stacked, clustered, floating)	
			Line	
			Pie (exploded and unexploded)	
			Text	
			Organization	
			Gantt	
			Scatter	
			Histogram	
			Area	

TOTAL	AVAILABLE	NEED		COMMENTS
			Text on Charts & Graphs: titles, labels, legends	
			Text placement options	
			default placement	
			preset increments	
			user-controlled placement	
			Option to annotate data or add special notes	
			Acceptable character limit for labels, legends	
			Multiple line titles	
			Floating label, legend	
			Axes	
			Change X axis increments	
			Different scales for X & Y axes	
			Multiple Y axes	
			Categorical (named) X axis	
			Logarithmic	
			Flexible tick mark & label format	
			Data Features	
			Is data format appropriate?	
			My needs require a _____ data format	
			(WKS, DIF, DBF, ASCII)	
			Adequate number of data sets	
			My needs require _____ data ranges	
			Adequate number of data points	
			Statistical capabilities	
			Regression	
			Regression with confidence limits	
			Data entry	
			coordinate entry (X,Y)	
			interactive option	
			input file option	
			flexible entry of X data	
			(in non-evenly spaced increments)	
			Data display	
			absolute or additive display option	
			display error bars with data points	
			display data points with line graph	
			Data storage & retrieval	
			database	
			picture only	
			picture & data	
			data transportable to other programs	
			Data editing:	
			move data on graph	
			add/delete/change data without total re-creation	
			Other Features	
			Catalog of available chart templates	
			Combining different chart types on same display	
			Multiple charts per page/screen	
			3D	
			Free-hand drawing	

TOTAL	AVAILABLE	NEED	COMMENTS
			ILLUSTRATION/PAINTING
			Design Issues & Capabilities
			Menu driven
			Command driven
			Freehand capability
			Option to use freehand or geometric shapes
			Create separate objects within an image
			Objects can be manipulated
			____ stored
			____ modified
			____ deleted
			____ reordered
			____ named/identified
			Combine (overlay) different pictures
			Animation capability
			Optional grid aid
			user definable
			grid "snap" capability
			Zoom
			Pan
			Rubberbanding (stretching a temporary outline)
			Object Primitives and Attributes
			Rectangle
			Square
			Circle
			My needs require specifying:
			____ radius & center
			____ diameter
			____ 3 points
			Curve
			Ellipse
			Arc
			My needs require specifying:
			____ 2 points & center
			____ 3 points
			Line - adequate selection of line types
			My needs require:
			____ variable thickness
			____ selection of brush styles
			Color: Is adequate color available?
			(My needs require _____ colors)
			Able to create own colors
			Pallette aprons available
			Able to create own pallette
			Color swapping (on finished image)
			Fill-in capabilities
			My needs require:
			____ fill with color
			____ fill with patterns
			____ fill irregular shapes
			____ user definable fill hatch
			____ air brushing
			Blinking capability
			Curve smoothing
			Fillets (rounding corners)

TOTAL	AVAILABLE	NEED	COMMENTS
			Editing Features
			Easy storage & retrieval
			Ability to modify existing picture file
			Ability to merge different picture files
			Flexible movement
			to any text on graphic object
			move one item
			move group of items
			move by window
			Copying/Duplication
			by single object
			by group of objects
			by areas of screen
			by window
			mirror imaging (symmetrical duplication)
			Deletion
			by object
			by group of objects
			by window
			undo last item drawn
			partial deletion of objects
			Modifying Objects
			scale whole object
			scale by height
			scale by width
			rotation
			X axis
			Y axis
			Z axis (most common)
			change color
			text editing capability
			change text font
			change portion of object

Teaching "C" Programming to Artists

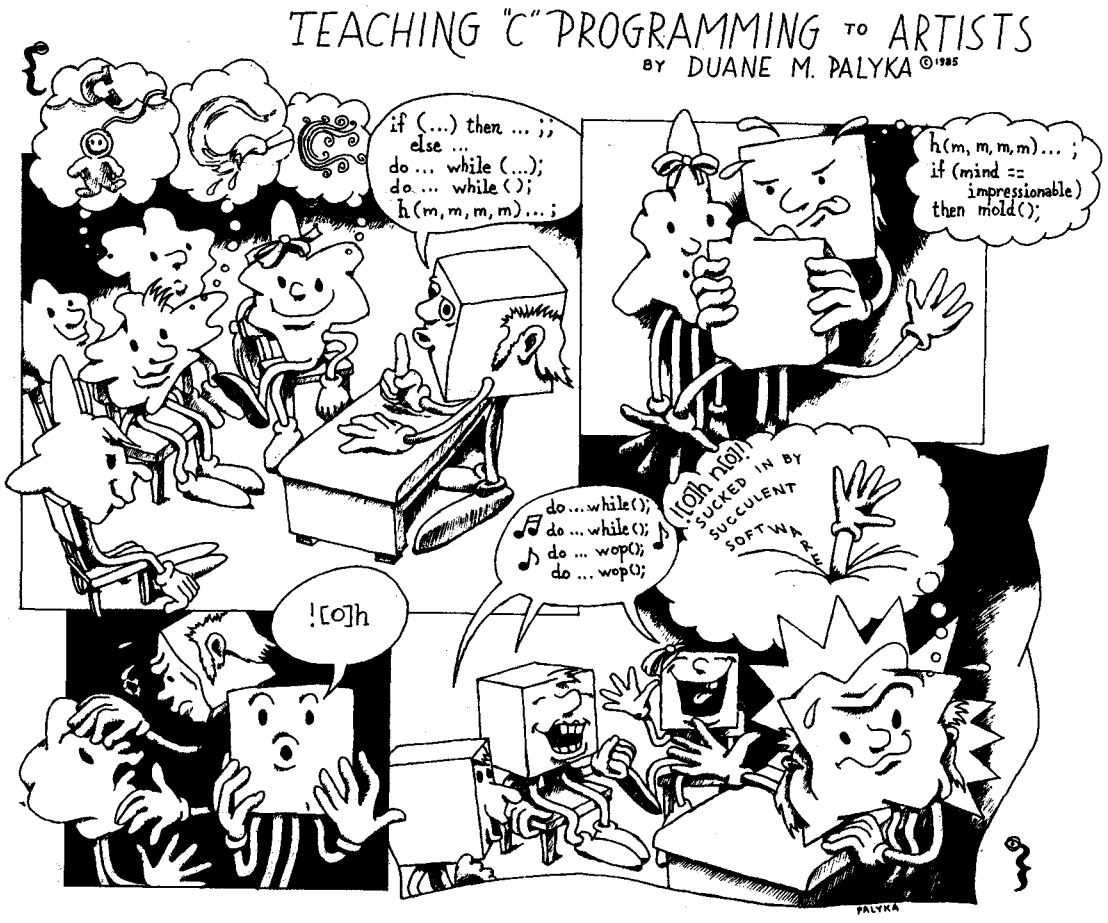
Duane M. Palyka

Computer Graphics Lab
New York Institute of Technology
Old Westbury, New York 11568

June, 1985

ABSTRACT

This paper tells you everything you ever wanted to know (and even didn't want to know) about Duane Palyka's class *Introduction to "C" Programming with Graphics Applications,* taught to artists at New York Institute of Technology. It explores the equipment used, the software both selected and created, the course philosophy, structure, and assignments, and the students' progress and results.



CH2218-6/85/0000/0039$01.00 © 1985 IEEE

Motivation

My responsibilities at New York Institute of Technology include (1) making computer images, (2) programming in the language "C", and (3) teaching one course a semester. Since I have been doing the first activity for about twenty years and the second for about eight years, it seemed that the best way to keep my work load managable was to combine the first two into the third and teach a course on "C" programming to artists. Was I ever wrong! My work load increased dramatically! Since I put so much energy into this new course, I decided to write this paper so others could learn from my experience.

First, I would like to deal with philosophical ideas and rationalizations that convinced me further to teach this course. It always seemed reasonable that artists should not only use systems that others design for them, but should have some experience in designing their own programs as well. I believe that the artistic medium is not limited to user interaction with already-designed packages, but that the aesthetic extends down into the code itself. Ideally, the student should be able to use his newfound skills to create art which other systems (computer or conventional) won't allow them to do. However, even the artist who does not adapt well to writing code, should at least gain an appreciation for what goes into the design of the systems that he now uses.

Why teach "C" in the programming course instead of "Pascal" or "Basic"? Although I currently do most of my programming in "C", I have extensive experience with many different programming languages. My intention was to teach a language that promotes good structured programming habits, which "Basic" does not. Many other "Algol"-like languages could have been used, like "Pascal", but the majority of the graphics community has chosen "C" as their language because it is so flexible and fun to work with and it's the language supported under the UNIX Operating System.

The problems with "Basic" have been discussed in many books, so I'll just briefly state my objections to it: In "Basic" it is difficult to do *modular structured programming* because (1) it allows and promotes extensive use of the "goto" statement, (2) it does not have function calls with symbolic names and arguments ("gosub" is a poor substitute), and (3) it has limited data structuring, looping constructs, and conditionals ("if ... then" but no "else"). Also, implementations of "Basic" interpreters promote writing code which is difficult to read by punishing the user with slower running time for inserting comments and extra spaces into his code. The programmer is thus forced to crowd his code together with minimal comments.

Basically, four factors contributed to the creation of this course: (1) As art students spend more and more time working with paint systems and computer systems in general, many perceive a greater need to learn programming. (2) It is foolish for a person with my background (programming and art) to just teach people how to use paint systems, which was what I was teaching spring semester last year. There are others who can do that just as well. I feel a desire to examine better ways for my unique background and skills to assist students. (3) Since the computer descipline is new to the arts, there is room for creative experimentation in terms of teaching this new area, so the administration graciously allowed me to teach this course on an experimental basis. (4) We already had four Mindset computers and two IBM PC's installed in the art department with very little software running on them. Since we already have several Images Paint Systems, why just bring up more paint systems on these new machines?

Hardware

Tim Carlin, an assistant professor in Communication Arts and Fine Arts, selected the computer configuration according to what would be best suited for our graphics needs; and, indeed, the Mindset Computer is an impressive machine for the price (about $2900). The Mindset Computer is an IBM-PC clone using the Intel 80186 micro-processor, a more powerful upwards-compatible processor than the 8088 that the IBM-PC uses. This means that one should be able to run all IBM-PC software on this machine at a faster rate-- programs that don't require additional hardware, that is. The "Achilles heel" of this fine machine is the inability to add outside hardware to it. New boards coming out which contain high-resolution frame-buffers cannot be put on the Mindset.

However, the real selling point of this machine is it's graphics co-processor which, at lightning fast speeds, does a lot of the normally-slow pixel-by-pixel operations that comprise traditional raster graphics. Using two internal frame-buffers, one blind and one visible, one can very quickly combine an arbitrary window in one buffer with an arbitrarily placed window (of the same shape) in the other buffer. Window combining, or BITBLT'ing as they call it, ranges from simple pixel group replacements to complicated logical operations on pixel groups. With this feature one could do CEL animation in real-time (provided one had software similar to a Mindset implementation of Interactive Picture Systems' *Moviemaker* software). The BITBLT operation is also great for simulating sprites for developing video games.

A common trick with this machine is to render different views of a three-dimensional form in the blind buffer, and BITBLT the 2D windowed images over the same spot in the visible buffer to make the 3D image look like it's changing in real time. If the only requirement is to see cycles of specific sequences of the same object turning, the results are truly impressive. However, equally impressive are other features of the graphics co-processor that allow arbitrary rendering of three-dimensional objects fast enough so that the viewer can maintain image continuity over time. On a simpler level, this amounts to the ability of the co-processor to render a filled polygon dithered in two colors at a lightning-fast speed. I chose to concentrate on this latter feature in my class.

The Software Nightmare

Since we have both IBM-PC's and Mindset computers, and since the Mindset is supposedly an IBM clone, I originally wanted to develop software that would run identically on both. So, I started off thinking that a combination of the MS-DOS Operating System, The Lattice "C" compiler, and the HALO graphics package was the right way to go. This was the software that Tim had purchased for my class. The MS-DOS Operating System is a pleasant combination of the old standard CP/M Operating System and a subset of UNIX, the operating system of choice for mini's. MS-DOS is in such wide use that if the students never learn anything about programming, their knowledge of the MS-DOS Operating System would at least give them some credibility in the marketplace. The Lattice "C" Compiler is a good implementation of the Bell Laboratory standard, and the HALO package seemed powerful enough for most student graphics applications. With all this great hardware and software, it appeared that putting this class together was going to be easy. Well, not so! My problems began when I tried to get HALO to run on the Mindset and couldn't. At that point I started doubting Mindset's contention as a true PC clone.

Since I couldn't get HALO to work on the Mindset, I began rethinking the situation. Perhaps it wasn't so important to have the same software working on the Mindset and IBM. The HALO package didn't make use of the greater color range on the Mindset (sixteen colors versus four colors) nor of the graphics co-processor; so I decided to use the Mindset Macro-assembler and to personally code the interface between "C" and the Mindset co-processer. The task didn't seem that ominous. For each routine, you just had to load a few registers with the right values and call an interrupt; and since each routine was organized so similarly, the macro assembler should have been able to generate routine-dependent code from a few macro templates.

Writing the code was fun, but getting it to run was horrendous! I had never seen a macro assembler with so many bugs in it! After hours of trying to determine which bugs were in my code and which were in the assembler, I finally found the right working subset of their assembler to implement my ideas. I even managed to develop the code so that each function could be expanded from a single line of code, which was the original plan. Since we have four Mindsets and two IBM's, and since all systems run MS-DOS, the IBM's could now be used for editting, compiling, and linking-- everything except running the Mindset-dependent programs. After generating some working "C" code to test my interface, all equipment was in place for the course. Now I just had to design the course.

Course Development

Since I had spent so much time and energy on software development, I had little left for course development. So, in my usual intuitive manner, I decided to design the structure of the course as we went along.

Actually, I had a general game plan, but I didn't know what would work with these students and what would not. So, I chose to play it by ear. My intention was two-fold: First, I wanted the students to learn as much of "C" as possible, using graphic output as seduction into writing code. I figured that what motivated me to learn programming in college should be good enough to motivate them as well. I did poorly in programming courses that I took for the sake of a grade, but once I found that I could make visual images through programming, I eagerly dove into the coding process. And Secondly, I wanted the students to enjoy the class enough to invite me to a *free sushi dinner* on the last day of the semester. Obviously, this was the harder of the two goals.

I purposely called the course "Introduction to "C" with Graphics Applications" so as not to imply that the complete "C" language would be taught. Hopefully, the students would learn as much "C" as was comfortable within the structure of the course, and should have acquired the ability to later extend their "C" skills as necessary. Briefly I considered calling the course "Introduction to "C" with Free Sushi Output", but I figured that it wouldn't look good in the course handbook.

The First Two Assignments

In the span of the fourteen week course, I presented four problems to the class. The first problem was simply the assignment to print something on the terminal. Besides providing an introduction to the functions "main" and "printf" in "C" programming, it required the students to become familiar with the "edlin" editor, the lattice "C" compiler and linker, and the MS-DOS Operating System.

The second problem dealt more with graphics. I gave each of the students a copy of an already running program that drew one rectangle in a preset position on the screen, and challenged them to modify this program to make an aesthetically pleasing composition consisting of three differently colored blocks. Since each block could be dithered with two different colors, the sixteen-color palette of the Mindset now expanded to a 256-color palette, thus enabling them to concentrate on subtle color relationships. I suggested emulating the color sensitivity found in the work of Joseph Albers, Ad Reinhardt, and other minimalist and colorist painters.

After each student got his 3-block program barely working, he was forced to explore how to make sensitive color and positional changes without recompiling and reloading every time he needed a variable change. Naively, I suggested use of command line arguments, but the internal mechanism of parsing those arguments proved too complicated for beginning students. Instead, one of the more experienced students jumped to page 105 of our textbook *(C Progamming Guide,* by Jack Purdum, Que Corporation), and used the "scanf" function for numerical input. "Scanf" for input complements "printf" for output, and worked just fine once we worked our way through the bugs in the Lattice "C" implementation. This was one of the few functions in Lattice "C" with which we had problems.

Incidently, although the Purdum book looked like a good textbook at the time Tim and I were investigating them, it doesn't seem to have enough good examples for the students to follow. Further books that surfaced during the course of the semester were *The "C" Primer,* by Hancock and Krieger (Byte Books, McGraw-Hill, 1982) and *Programming in "C"* by Stephen G. Kochan (Hayden Book Company). I'm going to have to investigate textbooks further before I teach this course again.

Although I wanted the students to work strictly within the scope of this problem, I did accept other "block program" solutions as extra credit as long as the student did not neglect the original problem. Theoretically, in grading, I weighed four factors: (1) creativity in the visual image, (2) creativity in programming, (3) amount of energy expended (including what they did for extra credit), and (4) amount of previous programming experience. In actual practice, I depended heavily upon my accumulated knowledge of what each student did.

The Last Two Assignments

For the next two problems, we exploited the speed of the graphics co-processor by working on time-based projects. In problem three I required the students to write a program which interpolated one polygon into another (with the same number of sides) having at least eight rendered steps between. The student would define the first and last polygon by inputting lists of numbers which formed

the (x,y) points of polygons in the 2D cartesian coordinate system. To assist them, I gave them a two-page verbose prose explanation of the algorithm. Up to this point I had been limiting the students to use of global variables and simple function calls, but now we started exploring the use of conditional testing, nested loops, and arrays.

In the solution to the problem the student had to use at least two loops and a little arithmetic. The outer loop controls which polygon of the sequence is to be drawn, i.e., how much weight this polygon is given relative to the first and last polygon, and the inner loop calculates all of the (x,y) vertex values of this particular polygon based on that weight. Because the program ran so quickly, the student also had to use "wait loops" to kill time between polygon changes so the viewer (who didn't function at microsecond speeds) could see each iteration of the transforming polygon.

Again, we relied upon "scanf's" for input. However, since the input consisted not only of colors and timing information, but also all of the (x,y) coordinate numbers for both the first and last polygon, we started redirecting the scanf's from tty input to file input. Redirection of standard input is a nice feature of the Unix Operating System that was also implemented in MS-DOS.

Originally, I wanted the students to continue to use the two-color-filled polygon tiler in the graphic co-processor, but since the tiler has some restrictions placed upon how the polygons are designed, I allowed them to render their figures as line-drawings if they so chose. The tiler restrictions allow concavities in the horizontal direction but not in the vertical. Some students chose to combine both solid figures and line-drawings in the same work.

During the course of this problem, one of the students brought it to my attention that it would be nicer to write programs that had more sophisticated graphic output. Well, I couldn't help but agree. I had really wanted the class to explore some three-dimensional computer graphics concepts involving development of a 3D database and rotation, scale, and translation of that database. The display should show 2D instances of the transformed database with hidden surfaces removed. So, in the last few weeks of the course I developed the software to do the above and presented it to the class as as module containing two relevant function calls. Calling the first function defined a 3D block figure by inputting a 2D polygon from a file, extruded it a certain "z" value into space (reversing its direction), and calculated all the polygons along the edge separating the two faces. Reversing the back polygon's direction made any "normal vector" calculation show the back and front polygons to face in opposite directions. Calling the second function created a transformed 2D instance of that figure. The first function was called once in the program and the second called each time the student wanted to change the picture of the figure.

The first function simply built a solid points/polygon database from a 2D figure-- a figure which I suggested could be one of the interesting intermediary polygons created from problem three's interpolation. The restrictions to the shape of the polygon were (1) it must not have any vertical concavities as mentioned above, (2) it must be numbered in a clockwise direction, and (3) the first and last three points must be convex. The first restriction again originates from the hardware restrictions of the polygon tiler in the graphics co-processor. The second and third limitations were required for calculation of whether or not the polygon was facing towards the viewer or away from the viewer.

To see if the polygon should be drawn or not, I used the first three points of the transformed polygon to calculate the "z" component of the "normal vector". This is commonly known as the "poor man's hidden surface algorithm" and works very well for a certain set of solidly-defined 3D figures. This method is fast and allows the figures to be rendered close enough to real-time so that the viewer's mind can form a continuity of action. For each 2D instance of the object, I transformed each polygon and calculated its visibility (stopping the transformation short if the partial "normal vector" calculation of the first three points indicated that it was not visible). Then, with each visible polygon stored in a list, I rendered that list as above. In the Mindset configuration, the graphics co-processor would be rendering a polygon while the 80186 would be retrieving the next polygon on the list.

The point of this problem was to provide a fairly sophisticated 3D graphics experience for the student with functions that were very complete in their own right and could be easily incorporated into the student's program. The user's creativity entered by allowing him the freedom to create fairly free-form 3D objects and manipulate those objects in near realtime.

Results

Whenever I consider the art of computer programming, I think of using a set of tools to "paint" a logical output. The working program is somewhat analygous to some sort of kinetic sculpture which paints pictures. I have always reacted strongly against learning things by rote, and frown upon teaching like that. I like to give reasons for why information should be learned. I think of the elements of a programming language as building blocks whose knowledge of which gives me more freedom. Contrary to this philosophy, I found myself feeding the students by rote more than I had wanted. Instead of handing them programming ideas to explore, I had to practically give them answers by coming dangerously close to the solution in each assignment description. The unfamiliarity of the programming experience probably required this approach, for they seemed to relish freedom in the graphics part of the assignments, but this issue is still bothersome to me.

The fact that computer languages require precision in syntacical expression, coupled with the fact that we used a compiler instead of an interpreter, made life difficult for the students. (It's no wonder that an inferior language like BASIC could become so popular!) Constant reeditting and recompiling using floppy-disks ate up large amounts of time. Syntax errors were the killer. There was always a missing semicolon or an ending bracket which didn't match a beginning one. Ninety percent of my time was spent running from student to student trying to help each individually find his or her syntax error. In spite of the feeling of overwhealming popularity having so many students cuing so intensely for my time, I sometimes thought twice about coming to class. Until I caught on, the more aggressive students got their programs working faster than the quieter ones.

"Syn"ically, I thought, if I could tax "syn", I'd be a rich man today. The students taught me that programming could be a "syn"ful and "pun"ishing experience. I wish I could have spent my energy in more constructive ways than continually correcting syntax errors! (or thinking of bad puns!)

Even on the next-to-last day of class I had three separate requests in a row for help concerning missing end brackets. When out of frustration I started spontaneously composing and singing a song about matching squigley brackets, the "squigley-bracket questions" ceised quite suddenly. There's power in bad singing! -- probably more than in bad puns.

Another mistake was the use of MS-DOS's very unforgiving editor combined with the requirement that the students include some of my code in their module (I had difficulty using the "#include" statement in the Lattice "C" compiler). Torturously, a few of students spent honest effort into finding ways to lose at least half of their file during the editting process. Their tears received no sympathy from me! When they retyped the lost part of their code, they always had to retype my previously-working code as well, forcing me to tinker with their copy of my own code to get it to work again. Trying to understand what they were doing, I inevitably failed to make the "correct mistakes". Finally, in an anxious moment near the end of the semester, I managed to lose part of my file in a similar manner. It took a weak moment of emotional cloudiness to wipe out my usual, cautious, step-by-step well-instituted programming habits enough to find the file-distroying error. However, because of my "intensive training in good programming practices", I was able to keep track of the sequence of events that led to the problem, and, at last, make some headway against their "spoiler tactics". Emotional expressionism has no place in the world of computer programming.

Seriously, however, on the positive side there were some students who surprised me with some inventive software design. John Mack went off on his own and generated several interesting programs that, besides being the most advanced programs in the class, also paid noble tribute to "NYIT" and "C". Leslie Nobler wrote her software in mature and creative ways. David Schilling invented a unique "man-into-dog de-

evolution" sequence, followed by a "conformity" program in which human heads transform into geometric forms. Jerry Sarrantonio's program created color blocks so subtle that the viewer had to adjust the brightness knob on the monitor in order to see them. Cynthia Appold's "Buddah" (not to be confused with "Buddha") transformed a meditating figure to and from vertical and horizontal rectangles. Andrew Borg combined all of the problems into one grand piece whose pacing matched the music that only he hears on his tapedeck with headset. And, finally, Lynn Thompson came up with a transformation sequence that was so interesting and bizarre that neither she, nor anyone else, could figure out how it was done. Then, for an encore, she lost the program and its source code.

I encourage the students to take advantage of serendipitous effects whenever they can. The fact that Ms. Thompson got her program to run and produce interesting results in spite of many unknown factors is to her credit. I actually miss the days when I was a naive programmer and had many programming mistakes to take advantage of. Now I must rely upon program complexity to produce serendipitous results.

The last two class problems, being time-based, brought to my attention the lack of experience the student's had in working in the time domain. Most of the timing was rather slow and mechanical. Either they simply did not do enough time-based editting in their other classes, or they were just so overwhealmed with this new programming monster with its syntactical teeth that they didn't have enough time or energy to concentrate on this factor. Actually, it was probably a combination of both. I suppose that one can overlook these factors in first course.

It actually felt like the course was just getting started when it was over. All the students and the instructor concurred that there is a definite need for a followup course. Unanimously, they said that they would take such a course if it were offered. Really, I knew that the course was going well when John Mack suggested that the class take the instructor out to have sushi on the last day. (Aha! The ultimate goal of the course was about to be realized!)

The Flip Side

Before I start patting myself on the back, I should balance the above with a few of students' complaints: (1) Too many advanced tangents were taken during lectures. (With one or two experienced programmers in the class, I sometimes got caught up in answering their more advanced questions while the rest of the class stared glassy-eyed into space.) (2) Neither the book nor the instructor presented enough examples of how different "C" constructs are used. (Alas, again I am reminded to correct my teaching style.) (3) There was too little machine availability and not enough assistance available outside of class time. (Nobody else knew "C" but me. *I* was the one who *really needed the assistance!* Fortunately for me, John Mack, an advanced Basic programmer studying "C", quietly assisted me by helping some of the students– without me even asking. Also, problems were created by a badly-proportioned student-to-computer ratio. Two-to-one would have been better than our three-to-one.) (4) Some of the students felt inadequately prepared for the class. (Closer attention has to be paid to prerequisites.) And, finally, (5) the sushi restaurant that the instructor chose was too expensive. Because of limited student budgets, he was actually forced to pay his own way. (The ultimate objective of the course was wrought with partial failure!)

ABOUT THE AUTHOR

Duane M. Palyka is a senior research scientist at the New York Institute of Technology Computer Graphics Laboratory. He received both a BS in mathematics and a BFA in painting from Carnegie-Mellon University. During his employment as a research associate and systems programmer in computer graphics at the University of Utah, he received an MFA in painting from their art department. His computer art has been exhibited internationally since 1968. For the past two years, Mr. Palyka has taught the following courses in NYIT's Art Department: *Media and Art Applications of Computer Graphics,* and *Introduction to "C" Programming with Graphics Applications,* and has assisted Peter Voci and Tim Carlin with: *Microcomputer Graphics,* and *Images I and Images II* (using NYIT's paint system).

DIGITAL TEXTILE CONSTRUCTION: USING LOGO

Marie Morello Ozmon
P.O. BOX 153 Hopewell, New Jersey 08525

ABSTRACT

The LOGO computer language was used to simulate textile constructions. Textile constructions were defined as textile images that could be translated through weave drafts into plain, satin, or twill weaves or weave combinations and hence, reproduced in cloth. Thirteen procedures were manipulated through changes in variables, cursor placement and heading, and pen and background colors to simulate the appearance of woven cloth.*

Textile construction begins with the making of a weave draft, a schematic diagram illustrating a weave plan, which enables a weaver to determine how a specific textile image will be translated into woven form. It illustrates a variation of any one of the three basic weaves, plain, satin, or twill or combinations of the same. In the system used by the Philadelphia College of Art, the weave draft is developed by filling in blocks in a specific tiling sequence on graph paper. Squares which are filled in on the grid, "raisers," are warp threads which are raised above the surface of the thread plane. The white squares represent "sinkers," threads which either remain on the thread plane or are lowered below the thread plane in conformity with the type of loom employed for weaving.

Filling in squares on grid paper in a specific sequence to indicate the position of warp threads is a repetitious and tedious process often requiring several hours for the completion of one draft. The advent of the digital computer made it possible to reduce this time to a few minutes.

*This paper is adapted from a M.A. thesis entitled: THE DIGITAL COMPUTER AS A BASIS FOR TEXTILE CONSTRUCTION DESIGN: AN INVESTIGATION OF THE EFFECTS OF LOGO. PHILADELPHIA COLLEGE OF ART, MAY 1985.

Janice R. Lourie, a researcher for IBM, and W.G. Wolfgang, of the Philadelphia College of Textiles, were among the first to apply the computer to textile design and manufacturing. With the advent of microcomputers, access to programs for non-industrial drafting applications has been rapidly expanding.**

Currently available microcomputer drafting programs simulate a variation of manual drafting. However, they fail to use computer systems for providing a digital environment in which the textile artist can explore alternative means of composing digital textile constructions.

Previous LOGO Applications

During the past four years LOGO, a language developed by Seymour Papert and his associates at MIT and derived from LISP, a language of artificial intelligence, has become available to microcomputer users. In addition to its widespread use as a language for children, and as an exploratory environment for mathetic experience, LOGO permits textile artists to engage in the writing and testing of recursive procedures. Recursive procedures may be written with a variable input, :LEVEL, which determines the number of procedure repetitions. In addition to supporting recursive programming, LOGO permits "syntonic" or bodily-associative learning.

** Janice R. Lourie, Textile Graphics/Computer Aided (New York: Fairchild, 1973), pp. 104 and 243, and pp.177-188.

W.G. Wolfgang, "A Computer Program to Generate Weave Structures," Computer Graphics and Art, November 1976, pp. 10-17.

For examples of microcomputer drafting systems see:

Weave Master: Operator's Guide Reference Manual (York, Maine: Macomber Looms,1982).

Generation II (Chico, California: AVL Looms).

In the LOGO microworld, the textile artist may describe through the cursor or turtle the movement of threads in terms of self-movement: forward, back, left and right. Likewise, the artist may choose to use the fabric concepts of wrap, fold, stretch, and shrink to describe thread movement.

Complex Curves and Networks

During Summer 1983 the writer discovered that complex curves could be manipulated via size changes, rotations, manipulation of the raster image, and extensions of levels of complexity to simulate textile imagery. These curves are discussed by Abelson and DiSessa, Winston and Horn, Thornberg, and Mandelbrot in his definitive text.*** A research study of the use of LOGO for textile construction was begun. The study investigated whether LOGO could be used to make textile constructions. Here textile constructions means textile images which can be interpreted via weave drafts in plain, satin, or twill weaves or combinations of the same and hence, reproduced in cloth.

Thirteen procedures were either adapted or developed which produce complex curves, irregular and/or fragmented at all scales (See Appendix). Each procedure can replicate itself as many times as specified by the input variable :LEVEL. A :LEVEL of 1 produces a "generator" or basic shape as shown in Plate 1. The "generator" or subsequent scaling copies can be enlarged, wrapped around the screen, and rotated by changing the cursor heading and/or position prior to or during the execution of a procedure. The choice of a specific numeric input for :LEVEL is not only determined by the artist using the fabric concepts of wrap, fold, stretch, and shrink, but also by the limitations of system memory and resolution capacity.

All of the thesis procedures when preceded by a RT 33 or RT 45 LOGO command and :Size inputs of more than 500 produce wrapped and stretched generators which visually simulate network constructions. Networks are fabrics which are composed of looped threads. Hair and fish nets are examples of networks.

This propensity for LOGO procedures to simulate network constructions is directly related to the nature of the LOGO graphics and the wraparound characteristics of the raster screen.

*** Abelson, Harold and diSessa, Andrea. Turtle Geometry: The Computer as a Medium for Exploring Mathematics. Cambridge, Mass.: MIT Press, 1980.

Plain and Twill Constructions

Procedures which produce networks can be manipulated through variable input manipulation, cursor rotation, and background color changes, to produce textile simulations of twill fabrics. Translation of these simulations into a twill fabric would require a Jacquard loom to accurately reproduce the complex sequence changes which occur in the diagonal twill line intersections.

Not all variable inputs will produce satisfactory images. The artist must explore the tolerances for each procedure. Otherwise, it is possible to produce a maze of scrambled lines. Suggested starting points for variable inputs are encoded in the preliminary notes for each procedure. Investigation of the procedures used in the thesis indicates that the largest category of textile simulations, based on complex curves, is that which can be interpreted in warp- or weft- faced plain weave. Within the context of this study the "generators" are shrunk or folded from their positions in the network and twill constructions to form weaves in which the warp or weft threads form the visible cloth. The second set of threads remain hidden within the cloth construction. The background rep cloth for each of these constructions results from either manipulating background color or overlaying the shrunk or folded "generators" over a rep cloth simulation.

One of the LOGO procedures developed during the thesis investigation, GRID2 can be manipulated, with specific variables to produce a simulation of a plain weave gauze construction. In this weave the warp and weft threads are evenly balanced and form an open gauze-like textile. The digital image of GRID2 may be stored to disk, recalled as desired, and drawn over with other images to produce simulations of embroidered plain weave.

The following plates illustrate the results of manipulating three different procedures. For each procedure the "generator," and resulting curve are shown, as well as an example of a related network, twill, and plain weave construction.

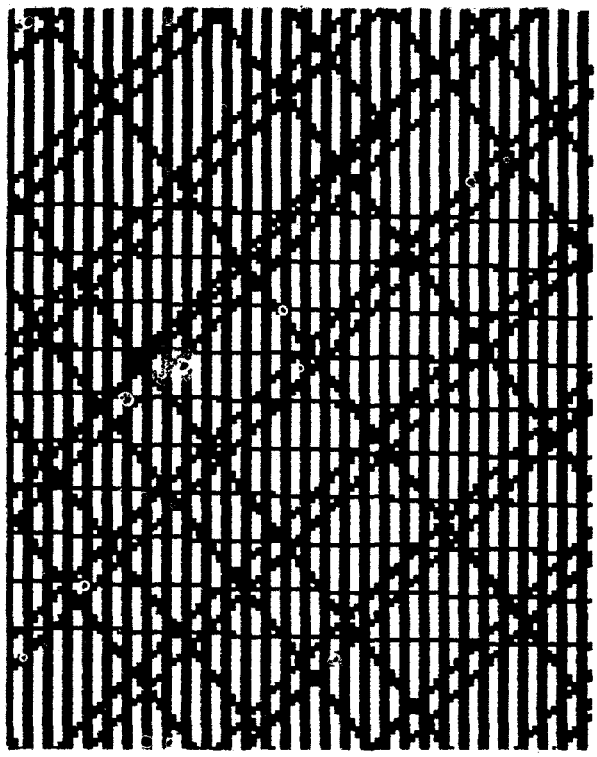
Mandelbrot, Benoit B. The Fractal Geometry of Nature. San Francisco: W.H. Freeman, 1982.

Thornberg, David D. Discovering Apple Logo. Reading, Mass.: Addison-Wesley, 1983.

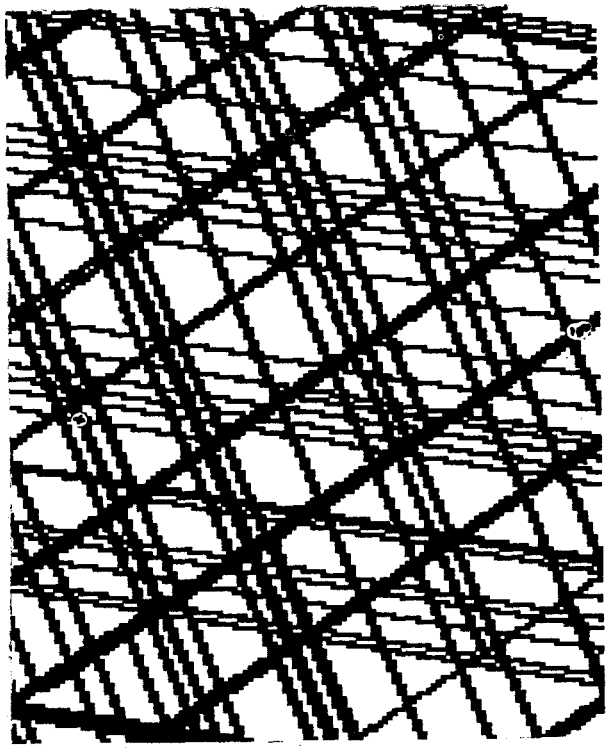
Winston, Patrick Henry and Horn, Berthold Klaus Paul. LISP. Reading, Mass.:Addison-Wesley, 1981.

Snow Generator

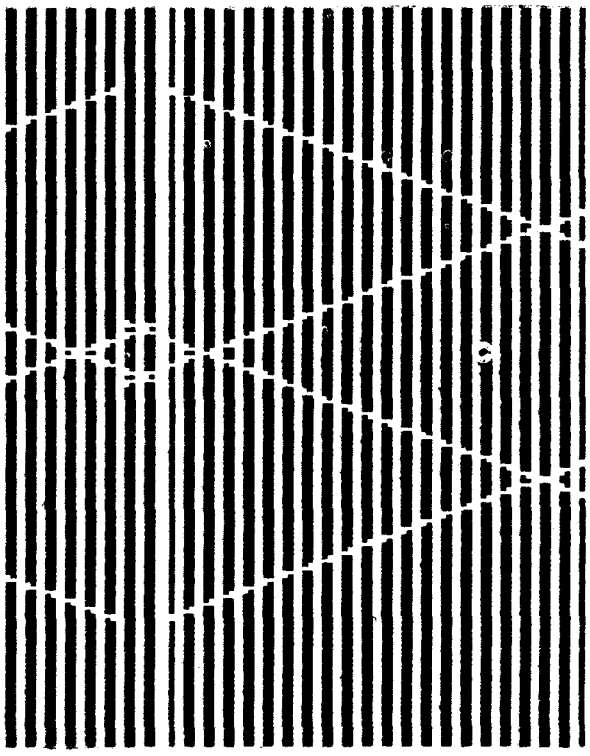
Snow Curve



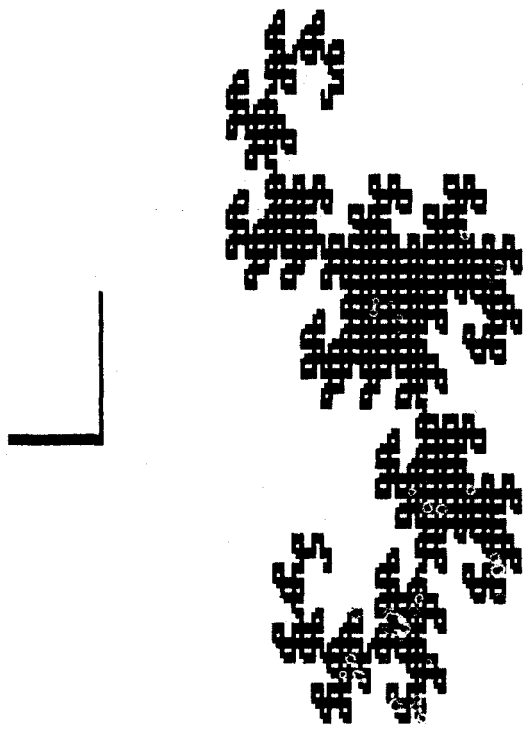
Snow Twill



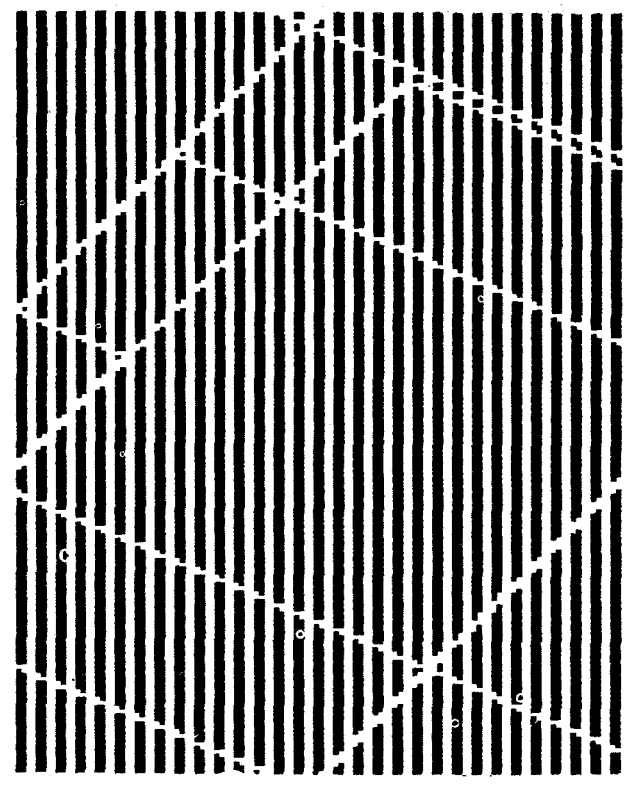
Snow Network



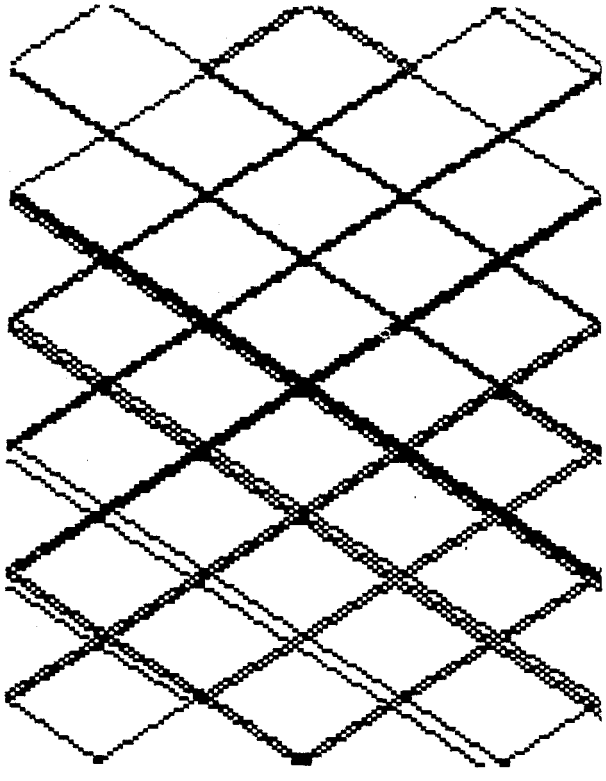
Snow Plain



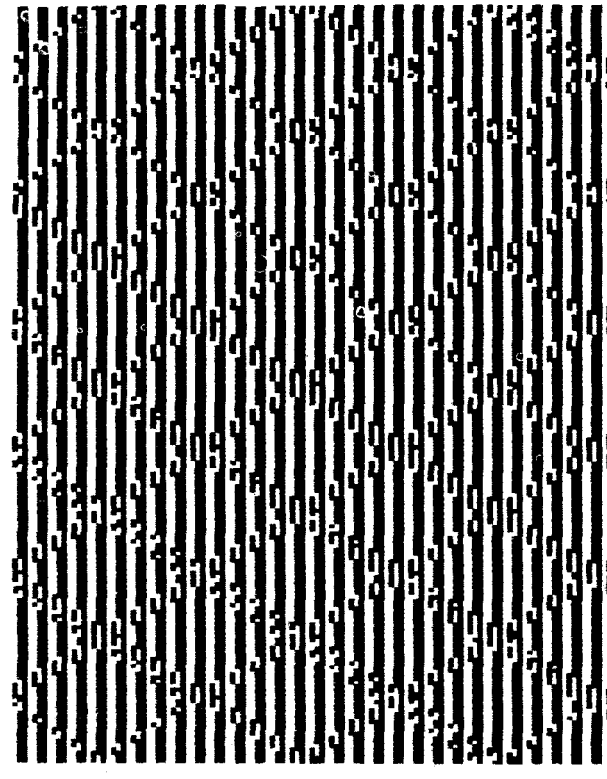
LD Generator and LD Curve



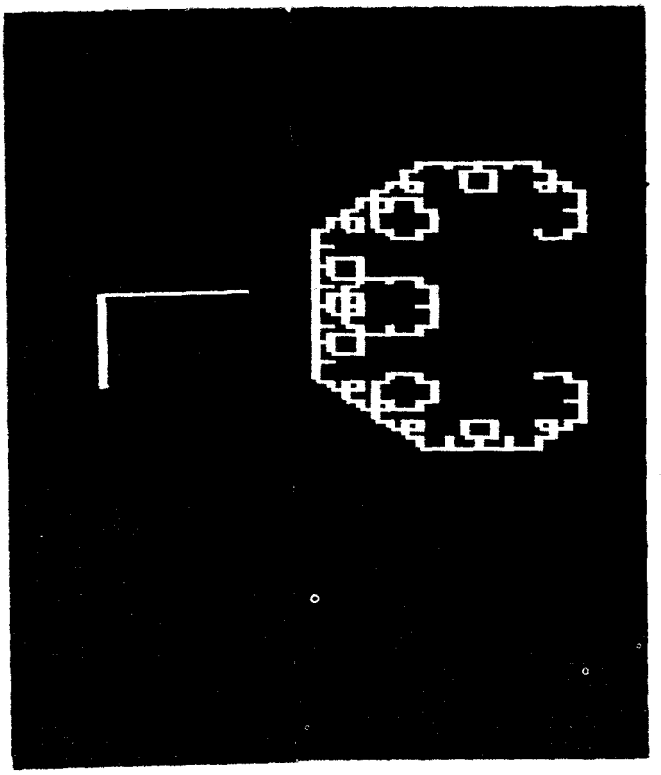
LD Plain



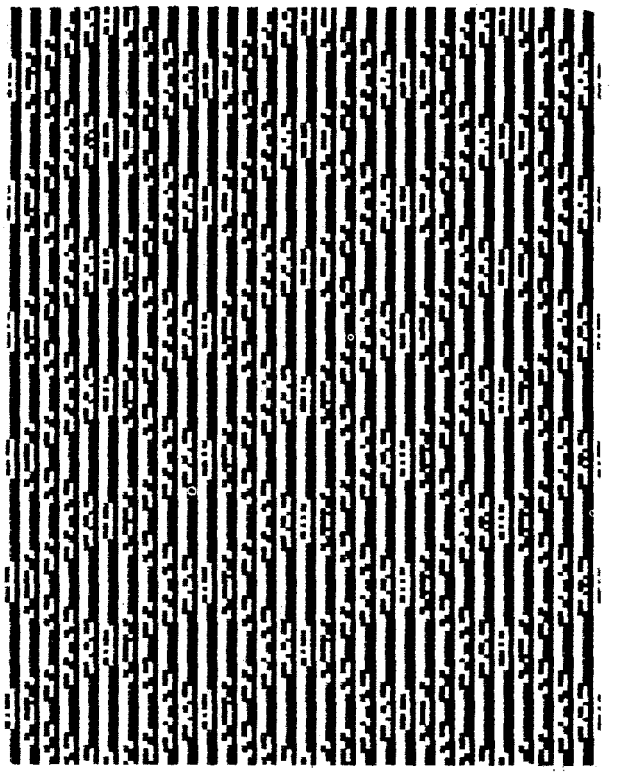
LD Network



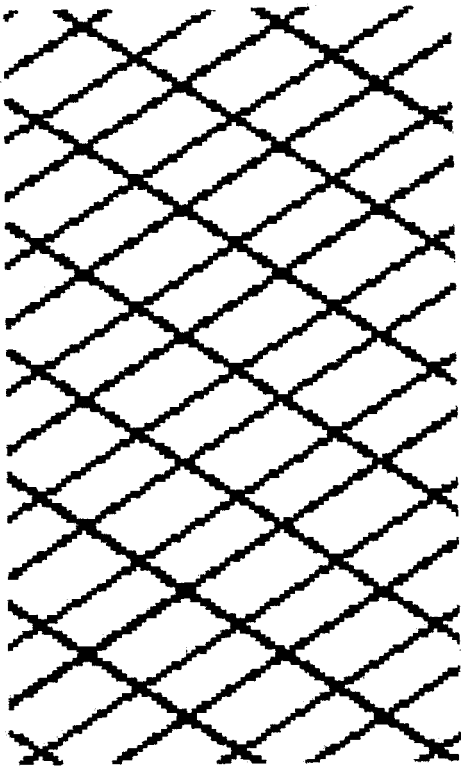
LD Twill



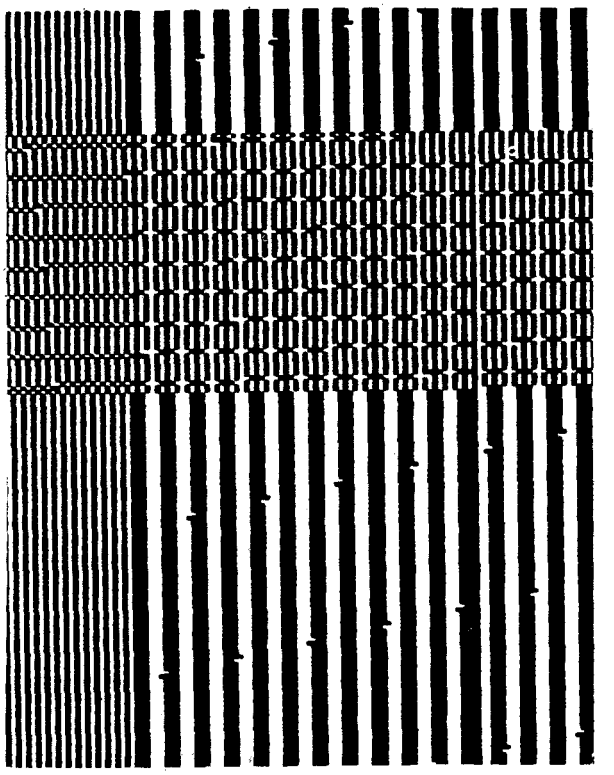
C2 Generator and C2 Curve



C2 Twill



C2 Network



C2 Plain

Theoretical Perspectives

A survey of the procedures themselves shows that all will produce either networks, twills, or plain weaves depending upon the variable inputs specified for :SIZE, :ANGLE, :LEVEL, and :PARITY. Preliminary notes encoded in each procedure cue the operator as to what range of variables will produce specific constructions.

Any one of the procedures will produce networks and twills if the cursor is first moved to a RT 45 position. Next a line length or :SIZE is specified which will wrap around the screen and make contact with itself to form diagonally intersecting lines. The background color is then changed to BG 2, BG 3, or BG 4. The diagonal lines interrupt the pixel on/off sequence and twills result. Alternatively, the background color may be changed as above for procedure execution.

Plain weave warp- or weft-faced fabric simulations are generated with line lengths shorter than those which will form repetitive diagonal intersections. The cursor position may or may not be changed prior to execution. In addition, a procedure may be repeated any number of times and the cursor rotated, or sent to HOME position at the center of the screen, between repetitions. The PU and PD commands must be used in conjunction with the HOME command so that unwanted lines are avoided.

A balanced, open, plain weave is possible with the GRID2 procedure. The most satisfactory inputs for this procedure are Repeat 2, :SIZE 500, :LEVEL 3, :PARITY 1. The result is a plain weave gauze which corresponds in "medium" print format to approximately 8 e.p.i. One could conceivably have a procedure which would generate a denser thread count. However, the density is limited by the width of the drawn line and the raster resolution. Close spacing of the lines would result in lines too close to be distinguished as separate entities.

While the currently available weave draft programs discussed earlier simulate manual drafting, LOGO permits the writing of procedures which simulate complex curves and serve as algorithms for many weave variations. The LOGO environment facilitates the writing of the interactive procedures which generate weave constructions based on a personal referent point. The artist-writer can determine within a relatively short period of time if a procedure results in variations of construction. Because the LOGO procedures can be tested rapidly, the writer is encouraged to alter procedures to see what other possibilities lie within the specific microworld.

It was in this manner, that the connections in appearance between networks and twills that are described above, was discovered.

While the LOGO procedures developed in this thesis result in textile construction simulations, translating these into woven cloth remains a tedious task. The LOGO simulations represent the appearance of complete cloth. In order to replicate this appearance a weave plan or draft must be completed. However, the most efficient way to accomplish this translation into woven cloth would be through a computer-controlled interface system configured with LOGO. As yet, such a system is unavailable.

While plain and twill weave LOGO simulations can be thecretically converted into woven cloth, satin weave appears not to exist in this micro-world. The absence of satin weave in this project may be due to the following: 1) The pixel points simulate visually warp- or weft-faced plain weave; 2) Diagonal lines when wrapped around the screen produce network or twill simulations by interrupting the pixel patterns; 3) The geometry of the satin weave differs from that of the plain and twill weaves in that no two adjacent threads may ever be woven at the same binding point; and 4) It is possible that the geometry of the weave itself contradicts the process used to generate images for this project.

The results of the thesis investigation suggest several areas for further investigation. For those readers with access to Jacquard looms, an in-depth exploration of the twill constructions based on these procedures is possible. An investigation as to the possibilities for network constructions and the relationship of these to textile constructions is indicated. The development of a computer-controlled loom system compatible with LOGO commands would facilitate the translations of LOGO-based simulations into woven cloth. Also, an exploration of other frames of reference, within the LOGO microworld, would determine if these constructions are related only to the complex curves.

APPENDIX

Listing_1
HIL

```
TO HIL :SIZE :LEVEL :PARITY
; ADAPTED FROM ABELSON AND DISESSA
; P. 98
;NETS : RT 45 HIL 2500 1 1
;TWILLS: RT 45 HIL 1500 2 1 BG 3
;REP: BG 3 RT 45 REPEAT 5 [HIL 100 1 1
RT 90]
IF :LEVEL = 0 THEN STOP
LT :PARITY * 90
HIL :SIZE :LEVEL - 1 :PARITY * 1
FD :SIZE RT 90 * PARITY
HIL :SIZE :LEVEL - 1 :PARITY
FD :SIZE
HIL :SIZE :LEVEL - 1 :PARITY
RT 90 * :PARITY
FD :SIZE
HIL :SIZE :LEVEL - 1 :PARITY * -1
LT :PARITY * 90
END
```

Listing_2
QUAD

```
TO QUAD :SIZE :LEVEL
;SEE MANDELBROT P. 50
;NETS: RT 45 QUAD 1000 1
;TWILLS: AS ABOVE THEN BG 3
;REP: BG 3 RT 45 [REPEAT 5 [QUAD 50 1 RT
90] BG 2
IF :LEVEL = 0 THEN FD :SIZE STOP
QUAD :SIZE :LEVEL - 1
LT 90
QUAD :SIZE :LEVEL - 1
RT 90
QUAD :SIZE :LEVEL - 1
RT 90
QUAD :SIZE :LEVEL - 1
FD :SIZE
LT 90
QUAD :SIZE :LEVEL - 1
LT 90
QUAD :SIZE :LEVEL - 1
RT 90
QUAD :SIZE :LEVEL - 1
END
```

Listing_3
C2

```
TO C2 :SIZE :LEVEL
;ADAPTED FROM ABELSON AND DISESSA P. 92
;FOR PLAITED CLOTH EFFECTS: BG 3 RT 33
;NETS RT 45 C2 5000 2
;TWILLS BG 3 RT 45 C2 3000 3
;REP: BG 3 C2 250 3
IF :LEVEL = 0 THEN FD :SIZE STOP
C2 :SIZE :LEVEL - 1
RT 90
C2 :SIZE :LEVEL - 1
LT 90
END
```

Listing_4
GRID2

```
TO GRID2 :SIZE :LEVEL :PARITY
;REPEAT 2 [GRID2 500 3 1] PRODUCES
;AN OPENWORK PLAIN WEAVE
;MAY BE USED AS GROUNDWORK FOR
;EMBROIDERY
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
Q4 :SIZE :LEVEL - 1 :PARITY
PU FD 10 PD
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
PU FD 10 PD
Q4 :SIZE :LEVEL - 1 :PARITY
PU FD 10 PD
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
PU FD 5 PD
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
Q4 :SIZE :LEVEL - 1 :PARITY
PU FD 5 PD
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
PU FD 5 PD
Q4 :SIZE :LEVEL - 1 :PARITY
PU FD 5 PD
Q4 :SIZE :LEVEL - 1 ;PARITY * (-1)
PU FD 5 PD
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
Q4 :SIZE :LEVEL - 1 :PARITY
PU FD 5 PD
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
PU FD 5 PD
Q4 :SIZE :LEVEL - 1 :PARITY
END
TO Q4 :SIZE :LEVEL :PARITY
;COMPARE WITH MANDELBROT P. 50
;FOR NETS TRY RT 45 Q4 5000 1 1
;FOR TWILL AS ABOVE THEN CHANGE BG
;FOR REP TRY BG 3 RT 33 Q4 500 1 1
IF :LEVEL = 0 THEN LT 90 * :PARITY FD
:SIZE STOP
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
Q4 :SIZE :LEVEL - 1 :PARITY
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
FD :SIZE
Q4 :SIZE :LEVEL - 1 :PARITY
Q4 :SIZE :LEVEL - 1 :PARITY
Q4 :SIZE :LEVEL - 1 :PARITY * (-1)
END
```

Listing 5
FLAKE

```
TO FLAKE :SIZE :LEVEL
;FLAKE IS ADAPTED FROM
;ABELSON AND DISESSA P. 91
;COMPARE WITH SNOW
;FOR NETS: RT 45 FLAKE 10000 1
;FOR TWILLS: FLAKE FLAKE 10000 1 BG 3
;FOR REPS: BG 3 REPEAT 3 [FLAKE 300 1]
REPEAT 3 [:SIDE :SIZE :LEVEL]
RT 120
END
TO SIDE :SIZE :LEVEL
IF :LEVEL = 0 THEN FD :SIZE STOP
SIDE :SIZE / 3 :LEVEL - 1
LT 60
SIDE :SIZE / 3 :LEVEL - 1
RT 120
SIDE :SIZE / 3 :LEVEL - 1
LT 60
SIDE :SIZE / 3 :LEVEL - 1
END
```

Listing 6
FUS

```
TO FUS :SIZE :LEVEL :PARITY
;AN ATTEMPT AT A TILING SNOWFLAKE
;SELF CONTACTS AT LEVEL 3
;SEE MANDELBROT P. 68
;NETS: RT 33 FUS 2000 1 3
;TWILLS: RT 45 FUS 1000 1 3
;REP: RT 45 FUS 250  1 3
IF :LEVEL = 0 THEN STOP
FUS :SIZE :LEVEL - 1 :PARITY * -1
LT 90 * :PARITY FD :SIZE * 2
FUS :SIZE :LEVEL - 1 :PARITY
RT 150 * :PARITY FD :SIZE * 1.75
FUS :SIZE :LEVEL - 1 :PARITY
LT 90 * :PARITY FD :SIZE
FUS :SIZE :LEVEL - 1 :PARITY * - 1
LT 60 * :PARITY FD :SIZE
FUS :SIZE :LEVEL - 1 :PARITY * - 1
RT 60 * :PARITY * - 1 FD :SIZE * 2
END
```

Listing 7
C4

```
TO C4 :ANGLE :SIZE :LEVEL :X
; A VARIATION ON C2
;NETS: C4 45 800 2 5000
;TWILLS: AS ABOVE THEN BG 3
;REP: BG 3 C4 33 50 4 100
SET
RT :ANGLE C2 :SIZE :LEVEL :X
PU SET FD :X + 1 PD
END
TO SET
PU SETX (-135) SETY ( -119) PD
END
```

Listing 8
LD

```
TO LD :SIZE :LEVEL
;ADAPTED FROM ABELSON AND DISESSA
;NETS: RT 45 LD 1500 3
;TWILLS: RT 45 LD 1500 3 BG 3
;REP: RT 33 REPEAT 4 [LD 300 1]
IF :LEVEL = 0 THEN FD :SIZE STOP
LD :SIZE :LEVEL - 1
LT 90
RD :SIZE :LEVEL - 1
END
TO RD :SIZE :LEVEL
IF :LEVEL = 0 THEN FD :SIZE STOP
LD :SIZE :LEVEL - 1
RT 90
RD :SIZE :LEVEL - 1
END
```

Listing 9
NSQ

```
TO NSQ :SIZE :LEVEL
;A CESARO'S TRIANGLE SWEEP
;SEE MANDELBROT P. 64
;NETS: RT 33 NSQ 1000 1
;TWILLS: BG 3 RT 45 NSQ 2000 1
;REP: BG 3 NSQ 500 1
REPEAT 2 [NQS :SIZE :LEVEL ]
RT 170
END
TO NQS :SIZE :LEVEL
IF :LEVEL = 0 THEN FD :SIZE STOP
NQS :SIZE :LEVEL - 1
LT 85
NQS :SIZE :LEVEL - 1
RT 170
NQS :SIZE :LEVEL - 1
LT 85
NQS :SIZE :LEVEL - 1
END
```

Listing 10
SNOW

```
TO SNOW :REPEAT :SIZE :LEVEL
;ADAPTED FROM ABELSON AND DISESSA
;FOR NETS TRY RT 45 SNOW 3 5000 2
;FOR REPS TRY BG 3 SNOW 3 500 1
REPEAT :REPEAT [SIDE :SIZE :LEVEL RT
120]
END
TO SIDE :SIZE :LEVEL
;ADAPTED FROM ABELSON AND DISESSA P. 91
;SEE MANDELBROT P. 43
IF :LEVEL = 0 THEN FD :SIZE STOP
SIDE :SIZE / 3 :LEVEL - 1
LT 60
SIDE :SIZE / 3 :LEVEL - 1
RT 120
SIDE :SIZE / 3 :LEVEL - 1
LT 60
SIDE :SIZE / 3 :LEVEL - 1
END
```

Listing 11
AK

```
TO AK :SIZE :LEVEL :PARITY
;SEE MANDELBROT P. 48
;NETS: RT 45 AK 1000 1 1
;TWILLS: BG 3 RT 45 AK 1000 1 33 BG 5
;REP: AK 100 2 3.75
IF :LEVEL = 0 THEN RT 90 * :PARITY FD
:SIZE STOP
AK :SIZE :LEVEL - 1 :PARITY  * (-1)
AK :SIZE :LEVEL - 1 :PARITY
AK :SIZE :LEVEL - 1 :PARITY
AK :SIZE :LEVEL - 1 :PARITY * (-1)
FD :SIZE
AK :SIZE :LEVEL - 1 :PARITY * (-1)
AK :SIZE :LEVEL - 1 :PARITY
AK :SIZE :LEVEL - 1 :PARITY * (-1)
END
```

Listing 12
KS

```
TO KS :SIZE :LEVEL
;SEE MANDELBROT P. 139
;NETS: RT 45 KS 2000 1
;TWILLS: AS ABOVE THEN BG 3
;REP: BG 3 RT 45 REPEAT 3 [KS 45 2 RT
90] BG 2
IF :LEVEL = 0 THEN FD :SIZE STOP
KS :SIZE :LEVEL - 1
LT 90
KS :SIZE :LEVEL - 1
RT 90
KS :SIZE :LEVEL - 1
RT 90
KS :SIZE :LEVEL - 1
LT 90
KS :SIZE :LEVEL - 1
END
```

Listing 13
S2

```
TO S2 :SIZE :LEVEL :PARITY
;COMPARE WITH MANDELBROT P. 142
;NETS: RT 45 S2 2000 1 6
;TWILLS: SAME AS ABOVE THEN BG 3
;REP: BG 3 RT 45 S2 400 1 6
IF :LEVEL = 0 THEN FD :SIZE STOP
S2 :SIZE :LEVEL - 1 :PARITY * - 1
RT 45 * :PARITY
S2 :SIZE :LEVEL - 1 :PARITY * - 1
RT 45 * :PARITY
S2 :SIZE :LEVEL - 1 :PARITY * - 1
RT 45 * :PARITY
S2 :SIZE :LEVEL - 1 :PARITY
END
```

A SELECTED BIBLIOGRAPHY

Abelson, Harold and diSessa, Andrea. Turtle Geometry: The Computer as a Medium for Exploring Mathematics. Cambridge, Mass.: MIT Press, 1980.

Abelson, Harold and Klotz, Leight, Jr. LOGO for the Apple II: Technical Manual. Stony Brook, N.Y.: Krell Software Co., 1982.

D'Harcourt, Raoul. Textiles of Ancient Peru and Their Techniques. Seattle: University of Washington Press, 1962.

Lourie, Janice. Textile Graphics/Computer Aided. New York: Fairchild Publications, 1973.

Mandelbrot, Benoit B. The Fractal Geometry of Nature. San Francisco: W.H. Freeman, 1982.

Papert, Seymour. Mindstorms: Children, Computers, and Powerful Ideas. New York: Basic Books, 1980.

Thornberg, David D. Discovering Apple Logo. Reading, Mass.: Addison-Wesley, 1983.

Winston, Patrick Henry and Horn, Berthold Klaus Paul. LISP. Reading, Mass.: Addison-Wesley, 1981.

Wolfgang, W.G. "A Computer Program to Generate Weave Structures." Computer Graphics and Art. November 1976, pp. 10-17.

Computer-Aided Printmaking

Isaac Victor Kerlow

(School of Visual Arts)
26 Gramercy Park South
New York, NY 10003

Abstract

Computer-Aided Printmaking describes some of my recent experiences in transfering computer-generated wireframe and shaded images to a variety of traditional printmaking media. The main steps of the generating and transfering processes are reviewed.

Categories: Computer Graphics, Printmaking, Image Processing, Fine Arts, Hybrid Hardcopy, Seed Image, Color Space.

1. Introduction

Computer-generated images can be presented in two different environments: the *softcopy* and the *hardcopy* environments. In general terms we can say that softcopy is the *natural* way of outputing computer images because they are displayed in real time on the computer's screen. Hardcopy output of computer-generated images usually takes place on a two-dimensional physical support such as a paper printout or a photograph.

Among the most popular hardcopy output techniques are the ink jet, dot matrix and electrostatic printers, and pen plotters (ref. 1). Each one of these technologies has unique technical characteristics that produce very specific visual qualities. All of these techniques are computer-driven and "dump" the contents of the frame buffer directly onto the output support.

Printmaking techniques traditionally used in the fine arts include metal etching and engraving, lithography, silkscreen and woodblock printing. Detailed explanations of each one of these techniques, and others, can be found in refs. 2 and 3. The procedures involved in the metal etching technique are described below.

Etching techniques involve a metal plate (usually copper) that is used as a matrix for making multiple copies of an image. The image must be etched on the plate, usually by bathing the plate in acid. In the most common etching technique, called *hardground*, the plate is covered with an acid-resistant varnish (fig. 1). The varnish is removed by the artist from the areas that are to be etched. This is done by scratching the varnish off the plate with special drawing tools. The plate is then inmersed in an acid bath where the copper is exposed and etched only in the areas where the varnish was removed. The acid will etch the plate in relation to the time that the plate is left inside the acid. Once the etching process is over the varnish is removed from the plate and the depressions created by the acid in the plate are filled with ink. Finally, the plate is covered with paper and both are rolled through a high pressure press that transfers the ink contained in the plate onto the paper.

My interest in both printmaking and computer image generation techniques led me to experiment with *hybrid* output techniques that would combine the best characteristics of both computer image generation and traditional printmaking. A conceptual similarity between computer image generation and printmaking convinced me that this hybrid techniques could be successfuly implemented: the idea of making several copies from a unique matrix is shared by several media including three-dimensional computer imaging, photography and printmaking. Traditional printmaking techniques employ one or several matrices (i.e. metal plates) to make copies of an image. Similarly, the computer uses the numerical description of a three-dimensional object as a matrix for producing different views and renderings. It seemed natural then that the results of a numerical matrix, the computer-generated images, could be transferred onto a printmaking matrix for creating new versions.

My imaging interests and the type of work that I do require the ability of combining images from different sources. This *compositing* can already be done entirely with a computer system by scanning every image and manupulating it with the system. But the types of imaging operations I am interested in are so varied that it would require a very powerful and flexible computer

CH2218-6/85/0000/0057$01.00 © 1985 IEEE

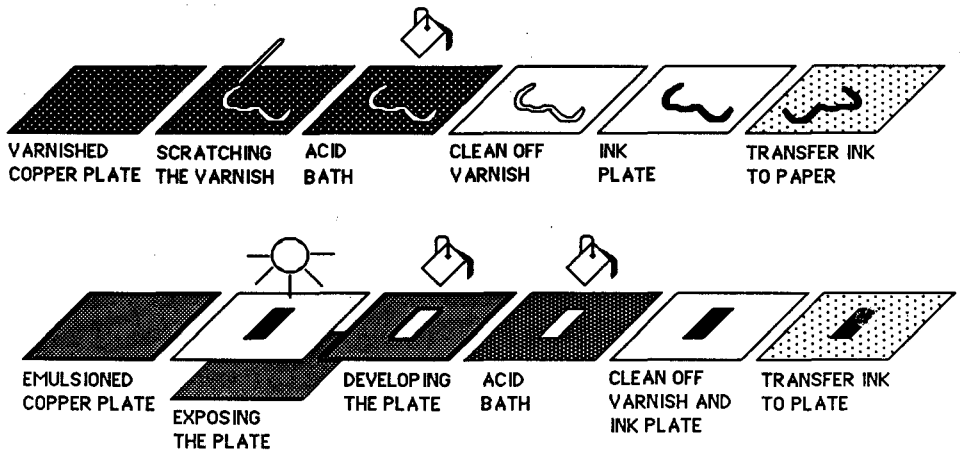


Figure 1. The etching process of a metal plate, both manual (top) and photographic (bottom).

system to perform them smoothly and flawlesly (ref. 4). Lacking such an *image-composer* system, I decided to take the hybrid approach. The advantages of this approach are varied:

a. Images from different sources can be combined while retaining the visual qualities and resolutions of different technologies. I am interested, for example, in preserving the subtle and irregular width of a line created with an ink pen, or the specific randomness of a dry brushstroke. Either one when scanned into today's computers would be transformed into something else. It is not a matter of better or worst technologies, it is a matter of *different* technologies.

b. Limited editions or unique pieces can be created on paper of archival quality and acid free.

c. Color separation procedures can be implemented in software, thus eliminating the need for the time consuming and expensive task of manually or photographically creating a color separation.

2. Planning the Hybrid Image

The creation of hybrid images considers the computer-generated image on the monitor only as a departure point and not as a final product. It is clear that the final result of this process, an etching print, differs substantially from the initial image on the computer's screen. But the goal in this case is not to achieve a perfect reproduction of the screen image with a high fidelity medium such as custom color photography, but to record a *seed* image that will transfer optimally to a new media. The hardcopy obtained from the monitor is not intended to be an exact replica of the computer-generated image but the beginning of a new process.

Some planning is required to make all the components of a hybrid image fit together, not only in terms of composition but also in terms of contrast, brightness, color and texture. I do not follow a rigid methodology for planning the final image. Sometimes I sketch the hybrid image before I even start generating any of its components, in other occasions I develop the hybrid image as I assemble already existing components. For example, if I have a drawing that I like I might develop a computer-generated image to go with it.

Planning the hand drawn images does require a sketch or two before the execution of the final drawing. Planning the computer-generated images is usually a more complex process and requires a larger amount of tests. The following are some of the checkpoints that I usually go through when planning or reviewing a computer-generated image for transfering purpouses.

a. Proportion. The computer's screen has a fixed proportion (1 to 1.3) that might be different from the proportion of the final hybrid image. This can be corrected by selectively cancelling areas of the screen to create areas with new proportions. These operations though, might also reduce the overall image resolution.

b. Resolution. This refers to the texture and visual quality of the computer-generated image. I usually work with medium resolution images (512x512 or 640x480 pixels), some of which are antialiased. Resolutions of 4000x4000 pixels surpass the resolution of film.

c. Scale. If the computer-image will be enlarged greatly, the loss of detail can be minimized by using a larger film for recording the seed image. For example, 4"x5" transparencies instead of 35 mm slides.

d. Brightness and contrast. As transfer generations increase, so does image contrast. This is evident by the darkening of dark areas and the lightening of light areas. A simple solution to this problem consists in over-exposing the

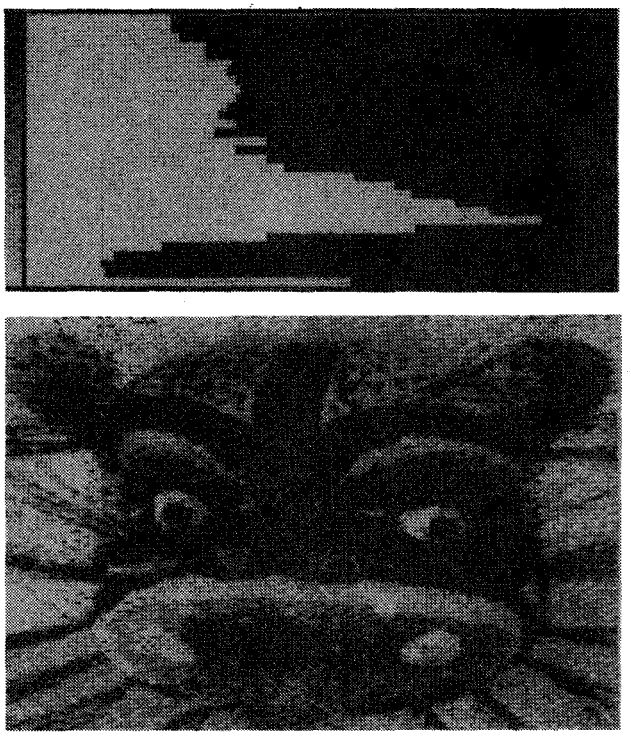


Figure 2. A histogram is a graph that can provide useful information about the distribution of gray values in an image.

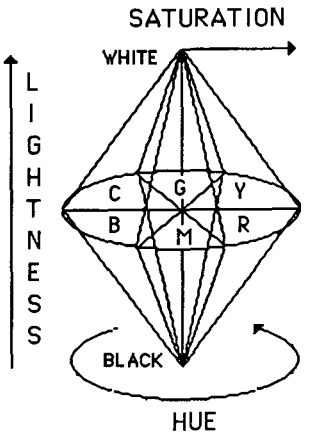


Figure 3. The HSL Color Space is useful for balancing the color and brightness of an image.

dark areas and under-exposing the light areas. A better solution involves using histograms for balancing the contrast and brightness levels of an image (fig. 2). The HSL (hue, saturation and lightness) color system is very useful for balancing brightness and color of an image (fig. 3).

e. Image content. Some techniques are better than others for representing a specific subject. One must select the most appropriate one according to personal taste or stylistic guidelines.

f. Color. Dealing with computer graphics and printmaking implies dealing with two different *color spaces*. The chromatic range of additive color systems (computer screen) is much greater than the range of substractive color systems (printing inks). The CYM (cyan, yellow and magenta) pigments could be stretched to cover more RGB (red, green and blue) colors, but a more straightforward way of avoiding this range imbalance is reducing the active color range of the computer.

g. Matte generation. Mattes are necessary for compositing or merging two or more computer images together or with images from other sources. The use of a simple matte is illustrated in fig. 4.

3. Generating the Computer Image

I use several types of computer-generated images in my hybrid works. Some are created with two-dimensional techniques and some are created with three-dimensional techniques, some are in raster format and some are in vector format. Most of my two-dimensional images are created with custom graphics routines written in BASIC or Pascal and some with commercial software (ref. 5). The procedures involved in creating a two-dimensional image are very similar to traditional drawing, painting and photography.

The basic steps for creating a three-dimensional image are illustrated in fig. 5, and the entire process is described in detail in ref. 6. I use the word *three-dimensional* to refer not to stereo or holographic images, but to two-dimensional images of three-dimensional objects and environments created in the virtual mathematical space of the computer's memory. The core of the software that I use for displaying three-dimensional shaded images is described in refs. 7 and 8.

Some of the topics that interest me in three-dimensional image generation include: the creation of complex objects with handmade-like qualities, random high frequency textures and transparency, and the compositing of vector and shaded images. Objects with handmade-like qualities can be best defined with one of two geometry description methods: contour digitizing or distorted geometrical primitives. Random textures can be created with a color aliasing method I describe in ref. 9. The advantage of using these textures in hybrid images is that they hold the shading detail very well when transferred to printmaking media (last image in fig. 5).

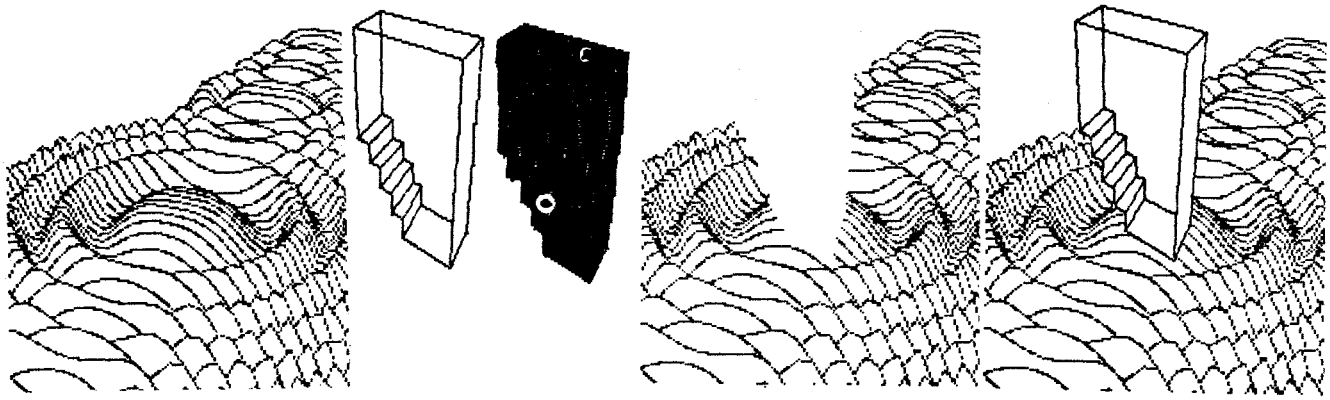


Figure 4. Matte generation allows compositing several images into one. In this example a background (a) and a foreground object (b) are merged together using a matte (c). The reserved space created by the matte in the background (d) is filled in with the foreground object (e).

4. Transfering the Computer Image

Once the computer-generated image exists in the computer, either as an image on the screen or as data in memory, it can be transferred onto the printmaking printing plate. There are two ways of transfering the image onto the plate: photographic transfer and direct output.

The photographic transfer process starts with a slide or photograph of the computer-generated image (taken with a camera off the monitor or with a film recorder), or a computer printout or plot (fig. 6). This initial image constitutes the first step in a transfering process that involves a black and white continuous tone original, a halftone screened paper print, a high contrast positive film, a metal plate with photosensitive emulsion, and a state proof.

The initial image is used to create the film positive that will be exposed onto the plate. Shaded continuous tone images require the use of halftone screens before they can be transferred onto the printmaking matrix, but line drawings and high contrast images can be transferred without them. Halftone screens of eighty five to one hundred dots per inch give excellent image quality with an almost unnoticeable dot pattern.

The film positive is exposed onto the plate with a light table (second part of fig. 1). The photo-sensitive emulsion on the plate hardens when exposed to light. When developed, the unexposed portions of the emulsion dissolve, leaving the dark areas in the film positive exposed to the acid. Before etching the plate it is necessary to apply a very fine *aquatint*, or grainy screen, that will better hold the transferred image. It is very important that the aquatint screen is fine enough so it does not cause interferance or *moire* patterns with the halftone screen (fig. 7).

One of the major drawbacks of photographic transfer is the deterioration suffered by the image through the various

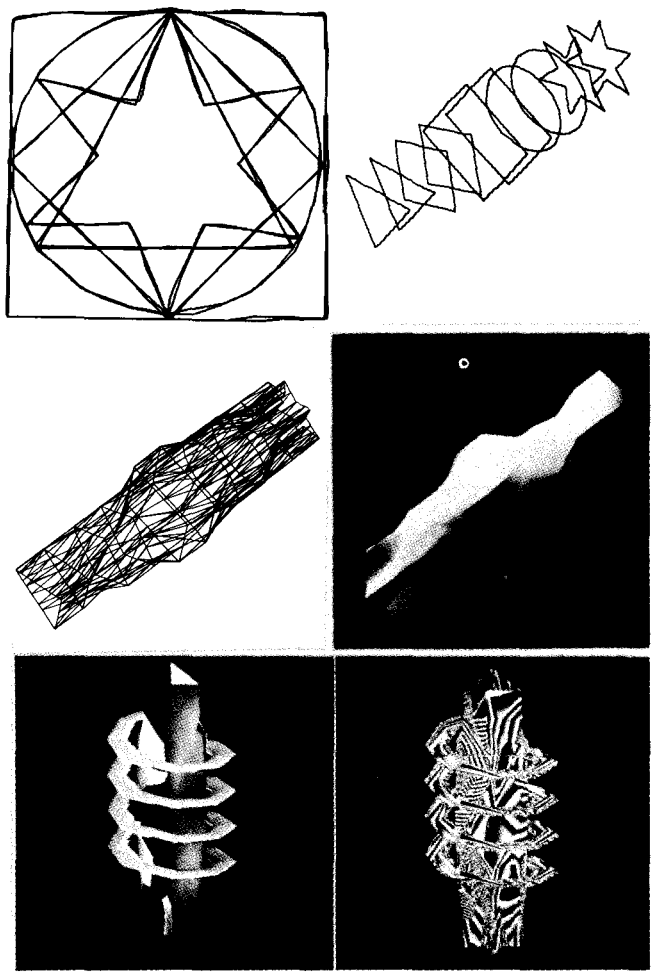


Figure 5. The basic steps for creating a three-dimensional image. Conceptualization and design require that the general characteristics of the object are sketched. Modeling the object includes defining a numerical database and positioning the object in three-dimensional space (a, b). The object illustrated here was described by manually digitizing two-dimensional cross-sections. The initial wireframe structure serves as a basis to create surfaces and volumes (c). Rendering includes the removal of hidden surfaces, the lighting and shading of the model (d), and the creation of color and texture (e, f).

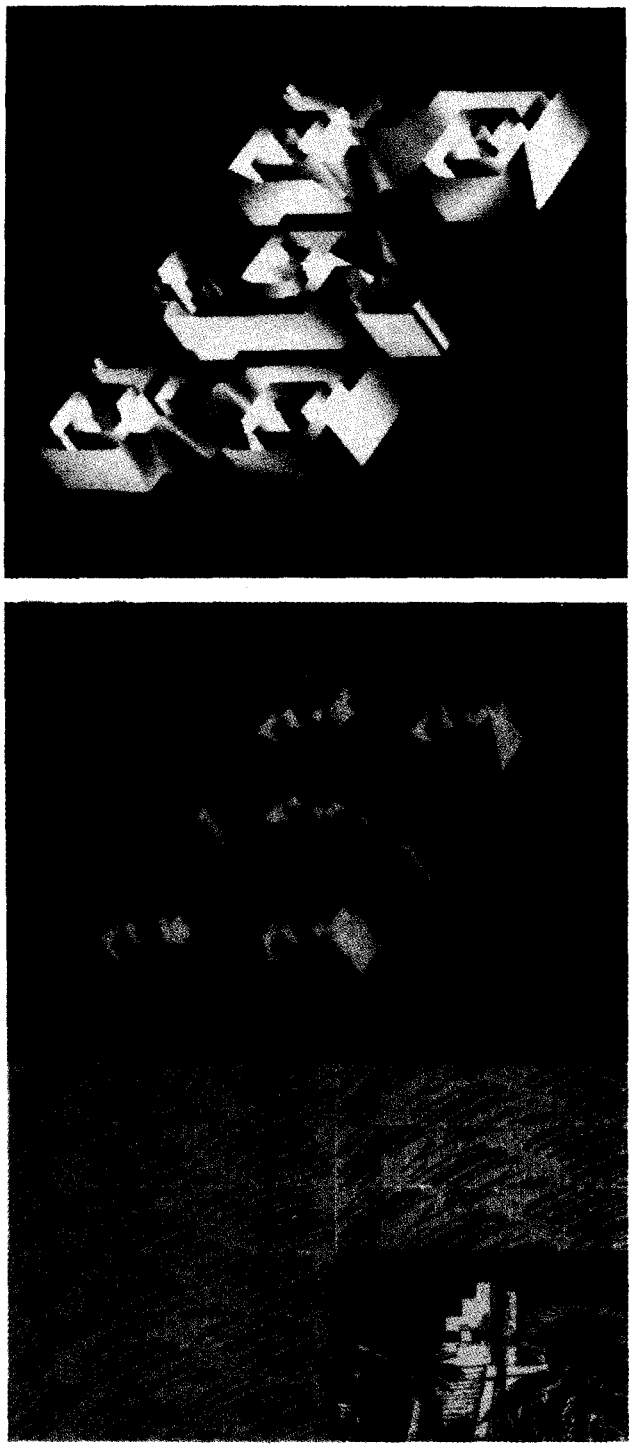


Figure 6. Close-up of an image on the computer's screen and the same image contained in a hybrid etching. The etching is based on the concepts of architectural patterns and "windows" that show parallel processes. It is entitled *PYRAMID IN BLACK AND WHITE NUMBER ONE*, and measures 16" by 20".

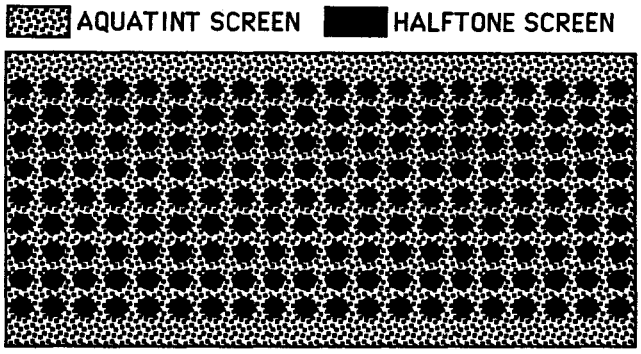


Figure 7. Size relation between the aquatint screen and the halftone screen.

Figure 8. Edge extraction (a) can be used to enhance image sharpness in fuzzy images (b).

generations needed to achieve the image transfer. Only after months of trial and error have I learned that the best solution to this problem is a strict quality control during all the photographic process, and that some apparently insignificant technical details can make or destroy an image.

When a dot matrix printout is used as the seed image a lot of detail is lost in the making of the film positive because the film cannot pick up the bluish tones of the ink used in most dot matrix printers. Furthermore, the dot patterns are often irregular, especially when magnified several times. The best technique for avoiding excesive loss of detail in dot matrix printouts through the transfering process is to make a direct positive enlargement on paper first and from that make the film positive. In that way the original dots are solid black and can be registered by the film positive.

Edge extraction is an image processing technique that can be used to compensate for the fuzzy edges of an image. The edges of an image can be overprinted on the full gray-scale tone image to improve image sharpness (fig. 8).

Another technique that helps keep image detail through several generations is the use of a negative film (kodalith) to

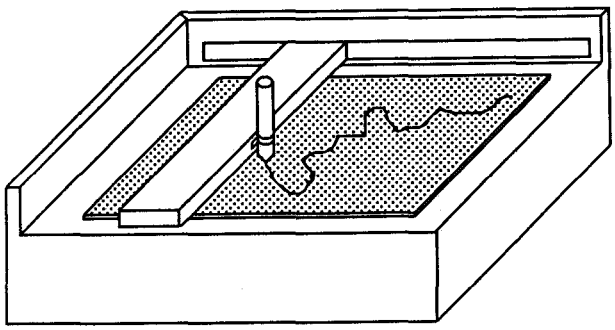


Figure 9. Plotting a computer-generated image directly on a metal plate.

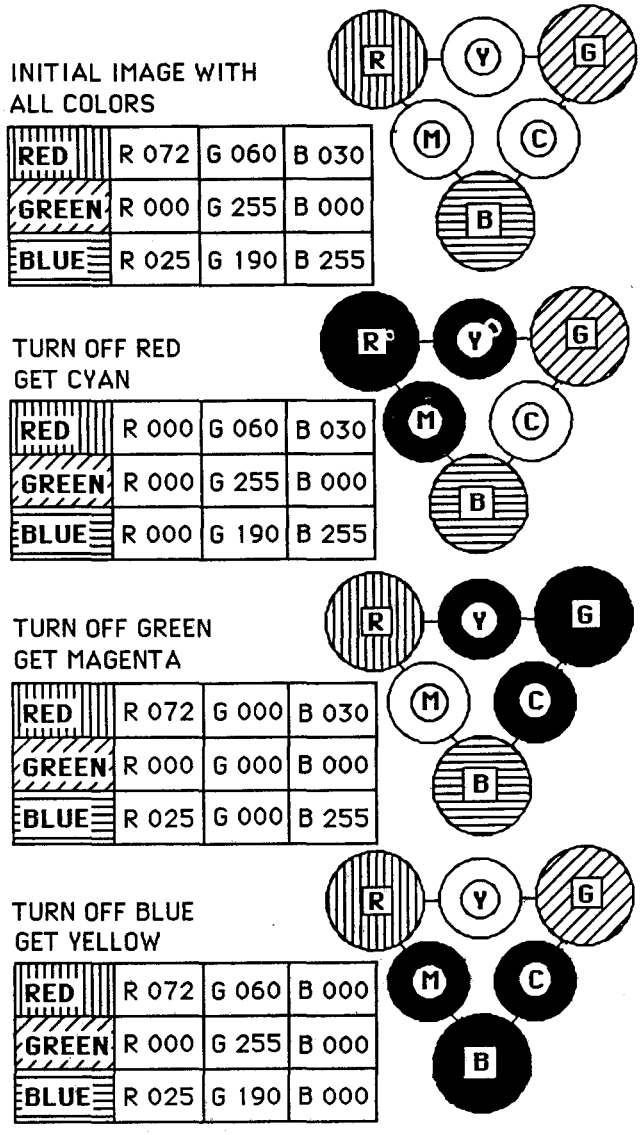


Figure 10. Color look-up table manipulation allows simple color separation procedures.

record the original artwork, and to create the film positive from it. Eventhough this proccess is almost five times more expensive than recording the seed image directly onto film positive, it is also much more precise and holds more detail.

The effectiveness of transfering a computer-generated image onto the plate by direct output depends on how sophisticated the output tool is. I use a very simple device that consists of a pen plotter where the pen has been substituted with a needle that scratches the protective varnish on the plate (fig. 9). If wireframe images are required it is necessary to translate the image from a bitmap format to a display list format in order to avoid broken lines.

Monochromatic images require correct brightness and contrast levels. Multichromatic images require that plus a color separation process. Color separations can be created for true four-color process or for flat colors. The flat color process can be implemented with mattes like the ones mentioned above and illustrated in fig. 4. When transfering the computer-generated image onto the plate photographically, four-color separation procedures can be implemented in software or with color filters. When transfering the computer-generated image directly onto the plate color separation must be achieved under software control, before the image is output onto the plates. There are many methods for direct digital color separation, one that I use often is based on color table manipulation and is illustrated in fig. 10. Another method of software-based color separation even creates the final film screens under computer control (ref. 10). The procedure for doing a color separation with color filters is illustrated in fig. 11.

5. Making the Final Hardcopy

Once the computer-generated images have been transferred onto the printmaking matrix, they can be combined with images from other sources such as drawings and continuous tone photographs. I personally find this type of collage activity very interesting, and think that it can enrich the visual meaning of computer-generated images.

The final hybrid image can be transferred onto paper after the matrix has been inked and passed through the press. It is possible to obtain multicolored prints if several plates were prepared with color separation procedures.

Factors that greatly affect the final quality of the print include: the resolution of the aquatint that holds the halftone screen and the resolution of the screen itself, the degree of ink transparency, and the texture and color characteristics of the final support, usually paper.

6. Large Scale Printmaking

A type of computer-generated print that deserves special attention is the large scale print. Its unique qualities include its larger-than-monitor proportions, the fact that it often reveales the raster texture of computer-generated shaded images, and the existance of large fields of color and visual matter.

I have used silkscreen and large scale ink-jet printing for creating large scale prints. Silkscreening computer-generated images is a very flexible technique that allows very high resolution, a wide choice of output support and very flexible positioning of the image on the support. The ink-jet printing process that I have used is the *Scanamural* process provided by the 3M Company of Minneapolis, Minnesota (fig. 12). This process is a type of photographic transfer because it scans a computer-generated slide into the computer's memory and then uses four ink jet guns to transfer it onto a canvas that is rolled around a drum. This process is close to printmaking because it uses a matrix for generating the image, but it is close to painting too because of its scale, texture and choice of materials.

An alternate, low-cost method for creating large scale computer-generated images makes use of small printouts created with dot matrix, ink-jet or electrostatic techniques. Each one of these printouts is an enlargement of a small section of an image and when assembled together they function as modules of a larger piece.

Conclusions

For the last two years I have experimented with methods for reproducing computer-generated images in a variety of printmaking techniques. My experiences include both photographic and computer-controlled image manipulation, color separation and transfering procedures.

There are still many technical details that remain to be solved before this hybrid imagemaking approach reaches its maturity. Technical issues concern me as an artist and a programmer because they affect directly the creative possibilities of the whole hybrid process. Nevertheless, artistic creation should not only be judged by its technical virtuosity but also, and most importantly, by its overall creativity and excellence.

Most of these procedures can be easily implemented in microcomputers. Therefore, I hope to see more of this type of work in the near future.

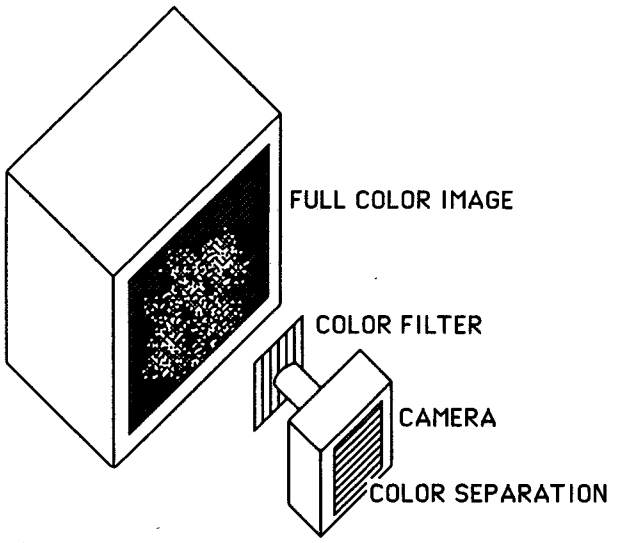


Figure 11. Color separation procedures with color filters. Cyan is separated with the red filter, yellow with the blue filter, magenta with the green filter, and black with a modified orange filter.

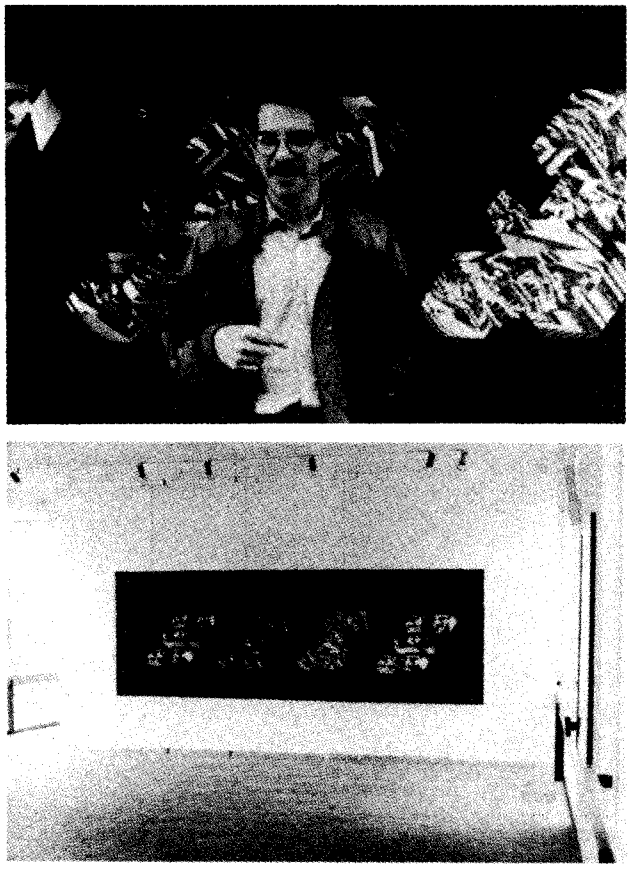


Figure 12. *QUARTET (FOUR RANDOM TEXTURES)*, a 5x20 ft. computer-generated work on canvas created with the Scanamural process. Detail with human scale (a) and full size (b).

References and Notes

1. Foley, James, and Andries Van Dam. *Fundamentals of Interactive Computer Graphics*. Reading, Mass.: Addison-Wesley, 1983, pp. 95-135.

2. Staff, Donald, and Deli Sacilotto. *Printmaking: History and Processes*. New York: Holt, Rinehart & Winston, 1978.

3. Brunner, Felix. *A Handbook of Graphic Reproduction Processes*, 5th ed. Teufen, Switzerland: Arthur Nigli Ltd., 1975.

4. The hardware that I have used for these experiments includes several microcomputers (Apple II, Macintosh, IBM PC) and one minicomputer (DEC VAX 11/780). Tasks implemented on these systems include two-dimensional paint, image processing and compositing, color separation, and three-dimensional wireframe and shaded modeling.

5. Some of the commercial software that I have used for two-dimensional image manipulation with microcomputers includes *SuperScan* by Magna Soft, *Lumena* by Time Arts, *Zoom Graphix* by Phoenix Software, and *Special Effects* by Penguin Software.

6. Kerlow, Isaac Victor. "The Computer as an Artistic Toool." *Byte*, September 1984, pp. 189-206.

7. Kropf, Noel and Cyrus Levinthal. "Serial Section Reconstruction Using CARTOS," in *The Microcomputer in Cell and Neurobiology Research*, ed. R. Ranney Mize. New York: Elsevier Science Publishing Co., in press.

8. Christiansen, Hank, and Mike Stephenson. *Movie.BYU*. Provo, Utah: Brigham Young University Press, 1983.

9. Kerlow, Isaac Victor. "The Maya Project." *Computer Graphics World*, August 1984, pp. 30-36.

10. Image Set Corp. in Sunnyvale, Calif. provides a digital color separation service that accepts image formats of the most popular microcomputers. The color separations and color correction are created with software and the color separation data is output with a Scitex *Response 350* system on film in a variety of specs.

Acknowledgements

The author wishes to thank several people that have contributed to this work: Paul Marcus for impecable editioning of my etchings, Enrique Cataneo for impecable editioning of my silkscreens, and Ron McNeil for suggestions on RGB to CYM color conversion schemes.

COMPUTERS VIEWING ARTISTS AT WORK

Joan L. Kirsch
Russell A. Kirsch

The STURVIL Corporation
Box 157, Clarksburg, MD, 20871

Abstract

Our title suggests an Artificial Intelligence approach to the use of computers in the fine arts. We consider computers to have capabilities beyond the utilitarian ones of aiding in art making. Rather, we will investigate the possibility of computers seeing, even understanding, significant form in art. This understanding cannot rise autonomously, but must be the product of careful tutelage by artists, critics, and historians. A powerful tutorial mechanism to use for computers to learn about art is the picture grammar, which allows large classes of compositional structures to be described to a computer by the scholar who has a deep understanding of the art works. In this paper, we illustrate how a machine can be taught the compositional structure of the paintings of the contemporary artist Richard Diebenkorn. With such grammatical instruction, the computer can analyze existing paintings, generate new ones of the same style, and provide a beginning to a computational theory of style.

Formalism in art and computers.

Computers are devices for manipulating symbols in formal systems. This characterization is broad enough to include uses ranging from numerical mathematics to computer graphics, although for graphics in art making, the formal properties must be supplemented by the physical characterization of output devices. But such a stretching of the definition is not necessary if we consider the use of computers in the formal description of art works. Here, the notion of formalism as it is understood in computer science and in art criticism come into reasonably close correspondence.

The formal analysis of an art work deals with its hermetic visual properties, such as color, line, shape, materials and their arrangements, the so-called plastic elements. By contrast, there are extrinsic properties like feelings, stories, metaphor which are not so easily described to a computer. These are examples of what are often called the expressive properties of the art. Another common version of this distinction is the dichotomy between classical and romantic art. At this point, we can deal only with formal qualities and not extrinsic ones.

Actually, a bold body of criticism ranging from Roger Fry and Clive Bell to Clement Greenberg maintains that an esthetic response to an art work depends solely on a purchase of its formal properties. Thus, describing the formal structure of art works to computers would seem a valid approach for making precise, to people and computers, the understanding of art.

Traditional methods of formal analysis have often been cumbersome. For example Loran[1] analyzed Cezanne's compositions with the use of schematic diagrams to account for space, planes, lines, volumes, and the other plastic elements. But a contemporary reader immediately realizes that such an analysis could be made more precise and complete with current computer graphic tools.

However still more powerful tools are available. They are drawn from the fields of image processing, pattern recognition, and computational linguistics, all parts of artificial intelligence. These tools arose [2] in 1964 when it was realized that the use of grammars for describing language could be generalized to describe images. Later, Stiny [3] further generalized these ideas, introducing the idea of a shape grammar. It is this form, slightly modified, that we have used in the study of painting.

Richard Diebenkorn's Ocean Park Paintings

Richard Diebenkorn is one of the most important and respected contemporary American painters. Between 1967 and 1983 he painted about 135 large oil paintings inspired by the look of the Ocean Park area of Santa Monica, California, where he has a studio. These paintings appear, on first inspection, to be largely geometric and, hence, to lend themselves to the kind of formal analysis that we find most immediately possible. On further examination, one sees subtle and complex compositions built of the apparently informal and rich handling of layered colors, lines, and textures. Thus, a formal analysis restricted to the geometric qualities is, at best, a beginning of a complete grasp of the paintings. It is this beginning that we have attempted.

A Diebenkorn Ocean Park Grammar

The grammar we have written (Kirsch 1985) is a modified shape grammar. It contains 42 production rules, most of which offer options for how to subdivide regions of the painting. Some of these rules are recursive insofar as they produce subdivisions that, in turn, reinvoke the original rules. Such recursion enables the grammar to account for an infinite number of distinct compositions.

Although the grammar does not account for color, for example, it provides places where future rules for color may be added. This is in keeping with the practice, in computational linguistics, of providing 'hooks' for the addition of features to an account of a corpus which is usually (in the case of language) arbitrarily large. In the case of Diebenkorn's painting, the complexity seems, at this early stage, arbitrarily large.

A shape grammar is a description of structure. For our grammar, it is a description of the linear compositional structure of this class of paintings. In any case, the grammar is equivocal with respect to two important possible uses. It may be used to analyze or to synthesize. The purpose of analysis is to reveal the structure of the paintings being described. Such an analysis can then be viewed by scholars to discover structure not necessarily evident in the finished work. The analysis is like the parse of a sentence. It conveys a significant part of the content of the object being analyzed. To see how such an analysis appears, Fig. 1 shows one of Diebenkorn's Ocean Park paintings, number 111 of 1978. Fig 2. shows the analysis assigned by the grammar to the painting. The rule numbers listed correspond to the grammar which assigns the analysis. Thus, although we do not list the grammar rules here, we can nevertheless see some of the structural analysis assigned to the painting by the grammar. We see several rules applied multiple times: rules 11,36,20, etc. These are some of the recursive rules. We also see how the grammar organizes the area of the painting. Finally, we see how the highest level organization of the painting influences the lower levels. This is denoted by the notation /S that appears at the first level in OP/S, which denotes a kind of 'suburban' landscape as opposed to the'rural' and 'urban' alternatives provided by the grammar. In some of the final regions, like w/s we see the influence of this high level choice manifest in how the lower level regions remain, at this stage in the grammar. These properties are used, by later grammars, in choosing colors and other properties. These later properties represent more'surface' characteristics of the painting. The present grammar accounts for the 'deep structure'.

When the final analysis is produced by the grammar, it results in Fig. 3. We can compare this diagram with the original painting represented in Fig. 1, and see how the construction lines (the pentimenti) are both preserved by Diebenkorn, and generated by the grammar. This stage of the grammar only accounts for that part of the painting shown in Fig. 3. But it appears possible to 'hang' further analyses on the superstructure assigned by this state of the grammar.

Testing the grammar

A grammar represents a theory, in this case, of the compositional structure of a class of paintings. There are a few ways to validate such a theory. The artist can be consulted to determine whether the grammatical analysis corresponds to his conscious plan of organization. There are perils in this approach, and of course, the uselessnes in the case of artists no longer living. Another form of validation consists of inspection by scholars with knowledge of the artist's oeuvre. They can compare their own intuitive analysis with that assigned by the grammar. Some day, it may even be possible for different grammars to be compared and evaluated, corrected, and combined, just as computer programs are today. But the most powerful form of test available to us at present is the resynthesis test.

The grammar is a description of structure. Separate algorithms must be written to perform analysis with respect to the grammar. But algorithms can also be written which will synthesize pictures ab initio from the grammar. These pictures need not correspond at all to any extant paintings. In fact, since the grammar accounts for an infinite number of paintings, it is highly unlikely that a picture generated at random from the grammar will correspond to one that actually exists. But all such pictures have the structure that the grammar allows. We can exploit this property to test the grammar. By generating a random picture (making random choices when the grammar provides alternatives) we can inspect the final pictures for similarilty with the artists oeuvre.

The picture of Fig. 4 is such a pseudo-Diebenkorn It was (randomly) chosen to be a suburban landscape just as is that of Fig. 3. It thus has both a 'busy' and 'open' region. When Richard Diebenkorn saw this randomly generated picture he had 'the immediate shock of recognition', notwithstanding the unusual history of its construction. The reader can, with the grammar (Kirsch 1985) in hand, perform such tests himself, comparing the product with his own intuitive ability to recogize Diebenkorn's Ocean Park paintings.

REFERENCES

1. Erle Loran, Cezanne's Compositions, Berkeley,CA, University of California Press, 1943.
2. Russell A. Kirsch, "Computer Interpretation of English Text and Picture Patterns",IEEE Trans. Elect. Computers, EC13 (Aug 1964), p. 363.
3. George Stiny, "Introduction to Shape and Shape Grammars", Envir. and Planning B, 7:(1980) p. 343.
4. Joan L. Kirsch et. al., "The Structure of Paintings: Formal Grammar and Design", Planning and Design, 12:4(1985), Forthcoming.

Fig 1. Richard Diebenkorn, Ocean Park No. 111, 1978, Oil and Charcoal on Canvas, 336.2 x 336.7 cm., Courtesy Hirshhorn Museum and Sculpture Garden, Smithsonian Institution

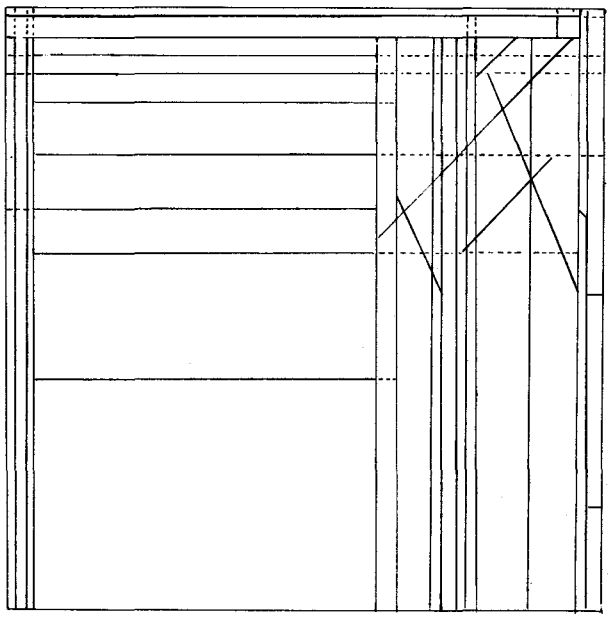
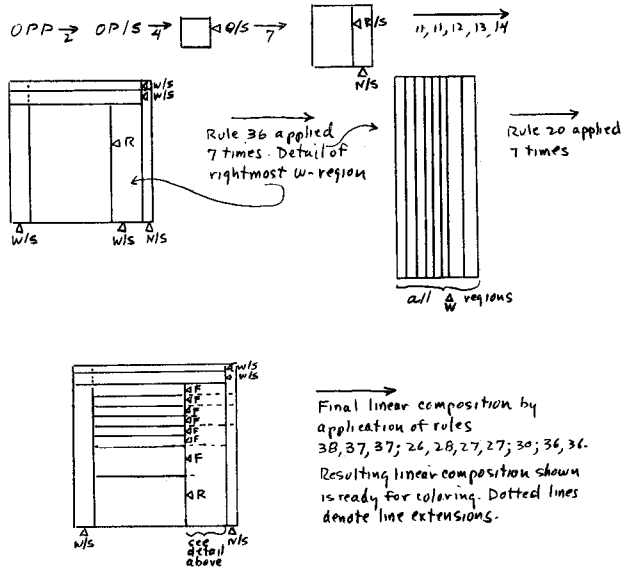


Fig. 3. Final linear composition assigned by the grammar.

Fig. 2. Linear composition analysis assigned by the grammar to Diebenkorn's Ocean Park No. 111.



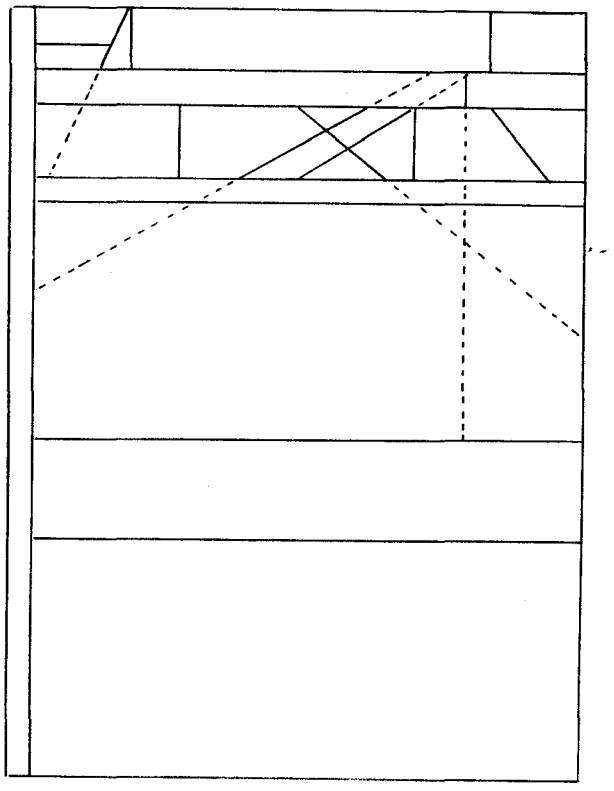


Fig. 4. A pseudo-Diebenkorn derived from the grammar by the following sequence of rule applications: 2,6,17,17,11,31,31, 31,30,38,37,30,31,30,30,30,32.

MUSIC AND DATA STRUCTURES: THE APPLICATION OF MUSIC THEORY IN PROGRAMMING COMPUTER ASSISTED INSTRUCTION

Gary S. Karpinski

Brooklyn College
Conservatory of Music

ABSTRACT

During the past twenty years, computer programs which help to teach music students have been conceived, written, and marketed. These CAI programs, like other music applications, must use some means of representing musical pitch in a computer. There is a relationship between musical structures and the corresponding data structures which represent them. This paper examines that relationship as it exists in CAI programs developed at the Lab for Computer Assisted Instruction in Music at the Brooklyn College Conservatory of Music.

INTRODUCTION

The typical data structure used to access musical pitch in a microcomputer places the pitches of the chromatic scale in an array numbered sequentially from the lowest to highest pitch. An array of this type may afford convenient access to all pitches, and even to chromatic passages through segments of the array, but it is not structured to allow the manipulation of even the most basic of musical figures such as chords and scales. Frequently, music programs treat each figure as a special case and encode long lists of notes or conditions in order to produce musically useful results.

Clearly, other data structures must be employed. They should be efficient and enable the programmer to control or randomly produce the musical figures which they represent.

After a brief discussion of some musical principles, I will examine one approach which offers the programmer such capabilities.

SOME BASICS

The chromatic array discussed above uses numbers which correspond directly to the height of each pitch. Such numbers will be referred to as pitch numbers.

A group of pitches is known as a collection and is represented by the pitch numbers of its members. A collection may be moved to a new pitch level. To do this, a constant is added to each pitch number in the collection. As an example, consider the first three notes from a major scale: 0,2,4. They may be moved to begin on 3 by adding the constant 3 to each pitch number: 3,5,7. This process is known as transposition and the constant is called a T-number.

Any musical pitch is considered to repeat itself at T-12. This principle is known as octave equivalency and is partly due to the fact that a pitch and its octave equivalent have the most fundamental frequency ratio of 1:2.

These terms and principles will serve to facilitate the following discussions.

PROBLEMS IN REPRESENTING CHORDS

In order to better understand the necessity for efficient and flexible data structures in programming music CAI, consider the example of a drill which presents students with the sounds of various chords and asks for details concerning their structure. For most instructors, the important details would be each chord's quality (loosely equivalent to pitch content) and inversion (loosely equivalent to pitch order).

Let's examine one specific chord quality: the major triad. Assuming a lowest pitch of 0, the pitch content of a major triad in root position would be by definition 0,4,7. Subsequent inversions of a chord are achieved by placing each lowest pitch in turn up to its next octave equivalent. In this manner, the first inversion of the major triad would be 4,7,12 (with 12 replacing 0) and the second inversion would be 7,12,16 (with 16 replacing 4). By transposing each inversion of the major triad down to begin on pitch 0, we have the following pitch collections:

Root position	0,4,7
First inversion	0,3,8
Second inversion	0,5,9

At this point, the structure of each inversion of the triad has been defined and may be used as a note list from which any transposition of that

CH2218-6/85/0000/0069$01.00 © 1985 IEEE

inversion is arrived at by adding a constant T-number to each element.

However, considering the number of chord qualities used in most CAI applications (usually 4-10) and the number of inversions of each chord (usually 3 or 4), defining a unique structure for each inversion of a chord would be an inefficient and cumbersome technique.

A SOLUTION

There is a method for deriving subsequent inversions of a chord from its root position. In defining a chord it is best to consider not the pitches themselves, but the distances between them. In this manner, a root position major triad may be defined by the distances 4,3 (4-0,7-4). Furthermore, since the process of inversion involves the repetition at T-12 of all or part of that series of distances, another distance which arrives at the next octave equivalent of the starting pitch must be included in the definition. This final number will, of course, be the complement of 12 and the total of the other distances. In the case of the major triad, the complement is 5 (12-(4+3)). Therefore, the complete definition of the major triad—known as its interval series—is 4,3,5.

By treating these distances as elements in a circular array or ring (figure 1), the different inversions of the triad are achieved by entering the ring at different points and adding the values of two successive elements to a starting pitch.

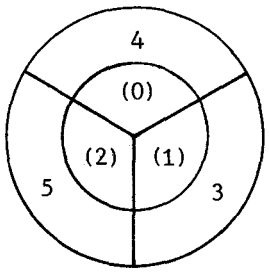


Figure 1. Major triad array

Entering the ring at element (0) yields a major triad in root position.

Starting pitch	0
Second pitch	4 (0+4)
Third pitch	7 (4+3)

Entering the ring at element (1) results in first inversion.

Starting pitch	0
Second pitch	3 (0+3)
Third pitch	8 (3+5)

Second inversion is reached through entering the ring at element (2).

Starting pitch	0
Second pitch	5 (0+5)
Third pitch	9 (5+4)

Thus, every chord may be defined by its interval series, and its inversions derived from that series in the above manner. In this way, musical inversion may be seen as a process of cyclical permutation. This approach has been used in the analysis of music by Chrisman[1], Regener[2], and most notably Perle[3].

Table 1 is a list of common chord qualities and their interval series which may be treated in the same circular fashion.

Table 1. Selected chord interval series

Chord quality	Interval series
Major	4,3,5
Minor	3,4,5
Diminished	3,3,6
Augmented	4,4,4
Major-major	4,3,4,1
Major-minor	4,3,3,2
Minor-major	3,4,4,1
Minor-minor	3,4,3,2
Half-diminished	3,3,4,2
Fully diminished	3,3,3,3

The four-note chords, also called seventh chords, require the addition of three successive elements to a starting pitch.

REPRESENTING SCALES

Applying this process to musical scales yields even more interesting results. The interval series for a major scale is 2,2,1,2,2,2,1. If these intervals are placed in a circular array, as seen in figure 2,

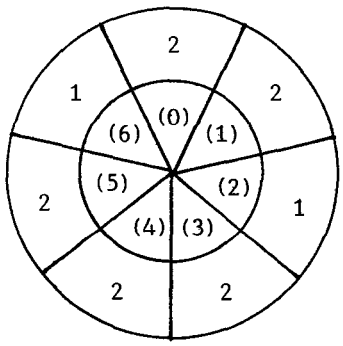


Figure 2. Scale array

entering that array at different points will produce various scales. The minor scale, for example, will result from starting with element (5) and proceeding around the array. Here are the various entry points and the scales which they produce.

Table 2. Entry points for scales

Starting element	Resulting scale
0	Major
1	Dorian
2	Phrygian
3	Lydian
4	Mixolydian
5	Minor
6	Locrian

This approach treats the scales as inversions of one another which, in a sense, they are.

A UNIFIED ARRAY

The above array, when considered without respect to a starting point, represents what is known as the diatonic collection. Since diatonic music contains not only the listed scales but also many chords, it is possible to use the same circular diatonic array to produce both scales and chords.

The process of building chords may be thought of as taking alternate pitches from a scale. For example, the first, third, and fifth notes from a major scale form a major triad. The structures of many chords may be derived from the diatonic array by combining the successive elements necessary to reach these alternate pitches. In the case of a triad, the elements must be combined into groups of 2,2,and 3; this will result in the interval series for a triad in root position which must then be manipulated as before in order to produce its inversions. The interval series for a seventh chord is derived from the array by combining successive elements into groups of 2,2,2,and 1.

Once again, take the example of a major triad. Its interval series may be arrived at from the circular diatonic array by adding elements (0)+(1), (2)+(3), and finally (4)+(5)+(6). The resulting distances will be 2+2, 1+2, and 2+2+1; these totals are 4,3,5—the interval series for a major triad.

Table 3 is a list of chords and where to access their interval series through the diatonic array. Remember that the numbers of elements to combine for triads are 2,2,and 3 whereas the numbers of elements to combine for seventh chords are 2,2,2,and 1. Also note that some chords may be produced by starting at more than one point.

Table 3. Chord entry points

Chord quality	Starting element
Triads	
Major	0 or 3 or 4
Minor	1 or 2 or 5
Diminished	6
Seventh chords	
Major-major	0 or 3
Major-minor	4
Minor-minor	1 or 2 or 5
Half-diminished	6

The relationships between the circular diatonic array and all of the scales and chords discussed thus far are indicated below. The values of the elements themselves are indicated on the inner circle of the ring with their index numbers in parentheses. The names of the triads are indicated at their starting points on the next circle. The seventh chords come in the following circle, and the scales produced by the array are printed on the outside of the ring.

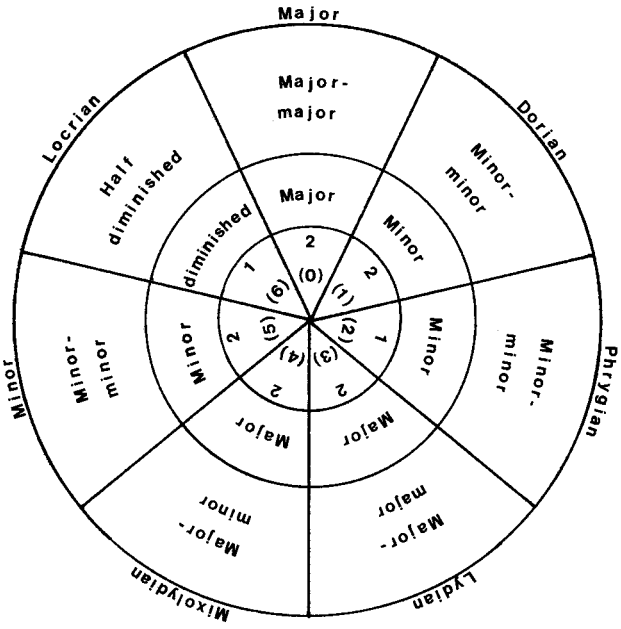


Figure 3. Circular diatonic array

The circular diatonic array contains the information necessary to produce seven scales, fourteen intervals of an octave or smaller, larger intervals of course, and the above chords plus many other chord structures. In addition, with the array it is possible to determine the scale degree of a pitch or chord within a given scale. Obviously, this array is extremely useful in programming music CAI and has implications beyond that area.

But what of other chords, scales, and the like? In particular, you may have noticed the absence of the augmented, minor-major, and fully diminished chords from the diatonic array. That is because these are non-diatonic chords and are not contained in the diatonic collection. However, all three may be derived from a circular array whose elements form the interval series of the so-called harmonic minor scale:

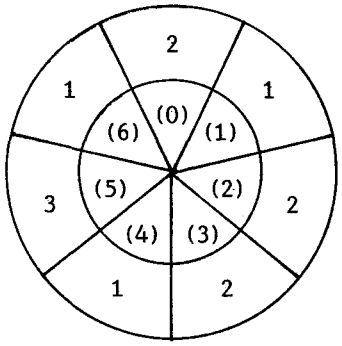


Figure 4. Harmonic minor array

The augmented triad would result from entering such an array at element (2) and adding elements in the necessary groups of 2,2,and 3. The minor-major and fully diminished seventh chords would begin at elements (0) and (6) respectively with the requisite 2,2,2,and 1 grouping of elements.

Other circular arrays may be used in this way in order to produce unusual collections which may occur in atonal, microtonal, jazz, popular, ethnic, and other genres of music. For example, what is known as the octatonic or diminished scale is found in jazz, Middle-eastern, and atonal music. Its interval series of 2,1,2,1,2,1,2,1 may be treated in the same fashion as the diatonic or harmonic minor collections above. Even though certain arrays may require different interval content, the important concepts of interval series, circular permutation, and interval grouping will still obtain.

CONCLUSION

The circular diatonic array proves to be an extremely powerful tool for use in programming music CAI. The principles under which it operates may be applied to other arrays and in other musical programming applications.

REFERENCES

[1] Richard A. Chrisman, A Theory of Axis-Tonality for Twentieth-Century Music, diss. Yale University, 1969.

[2] Eric Regener, "On Allen Forte's Theory of Chords," Perspectives of New Music, 13/1 (1974), pp. 195 ff.

[3] George Perle, Serial Composition and Atonality, 5th ed., Berkeley: University of California Press, 1981, pp. 3 and 148, passim.

The Yamaha DX-7 Synthesizer
A New Tool for Teachers

Dennis Righter and Rebecca Mercuri
Notable Software
P. O. Box 1556 - PS
Philadelphia, PA 19105
215/736-8355

INTRODUCTION

The Yamaha DX-7 is one of the most popular instruments in the history of keyboard synthesizers. This unique instrument's rapid rise to success is due, in large part, to the relative ease with which it can be played, along with the wide variety of realistic sounds that it can produce. These capabilities have made the DX-7 a natural choice for live performance as well as for studio work. Recently, music educators have discovered that this synthesizer can also function as a learning tool. It is important to note that the performance and synthesis aspects of the DX-7 can be used by teachers to present a wide variety of musical concepts. In this paper, we will discuss a number of techniques for using the DX-7 synthesizer in educational settings.

PERFORMANCE

In the past, the focus of synthesists has usually been on programming, rather than on performance. However, programming knowledge and abilities are not as imperative with the DX-7 as they were with the earlier synthesizers, due to the availability of a wide range of excellent presets provided by Yamaha, Notable Software and other software companies. With the older, "modular type" synthesizers, a thorough knowledge of programming was often necessary, even to just enable the instruments to produce sounds. Now, with the DX-7, novice synthesists can immediately have hundreds of sounds right at their fingertips, with no programming knowledge required, just by turning on the machine. This capability makes it possible to focus solely on the complex performance requirements of this multi-timbral instrument.

When musicians are faced with new pieces of equipment, they often tend to transfer their knowledge from previous experiences with similar equipment. It is common for keyboardists to apply pianistic or organistic chord voicings and "touch" when performing on the DX-7. This is acceptable when using synthesized piano or organ timbres, but it must be remembered that specific phrasings and orchestration are associated with woodwinds, brasses, strings and other familiar non-keyboard sounds. Students often program "outer space noises" or "bubbling volcanos" because there is no need to conform to any performance standards. Is someone going to criticize your rendition of "A Lawnmower Traveling Through the Center of the Earth?" Certainly not, unless they are jealous because it sounds better than their "Phase Dishwasher.

A complete knowledge of all the performance aspects of the various instruments that are being "re-created" (synthesized) should be included in any course of study dealing with electronic music. You only need to talk with the salesman at any organ store in order to understand that the way a sound is played is far more important than the actual components of the sound. The opening cadenza to "Rhapsody in Blue" has impressed countless organ buyers with the authenticity of the clarinet stop on many terrible organs. Likewise, the fanfare from the start of a horse race has been the single "trumpet stop" demo for years. The first day in "Organ Sales 101" class strives to impress the new organ salesman with the fact that a stop which only remotely resembles an oboe can be effective, as long as the opening strains of "Ravel's Bolero" are played. But there is more here than just sales hype. Many precious hours have been wasted by synthesists "messing with" parameters looking for "new" sounds, or trying to improve

CH2218-6/85/0000/0073$01.00 © 1985 IEEE

existing presets, when the solution to their problems could have been found in more appropriate performance techniques. Synthesizer students should be encouraged to listen to and analyze a wide variety of instrumental recordings so that they can learn to imitate different styles.

SYNTHESIS

The Yamaha DX-7 is the first programmable FM digital synthesizer made available to the public in an affordable price range. The FM (Frequency Modulation) technique used in the synthesizer is essentially the same as that used in radio transmission, the primary difference being that the synthesis frequencies are in the lower audio range. The simplicity of this technique means that one does not have to be a mathematician in order to create interesting and complex sounds. The Yamaha company has gone one step further in making this powerful technique accessable to performers, by designing the DX-7 instrument in a logical and easy to understand format.

John Chowning was the first to apply FM to music synthesis (see References section below). He soon discovered that simple carrier-to-modulator ratios could produce familiar sounds. A number of his early experiments can be duplicated using two signal generators (available in the Physics department of most schools) and a loudspeaker. The set-up is simple: feed the output of one of the signal generators (modulator) into the other one (carrier) and attach the resultant output to a loudspeaker. You can then monitor the effects produced by changing the frequencies of the two signal generators and listening to the output. Although the DX-7 uses digital synthesis, either analog or digital signal generators can be used for experimentation. The only requirement is that an analog signal be fed to the loudspeaker. An analysis of digital signal production would be instructive for more advanced students.

With this equipment you can produce a trumpet-like sound when both carrier and modulator are set at the same frequency (400Hz works well). A bassoon sound is created when the carrier/modulator ratio is 5/1 (try 500Hz to 100Hz). Percussive sounds are heard when a non-integer ratio is used (a nice bell is produced with the carrier/modulator settings at 200Hz and 280Hz). For added realism, place an envelope generator (also available in Physics laboratories) just before the loudspeaker connection and experiment with ADSR (attack, decay, sustain, release) functions.

Once these basics are fully understood, it is important to trace through each of the "algorithms" provided on the DX-7. This is done by shutting off all of the operators and then re-introducing them, one at a time, (experimenting with different orders) so that the interaction between the carriers and the modulators may be revealed. The envelope generator portion of the synthesizer should also be explored in a similar systematic way. In this manner, a thorough knowledge of numerous synthesis fundamentals will have been developed through the use of the DX-7.

SPECIAL EFFECTS

It is important for educators to focus on the "FUNCTION" section of the DX-7, as it is one of the most powerful areas of the instrument. The use of PITCH BEND, PORTAMENTO, AFTER TOUCH, dynamics and articulation should be stressed. [Specific functions provided on the DX-7 have been capitalized here for clarity.] The DX-7 provides a significant amount of versatility in the application of these special effects. For example, vibrato can be added through the use of the MODULATION wheel, the AFTER TOUCH, the MODULATION pedal, or the BREATH CONTROLLER. The vibrato effect can even be programmed directly into the sound, adding it automatically. The same can be done with AMPLITUDE MODULATION (tremolo) or ENVELOPE GENERATOR BIAS (attack, sustain and release). Experimentation with these functions, and a thorough understanding of their characteristics from a theoretical standpoint, is essential in order to utilize the full potential of the DX-7.

For a sound to be "authentic," it is necessary to analyze the characteristics of the instrument it is supposed to represent. For example, vibrato, PITCH BEND and PORTAMENTO must all be used to obtain a realistic trumpet sound. An in-depth examination is necessary for the "re-creation" of any natural sound on the DX-7, from a piccolo to a string bass. This analysis is similar to the manner in which the performance aspects of actual instruments were examined (see above). The bibliography at the end of this paper lists some of the many studies of

orchestral instrument sounds - data collected in these studies can provide a starting point for interesting DX-7 experiments.

There are dozens of possible ways in which the FUNCTION section of the DX-7 can be configured. A few practical arrangements are listed below. If you have access to a DX-7, you can enter and experiment with the following instructions, a step at a time, as you read them. If the instrument is not available, you can make these entries on a blank DX-7 Voice Data Sheet (reproduced from the inside back cover of your operation manual) while you read each sentence.

An easy to control, natural vibrato can be created using the AFTER TOUCH (FUNCTION Switch #30) set to PITCH=ON, with a relatively low range of 25 to 30 (FUNCTION switch #29) and the Pitch MODULATION Sensitivity set at 1 or 2 (EDIT Switch#15).

If for some reason you need more vibrato, set the MODULATION Wheel to PITCH=ON (FUNCTION Switch #18), and set the MODULATION Wheel RANGE to 65-70 (FUNCTION Switch #17). Use your left hand to add that extra vibrato with the MODULATION Wheel. If you buy another pedal, you can also set up the same effect using the MODULATION Pedal Functions (FUNCTION Switches #22 and #21). On sounds that use the PITCH WHEEL frequently, you have the option of setting up the BREATH CONTROLLER (FUNCTION Switches #26 and #25) to add vibrato, since your left hand might be busy bending notes. This technique is useful for wind instruments such as trumpet, trombone, and harmonica, which use frequent pitch bends.

Many players keep their PITCH BEND RANGE set very high, but it is possible to imitate a natural PITCH BEND with more precision at low settings. Set your PITCH BEND RANGE to 1 or 2 (FUNCTION Switch #3), and set the PITCH BEND STEP to 0 (FUNCTION Switch #4).

A necessity for controlling the PORTAMENTO is the optional PORTAMENTO Pedal (you can use your sustain pedal to try it out, but it's more practical to own both). With this pedal plugged into the PORTAMENTO Pedal Jack, the PORTAMENTO will only engage when the pedal is depressed. Set the PORTAMENTO MODE to RETAIN (FUNCTION Switch #5), and the TIME to 25-35 (FUNCTION Switch #7), depending on the kind of instrument you are playing. To work on the fine art of using PORTAMENTO, experiment with the Fingered Porta as described in your DX-7 manual on pages 53-54 [Special effects from {Data Entry} while in PLAY mode]. This important section of your manual is often overlooked and is very useful.

The settings presented here have been used successfully in education and performance. As in learning to play any new instrument, some of these techniques may be difficult to implement at first. Students should therefore be encouraged and challenged to use the FUNCTION section to enhance the realism of the sounds on the DX-7.

CONCLUSION

We have found the Yamaha DX-7 synthesizer to be suitable in a wide range of educational settings. Its versatility as a real-time performance instrument and programmable digital synthesizer, combined with its rugged construction and ease-of-use, have made it a perfect selection for scholastic applications. As you have seen, the DX-7 can be used in teaching both performance and synthesis techniques - its advantage over earlier synthesizers lies in the fact that these concepts can be taught separately.

In this paper, we have presented numerous educational applications for this synthesizer. In addition to the features we have listed, each DX-7 contains an interconnection system (MIDI) which enables it to communicate with other synthesizers as well as with computers. As units are added to the system, the educational possibilities increase dynamically. We, therefore, present the challenge to the reader to explore the tremendous potential of the Yamaha DX-7 as an educational tool.

REFERENCES

Yamaha Digital Programmable Algorithm Synthesizer Operation Manual, Buena Park, Yamaha International Corp., 1984.

Beauchamp, James W., "A Computer System for Time-Variant Harmonic Analysis and Synthesis of Musical Tones," in Foerster and Beauchamp, ed., Music by Computers, New York, Wiley, 1969, pp. 19-62.

Chowning, John M., The Synthesis of Complex Audio Spectra by Means of Frequency Modulation, Journal of the Audio Engineering Society, September 1973, Vol. 21, No. 7.

Howe, Hubert S., Electronic Music Synthesis, New York, W. W. Norton & Co., Inc., 1975. ISBN 0-393-09257-7.

Moorer, James A. and Grey, John, Lexicon of Analyzed Tones (Part I: A Violin Tone, Part II: Clarinet and Oboe Tones, Part III: The Trumpet), Computer Music Journal, Vol. 1, No. 2 (April 1977), Vol. 1, No. 3 (June 1977), Vol. 2, No. 2 (September 1978).

Righter, Dennis, RIGHTERSOUND™ I, Philadelphia, Notable Software, 1985. ISBN 0-925870-05-6.

Olson, Harry F., Music, Physics and Engineering, New York, Dover, 1967.

APPLICATIONS FOR COMPUTERS IN THE ARTS

DAVID A. BUTLER

4346 Hunters Club Drive, Raleigh, NC 27606

ABSTRACT

Computers have recently been showing up in some unusual places, often going beyond their typical status as a tool. The creators of visual art, music and dance are discovering a powerful new partner that can not only assist in the creative process but often facilitate totally new artforms based on the computer as the medium or as an actual participant in the art experience. This paper surveys current literature and research on computer applications in the arts and provides a broad resource base for those interested in studying this subject in greater depth.

1. INTRODUCTION

One of the most exciting new applications for computers and microprocessors is in the arts. Computer scientists and artists are just beginning to recognize the potential for such applications. Even though pockets of research have existed for nearly two decades, it is only within the last five years that any significant work has been done to expose the accomplishments of those who have pioneered this field. Most of the literature has dealt with specific applications, usually related to either music or visual arts. The objective of this paper is to survey current research over a broad base of applications and to highlight important resources for those interested in further study.

2. A BRIEF OVERVIEW

Probably the main reason why interest in computer applications in the arts has been on the rise is the proliferation of microcomputer systems. Students who are not enrolled in traditional computer-oriented curricula (e.g., computer science, math, engineering) are exposed to computers in increasing numbers, and the most common vehicle for the introduction of computing to youngsters is art (graphics and music).

2.1 A Historical Comment

As early as the mid-sixties, work was being done on computer graphics and electronic music synthesizers. Bell Labs is often credited for the development of the first computer music system, using a large and expensive mainframe computer. In 1966 Susan Sontag, in her book, *Against Interpretation*, described a new breed of artists who were using advanced technology to realize very unconventional artforms. She referred to such artists as the pioneers of a "one culture" that might bridge the gap between humanists and scientists [1].

An excellent history of computer applications in the visual arts can be found in H. W. Franke's book, *Computer Art--Computer Graphics*. Another account of the progress of computer art (visual) is found in an article written in 1978 by Grace C. Hertlein [2].

2.2 Artists vs. Scientists

Before exploring the many applications where computers are being utilized in accomplishing art, it is important to understand some of the implications of such a marriage. In general, there has always been a strong polarization of artists and scientists. Those rare individuals who have developed considerable skills in both art and computer science have found great satisfaction in combining these two disciplines. If one considers other fields where interdisciplinary knowledge has been requisite, there is usually a significant overlap in the skills needed to work in either area. For example, the emerging field of robotics combines electrical and computer engineering with mechanical engineering and materials engineering. Note that all of the contributing areas of knowledge have engineering (and thus math) as a common denominator.

As we learn better ways of making systems more user-friendly, the practitioner need not be an expert, or even particularly knowledgeable in the operational theory of the computer which aids him or her. Only the capabilities and limitations of the system must be understood, along with a working knowledge of the operational language, in order for an artist to

CH2218-6/85/0000/0077$01.00 © 1985 IEEE

effectively utilize a computer-related tool.

Certainly, the creation of such tools must take place in an environment that encompasses both those who can design computer-based devices or software systems and those who can provide the necessary insight into the creative requirements from the artist's point of view. Even in a development environment, it is unnecessary for the participants to have both an artistic and engineering background; however, the artist should have a general knowledge of the hardware and/or software being developed, and the technical designers should understand the creative process that is being automated or facilitated.

Certain applications that will be investigated involve the computer actively in the art itself (e.g., interactive art). Here, the creation of the artform requires a working knowledge of the computer; thus this artist often possesses more than a casual knowledge of programming and interface techniques. These applications are unique and experimental in nature and have become more sophisticated as artificial intelligence has come of age.

2.3 Is It Really Art?

One complaint often heard from artists using computer-based tools is that the computer cannot do this or that. It is this frustration that illuminates the single most important problem that today's "computer artist" must overcome. The computer, like every other medium for accomplishing art, has limitations. However, the artist must look beyond these limitations before he or she can recognize and exploit the vast capabilities that the computer possesses. In this author's opinion, the process of using the computer to imitate traditional artforms is only useful in helping the artist to understand and become skillful in using the new tool. However, before the computer is to become accepted as an important medium or tool for art, the artist must use his or her creativity to accomplish entirely new artforms that transcend the traditional while maintaining aesthetic appeal.

A good analogy to this premise is the artistic evolution of photography and cinematography. In the early stages of their development, both tended to imitate accepted artforms before their users finally exploited the vast potential offered by these new technologies to produce revolutionary new artforms [3]. The same analogy is drawn in [4] in reference to computer-aided lighting control boards.

The following quotations from an article by Ken Sofer help to put computers and art in proper perspective: "It is obvious that computers will continue to alter our culture, and in particular our notions of how art can be made and viewed. What is not so obvious is the form the art will take." Sofer goes on to say, "The most successful computer artists are those few who do more than merely replace the paintbrush with the electronic pen. These artists use the computer as a means of significantly advancing artistic concerns--a process that often requires conceptual and aesthetic breaks with practices that may have taken a lifetime to develop" [5].

3. A MULTITUDE OF APPLICATIONS

This section presents an exhaustive survey of the many different applications that are being developed or are already in use. Six categories have been used to organize the various uses for the computer, although it should become evident that there is much overlap between them. These categories are: (1) Music, (2) Visual Arts, (3) Dance, (4) Theatrical Design and Production, (5) Media Technology, and (6) Interactive Art. An additional category has been included (Other) so that several more applications might be discussed that do not fit into the former categories.

3.1 Music

The obvious use for computers in music is the analysis and synthesis of sound through the use of digital synthesizers and signal processing equipment. However, there are several other applications that warrant investigation.

A program has been written for home computers that facilitates the process of learning to sight read [6]. It is designed for the home or an educational environment and is basically a "drill and practice" program.

Another use is the automated transcription of recorded music. A special system has been devised that can analyze an audio tape and detect missing notes as well as print the sheet music using a special music font [7]. The system can be interfaced with a synthesizer for playback to ensure integrity. This same concept has been extended to electronic music; however, a universal notation for electronic and computer-generated music has not been developed. A system has been developed at University of Milan, Italy that can transcribe three parameters of electronic music: frequency, amplitude and duration [8]. This system, called *EMPS*, is useful to convey compositional techniques and to facilitate comprehension in an educational

context. This method of notation can be sufficient to communicate a complete score to a signal processing system, but is inadequate to convey a score to a performer.

Computers have been used with significant success as an aid to the composer; computer-based editing systems are numerous. These systems can be used for composing for conventional as well as computer-based instruments within the limitations of conventional notation. The *INTERSCORE* system is described in [9], and is specifically designed for computer music composition. AlphaSyntauri has marketed several packages for the Apple computer that include keyboard, software and an interface card. A composer can develop a score with this type of system in much the same way that a writer uses a word processor.

A lesser known application for computers in music is automated composition. Here, the artist must be able to write programs that utilize the computer's ability to generate random numbers and incorporate this into a structure that will produce musical passages. An example of such a system is found in [10]. Here, a statistical analysis of the incidence of particular chords in bluegrass music enabled the programmer to develop software that would play a virtually infinite number of variations on a theme. What is actually occurring is "randomness within structure," much the same way that Mozart composed his *Musikaliche Wuerfelspiel.* Don Slepian and Jim McConkey used this same concept in their performance of *Mozart's Musical Dice Game* at the 1984 Symposium on Small Computers in the Arts Concert.

By far, the most widespread use of computers in music has been the production of electronic music. Less than ten years ago, the use of computers to produce music was restricted to research laboratories housing large mainframe computers. Analog synthesizers dominated the electronic music scene, soon to be integrated with computers to facilitate their control. As microprocessors have become inexpensive, digital synthesizers have emerged that can *create* the waveforms and then convert the digital representation into an analog signal that can be played through standard audio equipment. Home computers, however, still use analog audio chips to produce the wave, and the computer is used as a control device.

Electronic music has gone a long way toward gaining popular acceptance. Today, it is the exception when one does not find the synthesizer as a participant in the music scored for movies, television, popular albums and commercials. Many contemporary composers have included the synthesizer in a prominent role in their scores. It is here that artists have moved beyond imitation of the traditional to capture the enormous potential for creative expression that electronic music offers. Marjorie Lyon puts this in perspective in her article, *A Third Medium for the Music Composer: Computers.* She observes that voice was the first medium and instruments provided the second medium. Computer-generated music should not replace these media, but should provide the composer with a "more enriched palette" [11].

One major limitation of digital synthesis is the enormous amount of computing power necessary for superb results. However as increasingly powerful computers are becoming less expensive, the use of computer-based synthesizers for realtime performance is not far off, which will also facilitate interactive composition. This is not to say that these things cannot be done today with inexpensive equipment. However, there is a trade-off with the complexity of the waveform. Even with small systems, complex sounds can be generated given adequate time for computation; but the ability to perform in realtime is sacrificed. Needless to say, composers have not been preoccupied with this limitation; mass produced digital synthesizers have proven to be more than adequate from a creative standpoint.

On the research level, The Center for Computer Research in Music and Acoustics (CCRMA) at Stanford, The Computer Audio Research Laboratory (CARL) at UCSD, and the Experimental Music Studio at MIT are some of the most notable laboratories in computer music today. These facilities employ mainframe computers and specialized signal processing equipment to accomplish realtime digital synthesis at a very high level. It is the research that is taking place at these facilities that will place increasingly powerful systems in the hands of composer/musicians.

Although computer music compositions are typically performed in the recording studio, live performances are becoming more practical. The annual Philadelphia Computer Music Concert is in its seventh year (now held in conjunction with the Symposium for Small Computers in the Arts). The International Computer Music Conference also puts on concerts in conjunction with its meetings. Reviews for these concerts can be found in the *Computer Music Journal.* Last year, the International Computer Arts Society co-

sponsored DIGICON 83 in Vancouver. Jean Piche set up an elaborate network via satellite so that musicians in Sydney, Australia and Tokyo, Japan could perform simultaneously with Jean in Vancouver. The piece, *Night Satellite Teleconcert*, was performed on Fairlight CMI synthesizers. Electronic delays were used to offset the time required to transmit the signals to each location. The applause of the audiences in each location was fed through audio and video channels, creating an unusual synergy [12].

Composer David Rosenboom put on a performance at New York University in Toronto. His piece was unusual because it was interactive with a person's brainwaves. He had done research on the neurological patterns associated with music perception. He then developed a program which would trigger events in a live performance based on realtime measurements of brainwaves. Patricia Smith reviewed the performance in [13] and called it "the most elaborate live performance."

A popular combination today is the use of visual art in conjunction with live or recorded computer music. Typically, the visual medium is a video image that can be interactive with the music or lasers and lights. Several "visual music" performances have been reviewed in recent literature. At DIGICON 83, Roger Powell and Richard Vanceunebrouck-Werth teamed up to use eight synthesizers along with lasers, smoke and lights to put on a well-received performance [12]. In Kenneth Square, Pennsylvania, Colvin Randall has put together an enormous water, music and light show called *Conservatory Fountains*. He uses microprocessors to control the water and lights, enabling a virtually infinite number of sequences to be possible. The show can be seen during the summer months [14]. Finally, one of the most elaborate multimedia performances occurred at the 1981 International Computer Music Conference. John Cage and Lejaren Hiller used seven harpsichords, fifty-one computer-generated sound tapes, over five thousand slides and several films in the performance of *HPSCHD* [15].

3.2 Visual Arts

Like electronic music, computer applications in the visual arts have gained widespread acceptance. In fact, some commercial production companies have opted to imitate the popular computer-look when they were too small to own the sophisticated equipment [3].

It is appropriate to consider computer graphics and computer animation separately. For the lack of a better term, computer art will be used to describe computer graphics that have been produced for art's sake, and computer animation will be used to refer to the production of animated sequences using computers to generate each image. In the general sense, computer graphics refers to all visual representations on a computer, whether the end use is for business, games or art. Computer art, in a general sense, can refer to any artform created with the aid of a computer, whether visual or otherwise.

Many software packages are available for small computers that provide the artist with a number of pre-defined functions. In creating computer art, each pixel (or dot) on the monitor screen is controlled by the program to create the overall image. These packages enable the artist to draw standard shapes, manipulate them and define their colors so that he or she is removed from the mathematics involved in producing the image. Often a digitalizing tablet or a "mouse" is used to draw the shapes and line segments onto the screen. One such system, *EASEL*, is described in [16]. It costs about $600 and runs on most microcomputers. Reviews for the various programs, often referred to as "paint programs," can be found in popular magazines devoted to a particular machine.

Paint programs, just like all other utility programs, often have unique capabilities and limitations. One artist describes an effective approach to deal with this problem. In [17], Ame Choate Flynn discusses the use of several packages to achieve the desired effect. The image is saved after each session and then re-opened under a different program to add the elements done best by that particular package.

An artist's workstation has been the topic of several recent articles. This refers to the totality of the equipment and software present in a complete system. Typically, the components include a computer, a high resolution graphics display, some form of peripheral memory device, a digitizing tablet, a mouse or joystick, and possibly a digitizing camera. This camera will generate a signal that can be stored in computer memory so that it can later be manipulated, merged or otherwise modified. A general discussion of such a workstation and the benefits of various components can be found in [18]. An in-depth discussion of the use of a workstation is included in [19].

One highly visible use for computer art has been in the Olympics. In 1980,

artist Leroy Nieman used a paint system developed at the New York Institute of Technology to capture the vivid visual imagery of the Winter Olympics. This year, artist Joni Carter used even more elaborate equipment to realize many scenes from the Summer Olympics in Los Angeles [20].

Galleries are displaying computer art with increasing frequency. Most major galleries have had a display at some time or another. Recently, the art gallery at the North Carolina State Student Center displayed a number of works from the 1983 SIGGRAPH conference. (SIGGRAPH will be discussed in section 4.) With the advent of Videotex, there has been interest in making computer art available to subscribers. One limitation is the poor resolution of most displays owned by the general public. An interesting variation on this, described by Martin Nisenholtz, is the *Electronic Gallery* [21], where a flat screen monitor is placed in public view (an office lobby, gallery, etc.), and new images are continually fed to the monitor. One such installation is in the Alternative Media Center at New York University's School of the Arts.

David Em, a famous computer artist, has found a number of avenues for displaying his work. The Silver Anniversary Issue of *EDN*, devoted to electronic technology, had Em to produce most of the graphics and art used in the issue. He has also shown his work in fine-art galleries, photo galleries, multimedia productions and both graphics and computer-technology conventions [22].

Some computer artists have experimented with computer-generated art. Here, the artist writes a program that will use random numbers in much the same way as described earlier with music. The artist introduces the art idea in the form of his program. The resulting creations would be related in the sense that they were the result of the same program [23]. Harold Cohen has developed a program called *AARON* that manipulates a "turtle" over a large sheet of paper laid on the floor. The creative process is automatic in that Cohen has no interaction with the system during the drawing process. According to Ken Sofer, the turtle draws "asymmetrical shapes and calligraphic scribbles with remarkably expressionistic lines" [5].

Let us move on to the world of animation, where practical applications can only be realized using equipment well out of the price range of most artists and even many universities. It will be shown, however, that highly creative animation is often accomplished with very inexpensive equipment, where aesthetics rather than commercial viability is the objective.

Computers were first introduced to traditional animation as a tool for automating many of the tedious, labor-intensive tasks that must be accomplished to obtain the final product. It has been said that Walt Disney could not afford to produce a *Fantasia* or a *Snow White* today due to the high cost of labor. The computer has been useful in automating in-between frames (interpolating the movement occurring between two hand-drawn frames) and doing the painting [3]. At Hanna-Barbera Productions, the computer has been utilized to such an extent that the production of a show like *The Flintstones* has become "largely an information management problem" [24].

Computer animation has also been useful for creating special effects in movies such as *TRON* and the *Star Trek* movies. In *TRON*, live actors were combined with animated backgrounds. Live action footage can be digitized and then manipulated via computer to accomplish effects with live actors that otherwise could not be achieved.

The process of using the computer to generate a complete animated sequence is becoming more and more feasible. There are a number of companies and research laboratories working in this area. (Several are discussed in section 4.) The most visible evidence of this is the proliferation of computer-generated logos and commercials on television. Robert Abel was a pioneer in this field with his famous series of commercials for Levi Strauss and Company.

The computation requirements for realistic scenes is mind boggling. To appreciate the amount of data that must be processed, consider that the typical resolution that is used for computer animation is 4000 x 6000 (or 24 million pixels)! Digital Productions in New York uses one of the world's most powerful computers to generate images that take up to seven minutes of computer time for each frame! (Consider that this machine, the Cray X-MP, has the capacity to process up to 630 million instructions per second [25]!) An in-depth description of the process used by Digital Productions to generate computer animation appears in [26], and a similar description of the process used by Information International, Incorporated can be found in [27].

At the 1984 Symposium on Small Computers in the Arts, many videos were shown that demonstrated the realism that can be achieved using such equipment. Many of

these animations were also shown at the SIGGRAPH conference earlier this year and were generally quite impressive. However, one cannot escape the feeling that the persons involved in these animations were caught up in the technical aspects of the process, while letting the creative aspect take a back seat. One of the most refreshing surprises of the SSCA was the presentation given by the students from Virginia Commonwealth University. Using equipment which could well be within the budgets of most high schools and many artists, numerous examples of the work being done at the School of the Arts at VCU were shown to an enthusiastic audience. In this author's opinion, far more creativity was demonstrated in these modest portrayals of computer art and computer animation than was evident in much of the work coming from the state-of-the-art laboratories. This is encouraging to the artist who will never have access to such equipment, for it is the level of creativity that separates art from high-tech computer graphics.

3.3 Dance

One of the newest applications of computers in the arts is the implementation of dance notation on microcomputers. "Dance notation has a low literacy rate among choreographers," according to Linda Hirschmann [28]. She quotes Dr. Stephen Smoliar as stating that none of the major dance companies take dance notation seriously. When implemented manually, notations such as Labanotation are quite cumbersome to use; however, the computer can be used to solve this problem.

The benefits of using a formal method of notation include the communication of desired movements to dancers, providing a facility through which the choreographer can play the "what-if" game while working on a piece, providing a means to achieve works that have been completed and, when implemented on the computer, the ability to analyze and critique the dancer's motions [56][57].

The system being developed at the time of Hirschmann's article (1979) used stick figures which could be given instructions via Labanotation symbols and modifying signs. Macros were being developed to accomplish the most common sequences. Shadows were used to make floor contact more obvious, and the model could also detect collisions between body parts. A new system was being developed that would allow video tapes to be shot of a dance from several angles simultaneously and subsequently be converted by the computer into Labanotation. Although this author has not found any further information on the progress of this project, it looked promising.

There was evidence of a similar system at the 1983 Symposium on Small Computers in the Arts. Here, a video tape was shown of *Plaisir Synthetique*, a dance and video synthesis production choreographed by Jean Marc Matus, who used a computer to aid in the design of the dance steps [55]. Although a computer was not used in the choreography, computer-generated stick figures were projected on a backdrop while seven live dancers coordinated their movements to three musicians and recorded electronic music.

At the 1984 Symposium on Small Computers in the Arts, the concert was dominated for the first time by dance-related performances. One performance was accomplished entirely on an Atari 800 computer. The dancers were stick figures projected onto large video screens against an animated stage and curtain. The music (composed and programmed by Marc Ruben) was also produced on the Atari. The dances were choreographed by Cathy Stadler and Manfred Fishbeck. Another performance utilized a digitizing camera to record a dance by Melanie Stewart in a taped piece called *Video Sketchbook*. The digital image of the dancer was modified using video synthesis techniques by Allan Powell and Connie Coleman. Still another dance piece was performed. This piece, entitled *Sign in Space*, was also choreographed by Melanie Stewart; here, five dancers performed live to music composed and recorded by John Hodian, while the audience watched a video image of "digitized dancers" over to the side. The synergy of the two events provided an exciting finale to the concert.

3.4 Theatrical Design and Production

Another emerging application for computers and microprocessors is in the theater. Whether producing a play, a dance or a musical performance, lighting must be controlled and sets must be designed. Microprocessor-based lighting control boards are commonplace in large performing arts centers, and CAD systems for set design are beginning to be used. The American Theater Association has already included sessions on computers in the theater in its annual convention. However, a less exciting application may yield even greater benefits: a data management system. Rick Graves has developed such a system [29].

Graves has formed a company together with the Guthrie Theatre in Minneapolis to produce and market a software package that will handle ticketing, sales, fund-raising, mailings and membership data bases. Other potential applications for such a system come to mind: accounting, inventory management and project manage-

ment. All of these packages are available for other industries and could be modified to match the needs of a performing arts center.

The periodical, *Theater Design and Technology*, has had several articles since 1982 related to computer-aided design tools for scenic design. The earliest article proposed a standardized graphics language that could be implemented on a computer [30]. It was interesting to note that the U. S. Institute for Theater Technology has a Graphics Standards Board. (*Theater Design and Technology* is a publication of the USITT, and it is this board that wrote the referenced article.) The next year, an article was printed in a new column called *Computer Exchange*. A system in use by the University of California, Davis is described. It is implemented on a PDP-11/55 computer and includes a high resolution monitor, a digitizing tablet and custom software. The system helps the beginning student in scenic design deal with the process of creating three-dimensional art, the most difficult and frustrating part of being introduced to scenic design [31].

In 1984, an article appeared that described another system in use that works on Apple computers. *ROBO GRAPHICS* is a marketed software package and utilizes a joystick or digitizing tablet along with a plotter (printing device). Once the image is created, it can be plotted, enlarged, reduced, rotated (in two dimensions), duplicated or mirrored. Module images can be stored and integrated into larger designs as needed. The principal applications are for floor plans and sections and for show plans [32]. There are a multitude of CAD packages available for other applications, and these could easily be converted to a theatrical application given the interest and time of a design and production person associated with a facility.

Lighting control systems have been ideal for integration with microprocessors due to the sequential nature of lighting cues. The lighting designer has far more room for creativity when he is not limited to what a human can accomplish with two hands. All of the cues can be programmed ahead of time and triggered automatically or on manual command or both. Dramatic effects can be accomplished by sequencing dozens of settings within a short interval of time. According to William Bellman, software consoles "can perform almost any task the scenographer can devise and may well inspire him to new creative heights. They are, in effect, artistic tools awaiting the genius of creativity" [4].

There is a wide range of lighting consoles available on the market in all different price ranges. The most sophisticated systems are well over $50,000. One other major advantage of such systems is the ability to remove the controls from the stage area to a location from which the stage can be viewed.

Another possible application for computers in lighting is one that this author has yet to see: the use of a simulation program to aid the lighting designer to see the effects of particular sequences and combinations of lights ahead of time, on a high resolution video screen. This would enable him or her to play the "what-if" game without taking valuable production time when cast and crew are present.

The use of lasers and holography in the theater is also discussed by Bellman in [4]. The possibilities for use of such high-tech effects are just beginning to be explored. Numerous laboratories and institutions involved in laser and holographic research are cited in [33]. A special on NOVA entitled *Artists in the Lab*, originally broadcast on November 15, 1981, also discusses the use of lasers and holography in art. These resources should prove valuable for those interested in applying these technologies to theatrical productions.

One final application that was discovered for computers in the theater involved the computer as a participant in a dance being staged. The computer program was, in effect, the score, which interacted through the use of sensors in realtime with the dancers as they moved about the stage. Musical events as well as the automatic movement of panels across the stage were triggered by dancers' movements, gestures and voices. Optical, video and audio sensors were interfaced with two Apple computers, stepping motors for panel movement and an audio playback system. The first major experiment was at the Festival of Two Worlds, Spoleto '81. The piece was entitled *Liberta a Brema* [34].

3.5 Media Technology

Media technology is used to refer to the methods by which art, whether it be music or visual art, is transferred to a medium such as audio tape or film so that it can be enjoyed by large numbers of people. Three broad application areas will be discussed: computers in audio production, filmmaking and commercial art.

Microprocessors have already gained widespread use in the control of audio mixing, in much the same way that they are used in lighting control boards.

Many recording artists depend heavily upon the creative talents of the mixing engineer to realize their final product. Modern recording studios typically have twenty-four track recording machines, creating an enormous control problem if dynamic fading, panning and equalization are required. One such system employing microprocessors is described in [35].

Audio cueing and control in filmmaking has also received considerable attention. Here, the additional problem of synchronizing up to six audio tracks with the film frames adds another dimension to the mixing problem. The computer offers an ideal solution, providing a facility for autocue based on frame numbers [36]. A notable example of a system that has been developed specifically for audio production in the film industry is the ASP system developed by James Moorer at Lucasfilm, Industrial Light and Magic Division. Included is an elaborate digital synthesizer; a utility for score editing and orchestration, as well as composition; a thirty-two track digital recording deck and mixing console; and an extensive array of proprietary signal processing devices, all controlled by an MC68000 microprocessor operating under UNIX. The system is capable of processing instructions at the rate of 18 million per second, a speed considered adequate for superior sound quality in a digital system [4][37].

Another problem which has been solved by computers is the labor-intensive process of film editing. Until recently, most studios had to handle every frame of film during the editing process, introducing many opportunities for damage. The film had to be physically cut and spliced to produce the final master. Just as in audio production, a great creative talent is required to do a good job of editing. Now, systems have been developed that largely automate the handling of the film. One such system is described in [38]. Again, Lucasfilm has developed a proprietary system, which it has marketed, called *EDITDROID*. Ralph Guggenheim is in charge of this project. Here, film is transferred to a number of video discs where it can be viewed in any frame sequence specified by the editor. Frames can be resequenced, deleted or added simply by keying in the frame numbers. The final sequence is printed out and is then used by the film laboratory to create the original master [39].

Another use of computers in the film industry is for the control of camera position. In specialized shots, extremely accurate control must be maintained. The *ACES* system has been used for this purpose and can be positioned dynamically with .01 inch accuracy. It can rotate 36 degrees per second up to 720 degrees and can move along a 46-foot track at speeds up to 3.5 feet per second [40].

The use of computers in commercial art encompasses most of the applications discussed thus far in this paper. Video synthesis developed as a result of the applications to be found in television commercial productions. Computer animation is the hot area in commercial production today and has been presented in section 3.2. Many commercial artists working in the print media are using graphic workstations as described in the same section.

3.6 Interactive Art

Some exciting and unique experiments in interactive art have been made possible by virtue of the fact that microprocessors and microcomputers have become inexpensive enough to be owned by interested artists. Interactive art refers to the ability for the viewers to interact with the art in such a way as to become co-creators in realtime.

One such experiment was designed by Michael Hayden and involves a computer-controlled neon sculpture which was displayed in a Los Angeles pedestrian arcade. Semicylinders that line up for 270 feet oscillate in intensity, tracking the movement of people through the arcade. The computer has 12,000 functions, and the sculpture consists of neon lights, plastic and stainless steel [41].

Another artist, Stephen Wilson, has experimented with interactive art in several environments [42]. In one experiment, the viewer could touch different parts of a painting which triggered responses relative to that spot from a video display. The viewer could choose from either a political or aesthetic frame of perception.

In the same article, Wilson describes an *Interactive Escalator* where riders could generate music as they passed sensors; more complex passages could be realized when people worked together. Finally, Wilson describes a computerized "Street Event" where passers-by could add five notes to a melody contained in the computer's memory. The participant could check the current status of this "community song."

3.7 Other Applications

In this final section on applications, the author has included several uses for computers in art that did not seem to fit into any of the major categories.

Frank Smullin of Duke University's Art Department has developed a system that enables a sculptor using tubular structures to design the complex miter intersections. The software is based on quantitative methods that are sufficiently complex and tedious that without a computer, few would attempt to use the method. The structure's balance can also be analyzed with the program [43].

Another application for computers in sculpture is described in [44]. Here, artist Deborah Blakely has taken an existing bronze sculpture and photographed it from several angles. The photographs were coded in black, white and grey before being transferred into the computer's memory. The image could then be manipulated, juxtaposed or stretched. Random deterioration of the image was introduced to simulate the erosion that occurs in the environment. Some of the resulting images were then transferred to canvas using acrylics.

One of the more creative presentations brought to the 1984 Symposium on Small Computers in the Arts by students from Virginia Commonwealth University was *Gretchen Prisoner of Love*, written by Nora Wilson [45]. An Apple computer was used to present a poem through the use of abstract, real and illustrative images. The result was quite effective as Wilson and her partner, Walter Wright, successfully solicited an emotional response to a medium which generally is considered to be capable of appealing only to the intellect. This novel use of computers in poetry is an excellent example of artists who have been able to go beyond the implementation of traditional art on a computer to create an artform which is greater than the sum of its components: written prose and visual imagery. The limitations imposed by an inexpensive system have caused the artist to apply a level of creativity which might otherwise have been overshadowed by technology.

4. CROSSING TRADITIONAL BOUNDARIES

There are an increasing number of groups in the United States that are directed toward the advancement of these new arts technologies. Likewise, there are a number of publications that are dedicated to, or devote a significant amount of space to new applications, current research and philosophical debates relating to the use of computers in the arts. These assemblages and publications serve as the impetus as well as provide a broad resource base for those who are pioneering this new field. A number of these groups and publications are presented in this section.

4.1 Small Computers in the Arts Network

SCAN is the most broad based group in existence today. Originally the Personal Computer Arts Group, SCAN is based in Philadelphia and is committed to the promotion of small computers in the arts through a monthly newsletter and through its annual Symposium on Small Computers in the Arts. The Philadelphia Computer Music Concert (which is no longer just computer music) is held in conjunction with the Symposium, which held its fourth meeting in October of 1984.

SCAN is unique in that it is involved in all application categories. Most of the other groups are dedicated to a particular category of applications, such as music or graphics. There are many obscure applications which get attention through SCAN that might otherwise go unnoticed. The proceedings of the Symposium have served as an important resource for this paper and typically contain papers written from the perspective of artists and engineers.

Although the Symposium is sponsored by the Institute of Electrical and Electronic Engineers Computer Society and the Association for Computing Machinery's Special Interest Group in Graphics (Delaware Valley SIGGRAPH), the participants (nearly 300 in 1984) come from all walks of life and are more likely to be oriented toward art than computer science or engineering. This combination provides for an unusual experience and offers a unique opportunity for people from "both sides of the fence" to meet each other to exchange ideas. One cannot help feeling that a number of artists will develop a strong interest in the technical aspects of these applications (and vice versa) as a result of these gatherings and the other work done by SCAN.

4.2 International Computer Music Conference

The ICMC, held annually in different locations around the world, is the premiere computer music conference. Lately, it has been held in Texas (1981), in Venice (1982), at the Eastman School of Music (1983), and the tenth annual ICMC was held in Paris (1984). It is co-sponsored by the Computer Music Association (CMA) along with a local group from the country where it is being held. The CMA, located in San Francisco, publishes the proceedings for the conference. When the conference was held in New York at the Eastman School, a reviewer for the *Computer Music Journal* observed that a noticable shift had occurred "from a *computer* music to a computer *music* conference" [46].

4.3 ACM/SIGGRAPH

The ACM is the principal organization for computer scientists and has numerous conferences annually on different topics. Siggraph (Special Interest Group in Graphics) is a part of the ACM and holds its annual conference during the summer. (Over 20,000 people attended the 1984 SIGGRAPH Conference held in Minneapolis.) Although SIGGRAPH's function is to promote the advancement of computer graphics in a general sense, their conferences seem to attract a large number of people who are primarily interested in the art-related applications.

The contrast between artists and the representatives of big business was evident at the 1983 Conference, as the more liberal factions in the crowd booed the impressive defense-related animations. This contrast was also evident in a lively discussion in which artists argued that the randomness of real life should be integrated into computer-generated film and art, while an auto industry executive was calling for more precision in the CAD graphics used in his industry [47]. Clearly, the answer to this conflict is that both are correct; that is, the development of independent systems can achieve both goals.

SIGGRAPH has a monthly publication, *Computer Graphics*, which is oriented toward the programming techniques and theory involved in accomplishing computer graphics. Another service of SIGGRAPH is the production each year of videocassettes with the highlights of all the animations that were submitted to the conference. The *SIGGRAPH VIDEO REVIEW* series is available through the SIGGRAPH office in Chicago. Each year's tapes are shown at SIGGRAPH and at the Symposium on Small Computers in the Arts, and contain a good mixture of work accomplished on microcomputers as well as the large supercomputers.

4.4 Center for Advanced Visual Studies

CAVS is affiliated with MIT and basically promotes the collaboration between artists, engineers and scientists. Many varied events and exhibitions are the result of work done here. This center in many ways represents the leading edge of technology in visual arts, providing a stimulating environment for the unique group of Research Fellows that are affiliated with CAVS.

The center is under the direction of Otto Piene, a well-known artist and teacher, and operates under the philosophy that "scientists, scholars, engineers, and artists should work together for a better spiritual and physical environment" [48]. The works created by the affiliates of CAVS may end up in any part of the United States or the world. Some of the more notable events that have resulted from the center (as presented in [48]) are described in the following paragraphs.

Centerbeam, held on the mall in Washington, D.C., involved a 144-foot straight structure carrying pipelines that enveloped gas, steam, water and electricity for sound and other communication equipment. Along the structure were works using holography, light, video and electronic and natural sound. Lasers were projected at night onto steam sculptures and helium-filled inflatables.

The Sky Art Conference, held in 1983 in Linz, Austria, included several of Otto Piene's helium sculptures. The hour-long performance of *Icarus*, a sky opera, involved a small orchestra, three operatic leads, a chorus, and laser images that were projected onto the airborne sculptures. Other events included the Vienna Philharmonic performing through speakers so large that the music could be heard from seven miles away, and a piece called *Sky Kiss*, in which Charlotte Moorman played a cello as she was raised off the ground by helium balloons. These events were a part of Arts Electronica, an annual festival for electronic arts. When the Sky Art Conference was held at MIT, *Sun Wheel*, a solar steam-driven kinetic sculpture by Joan Brigham, was displayed. (This work was also at the World's Fair in Knoxville.) A mirror collects and focuses light from the sun in such a way that steam is created internally and is directed so that the work is rotated.

Joe Davis, a colorful artist who is a Fellow with CAVS, is preparing for his latest expose, called *Ruby Falls*. This project is part of a NASA program where private individuals can send self-contained experiments up on the Space Shuttle. Joe's project employs an electron gun which will fire into the upper atmosphere, creating an artificial aurora borealis around the entire earth!

A video disc is available from CAVS which documents many of the events and projects in which they have been involved.

4.5 Other Notable Assemblages

Another unique program at MIT is the new h.D. degree offered in Media Technology. This program brings together research on new media for learning, telecommunications, publishing, broadcasting, cinema, visual arts and music. The initial areas of concentration are electronic media, learning technology and computer music. There will be a Media Laboratory that will include an experimental theater and

state-of-the-art working conditions for video, computer graphics, holography and computer music.

The New York Institute of Technology (NYIT) and Ohio State University (OSU) both have extensive graduate programs where computer animation is being researched. There is a severe shortage of experts in this area as related to the demand from the broadcasting and film industries. This has led to the formation of commercial companies affiliated with both OSU and NYIT to help fill this need. Cranston-Csuri Productions at OSU has been responsible for many of the dazzling animations seen in television commercials and network logos. Computer Graphics Lab, Incorporated is the commercial arm at NYIT whose work has also been well represented on television. CGL has also been working on a film called *THE WORKS*, which is completely animated by computer. There are several non-academic production houses that have also been in the forefront of computer animation: Digital Effects, Incorporated; Digital Productions; MAGI; Robert Abel and Associates; Pacific Data Images; Lucasfilm, Incorporated and Information International, Incorporated.

Another major graphics group which has recently been formed is the National Computer Graphics Association (NCGA), which drew in excess of 24,000 people to its 1983 conference! DIGICON, held in Canada in 1983, (as discussed in section 3.1) plans to have its second conference during the summer of 1985, again in Vancouver. Its sponsor, the International Computer Arts Society, has its own publication called *Page*.

Already mentioned in section 3.1 are the Center for Computer Research in Music and Acoustics and the Computer Audio Research Laboratory, both in California. These facilities, along with the one at MIT, are among the leaders in computer music research.

The California Institute of the Arts has an extensive undergraduate curriculum in electronic music, theater technology, animation, videographics and audio engineering. It seems to be the only major arts school that emphasizes these technologies (more on this observation in section 5).

Another group is the Council on Technology, the Arts, and Cultural Transformation. Their purpose is to "foster the furtherance of interaction between technologists and artists for their mutual benefit" [49]. They sponsored a workshop in 1980 on the interaction of arts and technology. The same issue of *Leonardo* which contains [49] also contains a review of the 1980 conference, Art in a Technological Society, where fifteen proposals were presented to further arts and technology [50].

EPCOT CENTER at Disneyworld is a showcase for technology and computers and has a number of exhibitions related to the arts. In one exhibit, *Stepping Stones*, a person can step on different "areas," causing various musical sounds to be generated. The areas are grouped by the following timbres: voices, percussions, violins and bells. Another exhibit has optical sensors that enable the participant to move his or her hands as a conductor would to direct an *Electronic Symphony* [51]. These interactive artforms are excellent examples of how computers can be used creatively by the average person to create a personal expression of art.

4.6 Periodicals and Publications of Interest

As a resource base for persons interested in further research, there are several publications that are either dedicated to or devote a significant amount of space to computers in the arts. Some of these that have already been mentioned are the *Proceedings of the Symposium on Small Computers in the Arts*, the *SCAN Newsletter*, the *Computer Music Journal*, *Leonardo*, *Page*, *Theater Design and Technology* and *Computer Graphics*. Another publication which has been the source of a number of references for this paper is *Creative Computing*. *Computer Graphics and Art* was a sister publication of *Computers and People*, but in 1982 it was merged into the latter and serves as a useful reference. *Theater Crafts* has had a number of informative articles recently dealing with computer applications in the theater.

Visual music is explored in two recent books. *Digital Harmony: On the Complementarity of Music and Visual Art*, by John Whitney, is reviewed in [52]; and Ronald Pellegrino's book, *Electronic Arts of Sound and Sight* is reviewed in [53]. Melvin Prueitt recently published a book entitled *Art and the Computer*, which he discusses briefly in [54]. Another recent publication is *Art and Computers, the First Artificial Intelligence Coloring Book* by Harold Cohen et al. (*AARON*, described in section 3.2, was developed by Cohen.)

5. COMPUTER ART--FOR ART'S SAKE

It is not surprising to note that the majority of the university programs that are involved in computer applications in the arts are engineering or computer science oriented. The California Insti-

tute of the Arts, due to its long-standing relationship with the film and television industries in Los Angeles, is a notable exception. The School of the Arts at Virginia Commonwealth University is also very involved, as is the Massachusetts College of Art. It is in arts schools such as these that the greatest potential for progress exists; unfortunately, lack of money and qualified faculty has proven to be a major obstacle in achieving this potential. However, a school can begin to overcome these two problems by capitalizing on the exposure that can result from the involvement in a new technology. Also, as systems become more friendly, it will no longer be necessary for an arts school to employ highly trained engineers and computer scientists to implement computer applications.

There can be many benefits as more arts schools are able to integrate these new technologies into their existing programs. William Buxton, in his keynote address at the 1982 Symposium on Small Computers in the Arts, warned that cyberphobia (the fear of computers, machines and technology) will be a major problem in our society [55]. Not only will the students directly involved with these computer applications overcome any cyberphobic fears that they may have, but their fellow students, through a more indirect exposure, will most likely gain a better appreciation for technology. Furthermore, the society as a whole will be able to see computers in an aesthetic context as these new artforms and techniques for accomplishing art become more visible.

6. CONCLUSION

What is the meaning of all this? Where are we headed? How well will the computer art of today hold up against the supercomputer art that will surely exist in the next century? No matter how these questions are answered, the artists who are pioneering this rapidly expanding field are establishing a new direction that will fundamentally alter the way that art is accomplished and enjoyed.

7. REFERENCES CITED

[1] William Matthews, "Computers, Music, and the Arts: A Liberal Arts College Course," *Proceedings of the 3rd Symposium on Small Computers in the Arts*, 1983, p. 86.

[2] Grace C. Hertlein, "Computer Art: Review, 1968; Survey, 1978; Predictions, 1988," *Computers and People*, vol. 27, no. 8-9, Aug./Sep. 1978, p. 8.

[3] Donna Mansfield, "Computer Animation as an Art Form," *Proceedings of the 3rd Symposium on Small Computers in the Arts*, 1983, pp. 13-19.

[4] Willard F. Bellman, *Scene Design, Stage Lighting, Sound, Costumes, and Makeup: A Scenographic Approach*, Harper and Row, p. 365.

[5] Ken Sofer, "Art? Or Not Art?" *Datamation*, vol. 27, no. 11, Oct. 1981, p. 118.

[6] David B. Clark et al., "Computer-Aided Sight Reading," *Creative Computing*, vol. 6, no. 6, Jun. 1980, p. 84.

[7] John Craig, "The Music Men and Their Incredible Printing Machine," *Creative Computing*, vol. 5, no. 6, Jun. 1979, p. 48.

[8] Goffredo Haus, "EMPS: A System for Graphic Transcription of Electronic Music Scores," *Computer Music Journal*, vol. 7, no. 3, Fall 1983, p. 31.

[9] Przemyslaw Prusinkiewicz, "INTER-SCORE--An Interactive Score Editor for Microcomputers," *Proceedings of the 4th Symposium on Small Computers in the Arts*, 1984, p. 58.

[10] Michael Keith, "Automatic Computer Composition of Bluegrass Tunes," *Proceedings of the 2nd Symposium on Small Computers in the Arts*, 1982, p. 29.

[11] Marjorie Lyon, "A Third Medium for the Music Composer: Computers," *Creative Computing*, vol. 6, no. 6, Jun. 1980, p. 64.

[12] Julie White, "DIGICON 83," *Computer Music Journal*, vol. 8, no. 1, Spring 1984, p. 43.

[13] Patricia Smith, "Computers Make Music," *Creative Computing*, vol. 9, no. 7, Jul. 1983, p. 111.

[14] Mike Williams, ed., "Fountains of Light," *Lighting Dimensions*, vol. 8, no. 4, Jul./Aug. 1984, p. 20.

[15] John Strawn et al., "Report on the 1981 International Computer Music Conference," *Computer Music Journal*, vol. 6, no. 4, Winter 1982, p. 11.

[16] Barbara Mackowiak, "EASEL: A System for Artists," *Creative Computing*, vol. 10, no. 2, Feb. 1984, p. 44.

[17] Ame Choate Flynn, "Graphics Software Interaction Using the Apple Computer," *Proceedings of the 2nd Symposium on Small Computers in the Arts*, 1982, p. 35.

[18] Alice Kaprow and Jane K. Shafran, "The New Studio: The Computer Graphics Workstation," *Proceedings of the 4th Symposium on Small Computers in the Arts*, 1984, p. 52.

[19] Walter Wright, "Using an Artist's Workstation," *Proceedings of the 3rd Symposium on Small Computers in the Arts*, 1983, p. 128.

[20] Demetrios Michalopoulos, "Computer Artist to Recreate Great Olympic

Moments Within Hours," *Computer*, vol. 17, no. 7, Jul. 1984, p. 119.
[21] Martin Nisenholtz, "The Electronic Gallery," *Proceedings of the 3rd Symposium on Small Computers in the Arts*, 1983, p. 83.
[22] David Em, "Computer Art and Butterfly Nets," *EDN*, vol. 26, no. 20, 14 Oct. 1981, p. 307.
[23] James L. Hockenhull, "Courting the Digital Muse," *Creative Computing*, vol. 8, no. 2, Feb. 1982, p. 84.
[24] Marc Kirkeby, "Computer Graphics: Is There Life After TRON?" *American Film*, vol. 8, Jan./Feb. 1983, p. 42.
[25] John P. Riganati and Paul B. Schneck, "Supercomputing," *Computer*, vol. 17, no. 10, Oct. 1984, p. 107.
[26] Gary Demos, Maxine Brown and Richard Weinberg, "Digital Scene Simulation: The Synergy of Computer Technology and Human Creativity," *Proceedings of the IEEE*, vol. 72, no. 1, Jan. 1984, p. 22.
[27] Ware Myers, "Computer Graphics: The Need for Graphics Design," *Computer*, vol. 14, no. 7, Jul. 1981, p. 86.
[28] Linda Hirschmann, "Computers and Dance," *Creative Computing*, vol. 5, no. 8, Aug. 1979, p. 42.
[29] Randy Ring, "Offbeat Careers With a Byte," *Graduating Engineer*, vol. 5, no. 2, Nov. 1983, p. 61.
[30] USITT Graphics Standards Board, "Recommendations for a Standard Graphics Language in Scenic Design and Technical Production," *Theater Design and Technology*, vol. 18, no. 1, Spring 1982, p. 8.
[31] William Daniel File and Kenneth I. Joy, "The Computer as an Aid in Scenic Design," *Theater Design and Technology*, vol. 19, no. 1, Spring 1983, p. 18.
[32] Robert Reinecke, "Product Report," *Theater Design and Technology*, vol. 20, no. 1, Spring 1984, p. 38.
[33] Margaret Benyon, "On the Second Decade of Holography as Art and My Recent Holograms," *Leonardo*, vol. 15, no. 2, Spring 1982, p. 89.
[34] Sergio Cavaliere et al., "From Computer Music to the Theater: The Realization of a Theatrical Automation," *Computer Music Journal*, vol. 6, no. 4, Winter 1982, p. 22.
[35] John Richards and Ian Craven, "An Experimental All-Digital Studio Mixing Desk," *Journal of the Audio Engineering Society*, vol. 30, no. 3, Mar. 1982, p. 117.
[36] M. H. Jones et al., "An Advanced Computer-Assisted Sound Mixing System for Film and Video Post-Production," *SMPTE*, vol. 91, no. 10, Oct. 1982, p. 931.
[37] C. Roads, "A Conversation With James A. Moorer," *Computer Music Journal*, vol. 6, no. 4, Winter 1982, p. 10.
[38] Robert Duffy and Joseph Roizen, "A New Approach to Film Editing," *SMPTE*, vol. 91, no. 2, Feb. 1982, p. 198.
[39] Stuart Gaines, "Lights, Cameras. . . Computers," *Discover*, vol. 5, no. 8, Aug. 1984, p. 76.
[40] Richard Hackmeister, "The Computers of Hollywood," *Creative Computing*, vol. 7, no. 6, Jun. 1981, p. 82.
[41] Michael J. Crosbie, "Computer Controlled Flickering Neon Sculpture," *AIA Journal*, vol. 72, Jun. 1983, p. 31.
[42] Stephen Wilson, "Environment-Sensing Artworks and Interactive Events: Exploring the Implications of Microcomputer Developments," *Leonardo*, vol. 16, no. 4, Autumn 1983, p. 288.
[43] Frank M. Smullin, "PIPEDREAM--A Complete CAD CAM System for Tubular Sculptures," *Proceedings of the 3rd Symposium on Small Computers in the Arts*, 1983, p. 112.
[44] Deborah Blakely, "New Tool for the Sculptor--Computerized Clay," *Creative Computing*, vol. 5, no. 6, Jun. 1979, p. 46.
[45] Nora Wilson and Walter Wright, "Computers and Poetry," *Proceedings of the 4th Symposium on Small Computers in the Arts*, 1984, p. 76.
[46] Robert Gross et al. "Report on the 1983 ICMC," *Computer Music Journal*, vol. 8, no. 2, Summer 1984, p. 7.
[47] Tim Onosko, "SIGGRAPH '83: The Graphics Event of the Year," *Creative Computing*, vol. 9, no. 11, Nov. 1983, p. 169.
[48] Randy Ring, "Engineering for Art's Sake," *Graduating Engineer*, vol. 5, no. 4, 1984, p. 83.
[49] Herbert Shore, "Report on the International Workshop-Seminar on the Interaction of Arts and Technology," *Leonardo*, vol. 15, no. 1, Winter 1982, p. 49.
[50] Herbert Shore, "Report on the Conference *Art in a Technological Society*," *Leonardo*, vol. 15, no. 1, Winter 1982, p. 51.
[51] Jim McConkey, "The Second Symposium on Small Computers in the Arts," *Computer Music Journal*, vol. 7, no. 3, Fall 1983, p. 25.
[52] R. Berger, review of *Digital Harmony: On the Complementarity of Music and Visual Art*, by John Whitney, in *Computer Music Journal*, vol. 6, no. 4, Winter 1982, p. 45.
[53] John Strawn, review of *Electronic Arts of Sound and Sight*, by Ronald Pellegrino, in *Computer Music Journal*, vol. 7, no. 4, Winter 1983, p. 59.
[54] Melvin L. Prueitt, "Cray on Art," *Discover*, vol. 4, no. 11, Nov. 1983, p. 68.

[55] Jim McConkey, "The Second Annual Symposium on Small Computers in the Arts," *Computer Music Journal*, vol. 7, no. 3, Fall 1983, p. 25.

[56] Tom DeWitt and Phil Edelstein, "Pantomation: A System for Position Tracking," *Proceedings of the 3rd Symposium on Small Computers in the Arts*, 1983, p. 61.

[57] Cathy M. Stadler, "Computers and Choreography," *Proceedings of the 3rd Symposium on Small Computers in the Arts*, 1983, p. 107.

[58] Jim McConkey, "Report on the Third Annual Symposium on Small Computers in the Arts," *Computer Music Journal*, vol. 8, no. 2, Summer 1984, p. 41.

NEW PAGE: The Changing Face of Composition

Joan K. Shafran

Computer Graphics Research/Design

ABSTRACT

Composition and experimentation with the presentation of information is not new. Computer technology has opened up more possibilities for the creative use of digital typography and design of the page than ever before possible. To understand the implications of these technologies in relation to the idea of 'the page', it is important to know what has been done, what is being done and what can be done.

INTRODUCTION

The creative and experimental use of typography as the predominant graphic element is not new. Historically, we can see stunning examples of illustrative type in old religious manuscripts, showing a unity between the content and the graphical elements of the page.

As technology progressed, and new methods of tranmitting information evolved, so did both the form and content of the page. As we examine modern page composition and typography, we can see many links with the past, and we can begin to see the ways these traditions will change and evolve with the use of new computer technologies.

We can examine the presentation of information by exploring some of the traditions of using text as a graphical element and by extending this view to looking at the effects of technological innovations. This can be applied not only to the realm of practical and commercial applications, but also to the world of art and poetry.

It is important to understand the experimental and artistic concerns of typography as well as practical ones. Although computer systems have certainly allowed for enhancements never considered in traditional typography, the effects of technical innovation are subtle, and we have yet to explore how the "new page" may evolve.

EARLY USERS OF PRINTED TYPOGRAPHY

Typography can be defined, in its most traditional terms, as the art or skill of designing communication by means of the printed word. It should not interfere with what is trying to be communicated. Typography is also a way of drawing our language and is designed to convey the sounds of the words through which the meaning is conveyed or simply unspoken thought that comes from the brain.

In a sense typography is an interpretation or an enhancement of the meaning. It should convey the rhythm of the line, and the interrelationships of all the elements to each other. It can be placed in combination with an image.

The early innovators at the time of Gutenberg were mimicing the hand written page. Through much experimentation, the printer became the designer, choosing formats, images and relationships between the two. These e4xperimenations have resulted in what we traditionally think of as a printed page.

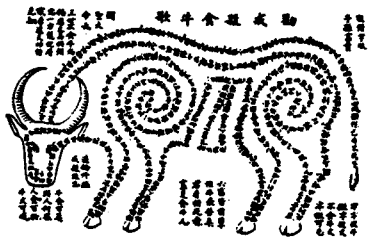


Figure 1. Early Chinese exhortation against killing oxen. Early example of the use of type to visualize content.

As early as the fifteenth century, there is a split in thought about how the typography should be used, the word as the image or the word reflecting the image. It was people such as Baskerville who thought of the creation of type as an art form and people such as Caslon that referred to it as a trade.

The early book designers and producers in a sense broke away from the tradition of calligraphy and relied heavily on the development of new equipment to guide the new styles and innovations.

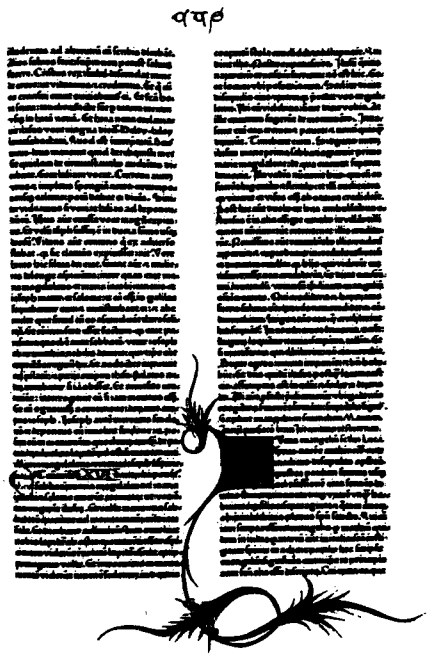


Figure 2. Bible in Latin, Johann Fust & Peter Schoeffer, Mainz, Germ 1462.

The Nature of Composition

True composition is the relationship of the words and letters on the page no matter where they are on the page. The image is usually concidered only by an area. The actual functions of composition are grouped into three levels. 1. Basic: font selection, hyphenation and justification, point size, word spacing, letter spacing and kerning. 2. Intermediate: area composition, relationship of groups of words and letters to the page, How They Look, column width, leading and shaping. 3. Advanced: formating, vertical justifiaction, copyfitting, layout and pagination.

Breaking Away From Tradition

At the latter part of the 18th century, a series of groups radically changed typographic and composition traditions, and influenced the establishment of new criteria for art making and page design. Groups such as the Futurists wrote manifestos on a " universal communication " through the use of typography, the idea of assymmetry and decoration came into play with the Art Nouveau period.

A new form called " visual poetry " or the use of typography as the prodominent visual element became an accepted form. Supported by such groups as the Constructivists, who saw arranging page like information in archetectural grids with assymetrical balance, using 3D letter forms and a strongly controlled sense of space. And the Bauhaus, in Germany, who felt the typographer and composer of pages should be free of the rigid one or two column page and not labor over preconceived notions, but design and compose the page with reflection of the content and with understanding of the technology being used to reproduce it.

The result of this experimentation was a series of directives called ` The New Typographers ' by Jan Tschicold in 1931. It is these ideas that drive the

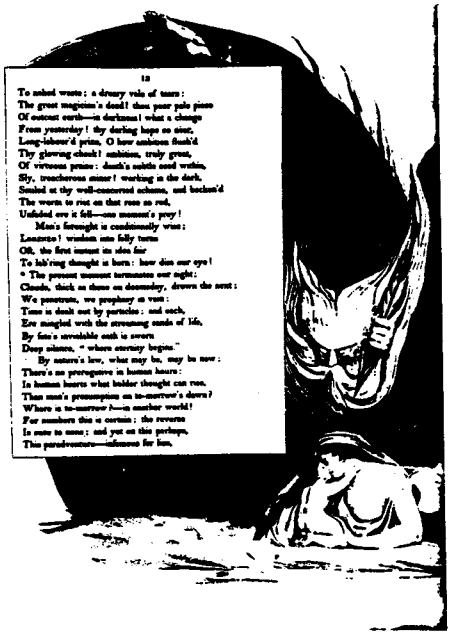


Figure 3. Illustration, Wm. Blake, 1797.

way we think about composition today. Tschichold ideas included the following:

- freedom from tradition
- geometric simplicity
- contrast of typographic material using different font styles and sizes
- machine dependency
- Consistency

The result of these 'revolutionaries' was a redefinition of what the 'page' was and a new set of rules. In a sense it was the loosening of traditional obsessions only to develop new obsessions. We see this in the compulsive use of white space, the rigid use of margins. Also, the typography and the design is no longer a self-expression (as the visual poets and DADAists would have hoped), but a methodolgy by which content could be slipped into a format and then read.

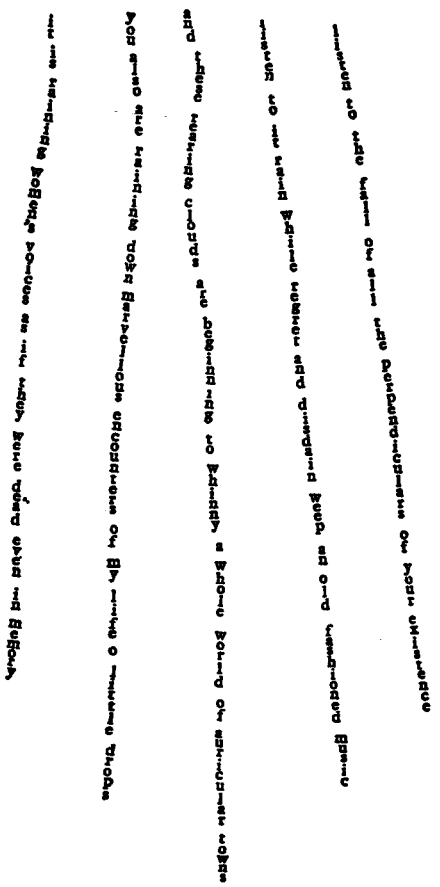


Figure 4. It' Raining, Appollinaire, 1918. Words are used visually to reflect the content and meaning.

ELECTRONIC PAGE and COMPOSITION SYSTEMS

Pagination and Composition systems were designed with these constraints in mind. They initially were developed to mimic and to speed up in-house publishing departments, newspaper layout,and personal publishing with the development of small computer software. They are concerned with the office page of 8 1/2 x 11.

Pagination and Composition is the electronic assembly of all componants of a document for output of completely composed pages. The formats, fonts, sizes and image areas are all designated by the operator and the database. The componants nessecary for a completed document consist of text, line art, halftones, full color, headings, rules, borders and folios. In the past these were handled in different phases and often by different people, and then 'composed' by paste-up artists or at the printer. With the computer these functions can happen simultaneously.

The black and white page system must handle all the typographic functions such as H & J, creation of column, add generic typesetting codes painlessly, observe image space,and footnotes. They should have an on-line dictionary, foriegn language, tabular material, math, PI, deal with running heads, format instructions, and track the work in progress. The ideal system should do all of this, plus allow the user or designer to interact with the pages.

To date most of these sytems are mimicing the photoset page. There is much experimentation needed to develop an understanding of the constraints and enhancements of these composition systems and integrate them into the conceptualizing process.

NEW PAGE

With the increasing number of computer systems, there will be an further changes in the way textual information is used and formatted. The most dramatic changes will occur in the composition environments and with the 'soft page'.

With the computer comes new ways of thinking about the page. It is no longer nessecary to think in linear terms. The videodisc has shown that random access 'books' are possible.

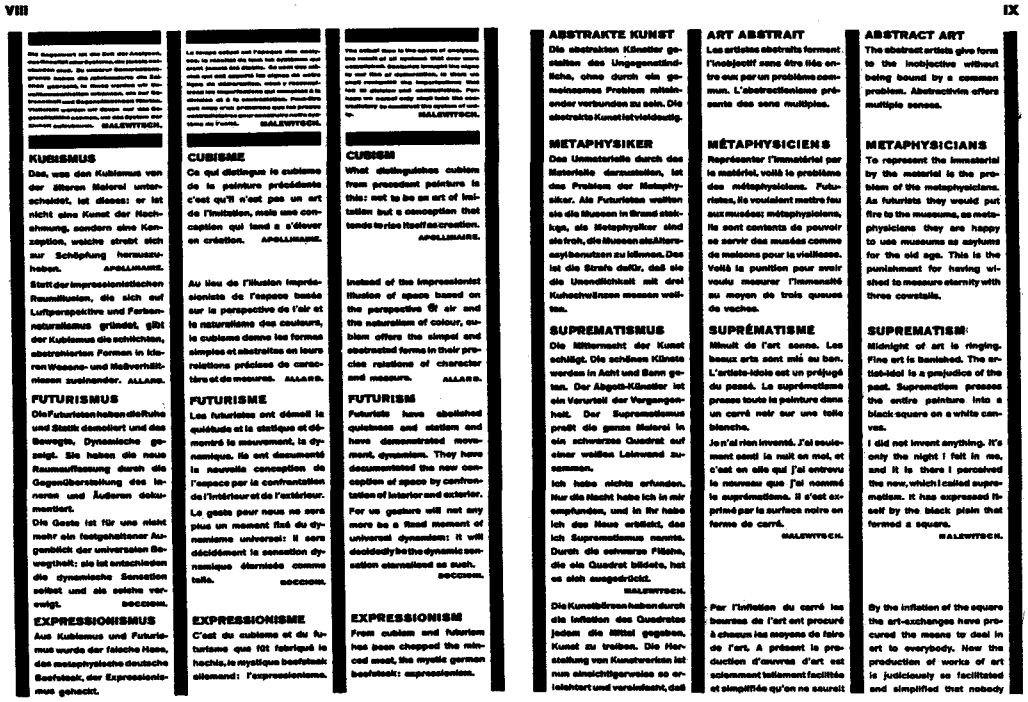
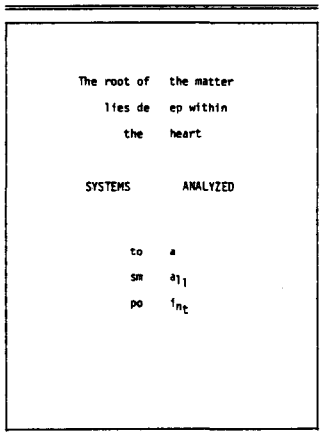
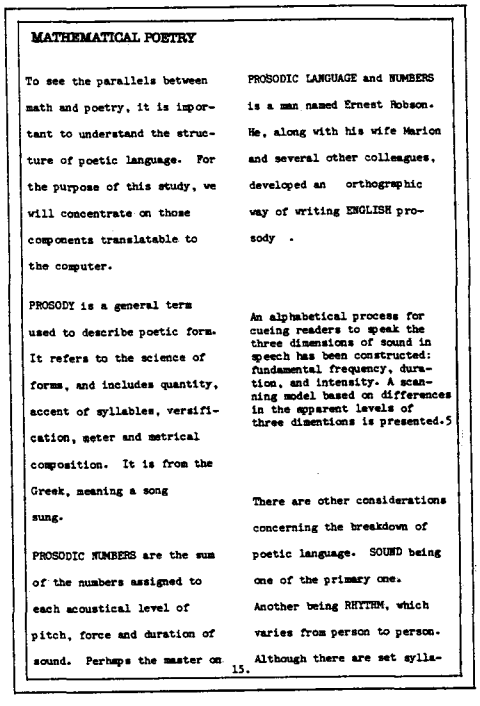


Figure 5. Page opening from "Die Kunistimen", a manifesto. The content or text is contained within a very imposing ruled grid.

That documents can be planned on the computer. Technolgies such as Videotex offer us new ideas of what publishing can mean.

This raises many questions. What are the factors of this sort of composition? Certainly brevity is one given the size of the monitors we use. How will this technolgy effect the visual form; is there a new visual form?
The computer offers us the option of creating a look that does not exist. The users of these systems will begin to establish yet another presentation level that enhances the credibility of this new form.




MATHEMATICAL POETRY

To see the parallels between math and poetry, it is important to understand the structure of poetic language. For the purpose of this study, we will concentrate on those components translatable to the computer.

PROSODY is a general term used to describe poetic form. It refers to the science of forms, and includes quantity, accent of syllables, versification, meter and metrical composition. It is from the Greek, meaning a song sung.

PROSODIC NUMBERS are the sum of the numbers assigned to each acoustical level of pitch, force and duration of sound. Perhaps the master on

PROSODIC LANGUAGE and NUMBERS is a man named Ernest Robson. He, along with his wife Marion and several other colleagues, developed an orthographic way of writing ENGLISH prosody .

An alphabetical process for cueing readers to speak the three dimensions of sound in speech has been constructed: fundamental frequency, duration, and intensity. A scanning model based on differences in the apparent levels of three dimentions is presented.5

There are other considerations concerning the breakdown of poetic language. SOUND being one of the primary one. Another being RHYTHM, which varies from person to person. Although there are set sylla-

15.

Figure 6. A page from the Poetry Generator, thesis project, MIT.J.Shafran,1980. This was designed and implemented on a word processing system.

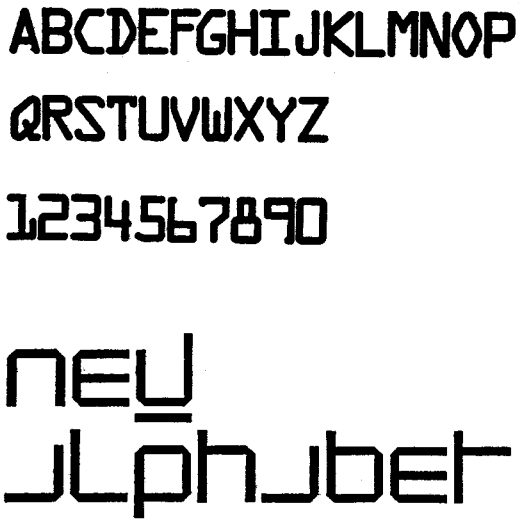


Figure 7. Top: Herbert Spencer, 1968.An alphabet for electronic scanning. Bottom: Wim Crouwel, an alphabet designed for CRT typesetting.

The Role of Visual Poetry and The Computer

Concrete poetry, which began as a movement in the 1940's, involves the breakdown of linear structures. This movement rebelled against the lack of self expression and the conditions placed on typographic design,and composition, establishing new ways of communication transforming the word and the poem into an active structure on the page.

As defined by visual, mathematical and computer poets, Visual Poetry in relationship to the computer have the following elements:

-the unity of the typography and the content

-the perception and synthesis of language

-associations and relationships of letterforms, both mathematically and visually

-hidden meaning

-flexibility of both the letter design and the tpographic compostion

-Combining of many elements

-higher levels on interaction

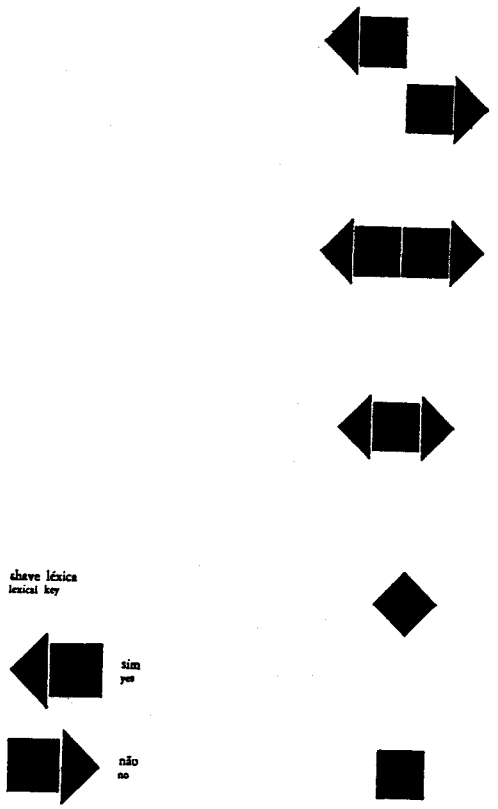


Figure 8. "Code Poem", Luiz Angelo Pinto, 1965. A symbolic Concrete poem.

With the further experiemtation of these technologies and the development of new ones, the old linear structures of our page will no longer be valid, establishing a new way of communicating this form of information. In visual form the word can be transformed to an active structure allowing the content to be perceived rather than read.

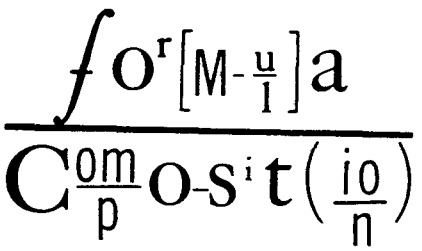


Figure 9. Mathemtical poem, Bern Porter. From Against Infinity, 1979.

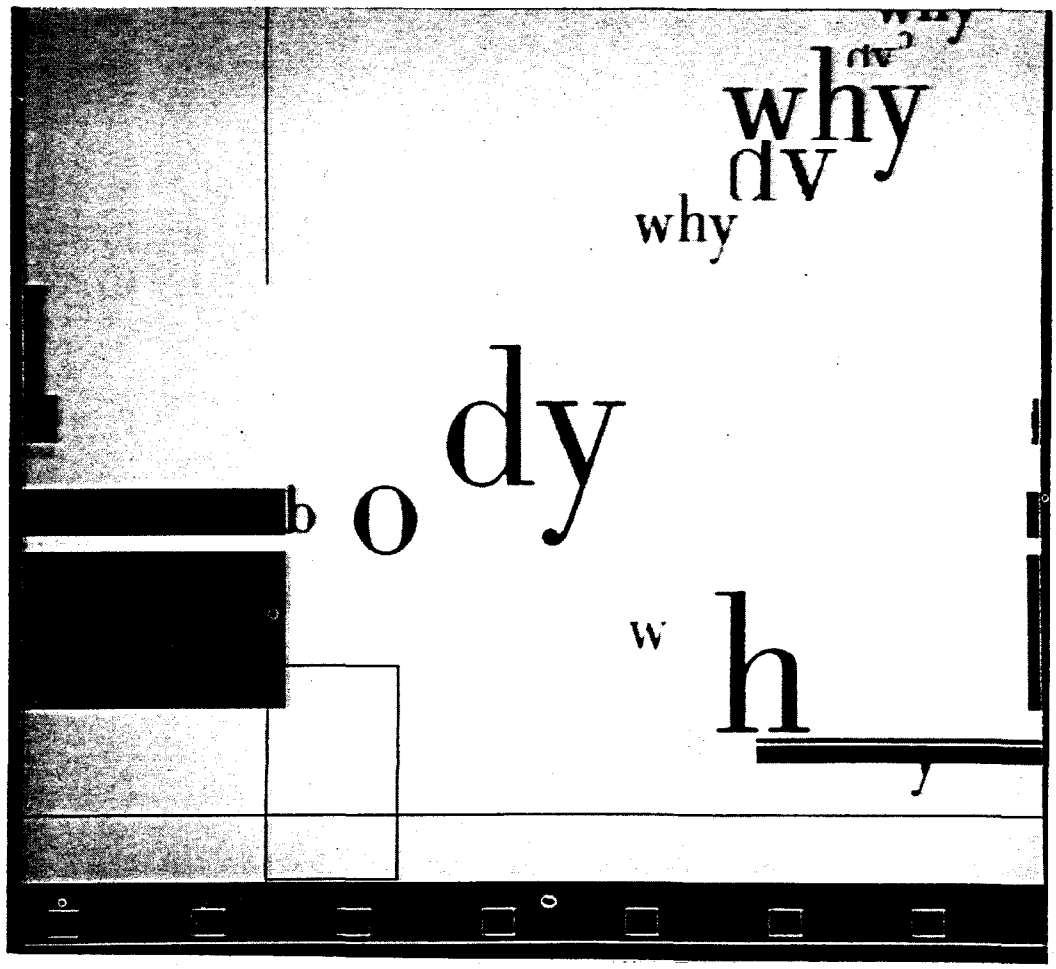


Figure 10. BoDy POem, Joan Shafran,1979. A page from a soft poetry book.

NEW AESTHETICS FOR MUSICAL VARIATION

Steven Berkowitz

Tyler School of Art, Temple University, Philadelphia

We recognize images by identifying the internal relationships of form and content from which they are built. We can alter aspects of these images as long as it done consistently, preserving the integrity of the relationships. If an image can be represented numerically, we can then perform mathematical alterations, hopefully creating interesting new variations. Mathematics can be the perfect objective modifier and its use can dictate a new aesthetic of image making.

THE RECOGNITION OF IMAGES

We are accustomed to recognizing altered visual images. We have seen dozens of take-offs on the Mona Lisa for example. They have run the gamut from Marcel Duchamp to the cover of Mad Magazine. More recently computer generated alterations have utilized mathematics to modify the image. Yet there is never a doubt what the image is because that set of internal relationships remains intact.

In music, however, we are much more limited in how we differentiate between variations on a theme as opposed to that which we give a new title and a new copyright. We vary the arrangements of instruments and tempos but as yet we do not listen to music in terms of internal relationships that can be shifted.

THE ALTERATION OF IMAGES

Traditional variations in orchestration change timing merely by increasing or decreasing the tempo and change pitch merely by raising or lowering the key. A musical score can be viewed as a field of events whose positions can be manipulated in various ways. The computer allows us to alter time and pitch by redistributing notes, creating different patterns from existing ones. It allows us to expand scores with fractal geometry. Pieces can be analyzed in terms of their density and direction. The ability the computer gives us to manufacture new sounds opens ways of linking timbre to compositional structure.

DATA SET-UP

Only a minimum of information is in fact needed to define a score. The scoring system that is used to experiment with the following compositional techniques comes from the early days of MUSIC 4BF running on a Cyber mainframe. In that system each note in a score occupied a row of an array (actually, a punched IBM card. Remember BATCH systems and card readers?) Each note required at the very least an instrument number, a start time, a duration, a pitch, and an amplitude. The same data set-up is used here, and all numbers are considered to be relative rather than absolute times and pitches. All the following transformations are done with only this much specification. Amplitudes are not really dealt with here because the densities and distributions of the notes have a great influence on the dynamics. However, the same manipulations can apply to amplitude.

To utilize the following score manipulations a digital music system is needed which allows access to the note files so that any values can be altered and the results can be replayed as a new score.

MOVEMENT

The most standard type of modification one can use is movement. In this case every note is affected equally by the translation. The notes are treated as a block of information that is moved either back or forth in time or transposed up or down in pitch.

TIME- The time location shifts forward or backward uniformly:

START = START + C

This modification is used to shift a section of a piece in time and does not alter the composition. This application is most useful when building a piece in a modular fashion, where sections are used more than once and placed in different positions against other sections.

PITCH- The notes transpose up or down uniformly:

PITCH = PITCH + C

This transposition is a rather standard operation, although special effects can be created by how the relative pitch values are assigned to the notes of a particular key. If the key signature is ignored and the notes are merely shifted, one is changing key. If, however, the scale is kept in the same key when the pitches are shifted the intervals will be altered.

LINEAR SCALING

If the values to be affected are multiplied by a given constant, then the range of these values will swell or shrink. The intervals of time and especially of pitch can be altered through ways that are not available in analog systems, creating a new range of compositional options.

TIME- The overall tempo increases or decreases:

START = START * C : DUR = DUR * C

With analog technology we only have the ability to change tempo by speeding up or slowing down a tape deck. Unfortunately, this means of changing tempo also alteres the pitch. With digital technology we can now either recalculate the time values in the score or change the tempo on a digital sequencer or keyboard recorder. This process changes the start times and durations of all notes uniformly without distorting pitch whatsoever.

PITCH- The range of pitches increases or decreases:

PITCH = PITCH * C

If the pitch range in a one octave piece is doubled to two octaves, all half-steps become whole-steps, etc. If the pitch range is reduced by half, the result is a quarter-tone piece. The real fun begins when the pitch values are rescaled by non-integer numbers, creating non-standard tunings. The values could be quantized back into a standard scale yielding a less drastic modification of the score, if so desired.

QUANTIFICATION & SCALE INJECTION

Quantizing has shown up recently in rhythm machines and digital synthesizers and sequencers as a means to clean up uneven keyboard technique or to just put a strict, even timed feel into a piece. The process actually removes the inbetween time values from the score and forces each note into a specific time slot. This same procedure can be used to standardize a pitch scaled score as mentioned earlier.

When generating scores by algorithm, ranges of notes can easily be created that are larger than can be played. In this case the ability to quantize the pitch values is necessary. In this case the process is used to squeeze the pitch values together into a more reasonable range. To compute this, first the highest and lowest note values are found, then the range of pitches is calculated, and finally all pitch values are scaled by the factor of the desired pitch range divided by the existing pitch range. If the pitch values are kept as integers when using relative notation, there will be a rounding off due to the compression of a larger range into a smaller one. This rounding off alone can present some interesting shifts in a composition.

Once the desired pitch range is set, the notes can be changed from relative to absolute notation by substituting the frequencies or note names for the relative numbers. If, for example, a scale of six notes per octave over five octaves is used, the pitch range will be thirty notes of specific frequencies. One can quantize a calculated score with a large pitch range down to thirty pitch values, then inject the predetermined frequencies of a scale into the score. This technique becomes even more important when using the following compositional tools.

EXPONENTIAL SCALING

This scaling can be looked at as a compression of data. If a group of notes is drawn on a page, then one edge of that page is curved away from the viewer and perceived straight on, the notes on that curved surface would appear closer together. In the same sense notes can be pushed together or pulled apart in a gradual fashion by scaling all values by an exponent. Exponents greater than one will push, while exponents less than one will pull notes apart, if the piece is then rescaled back to its original range. This process is analogous to plotting a score on logarithmic rather than linear graph paper.

TIME- The tempo retards or accelerates:

START = START ^ C : DUR = DUR ^ C

If the time compression is placed at the end of the piece, a continual diminution of note durations occurs. If the compression is placed at the start of the piece there will be a gradual slowing down of the piece. Time compressions can be placed in the middle of a piece also, yielding temporal swells. To do so the piece must first be moved so that TIME = 0 is at the center of the compression (negative time precedes 0, positive time follows). The exponentiation is then performed, preserving the sign of the negative time values. The piece is then rescaled back to its original length so that the earliest time is again equal to 0.

PITCH- The placement of the pitches centers around a given value (focal plane):

PITCH = PITCH ^ C

To place most of the notes of a piece in a specific pitch range, a combination of transposition and exponeniation can be used. First transpose the pitches so the central pitch (focus) becomes relative pitch 0. Pitches below zero become negative negative while pitches above are positive. Perform the exponentiation, preserving the sign of the original values. Rescale the pitches back to their original range and move the piece back to its original position by transposing again by the original transposition value. The degree of pitch compression depends on the exponent used and the original range of the piece.

The piece may have to be either quantized or have a scale injected to make it come out a reasonable sounding item- but who's to say what's reasonable?

DENSITY & MEAN PITCH

For the sake of analysis it helps to be able to visualize a score as a field of events in time and pitch space. Using the X coordinate to

indicate time and the Y coordinate to indicate pitch, graph each note as a line using the start time, duration and pitch to define the coordinates. Color can be used to indicate instrument assignment. This allows the structure of a musical piece to become readily visable. To take this a step further it is very easy to graph the average pitch and the density per unit time. When these functions are combined together the changes in direction of pitch the piece takes through time and how the dynamics are affected by density can easily be seen. These attributes of a score can become motivating factors in the construction of new compositions.

FRACTAL PATTERNING

To put it simply, fractals are structures that contain fractional parts that duplicate the overall structure at a much smaller scale. Looking at any small section of a fractal gives a good indication of how the entire piece is constructed. This notion has been used to create mathematical models of naturally occuring structures such as landscapes and coast lines. Musically it can be used to create patterns of patterns that act like variations on a theme.

A simple fractal variation replaces each note with the entire piece, transposed to start on the pitch of the note being replaced. What happens is a complex score that can be generated quite quickly. The notes overlay somewhat like a round with the parts shifting in pitch. This entire technique works best using phrases to create scores, otherwise any longer melody can create overwhelmingly large compositions.

A second fractal variation replaces each note with the entire piece, transposed as described above. The durations are compressed so the piece does not extend past the last note of the original score. This compression is another way of creating a build-up in density and a reduction in durations.

Another fractalization again replaces each note with the entire score, transposed as above. In this case, however, the durations are rescaled according to the relationship of the duration of the note being replaced to the duration of the first note of the piece. Here we introduce an actual change in scale of the time component. This takes us closer to the self-similarity principle inherent in the fractalization process.

A fourth variation considers both the time and pitch factors. Each note is replaced with the entire score and transposed as above. The durations are also rescaled. The pitch range of each replacement score is rescaled according to the duration relationship previously described. This results in a series of the same score rescaled in both time and pitch that graphically appears as a myriad of facsimile scores existing in different sizes. What is heard is a recapitulation of the phrase which is altered in both time and pitch.

If the original melody phrase is organic, i.e. asymmetrical yet not random, then the resulting fractal score mimics the complexities of pattern found in natural structures. In this way one can create scores that are models of the physical world, which is exactly the author's point in composing music.

WAVEFORMS & MELODIES

If a digital music system allows writing a waveform directly into memory, the patterns generated by any of the above compositional devices can be used to define waveform data. Even the density or mean pitch graphs of a piece can be used as waveform or envelope patterns. How all this all works is highly dependent on the construction of the piece, but the end result is a direct structural link between timbre and melody. This connection is an even higher example of a fractal structure because the self-similarity now includes information from completely different realms of the composition. There is something very satisfying about writing a piece of music that has this kind of internal consistency. The conceptual implications of such a situation are great and are well beyond the scope of this overview.

Paint Programs: An Artists Evaluation and Wishes

Trip Denton

As a professional user of several of the lower cost, microcomputer-based paint programs, I have found features in each that I like and dislike. This paper will discuss primarily two of the more wider used paint programs, PC Paint (by Mouse Systems, Santa Clara, CA), and MacPaint (Apple Computer, Cupertino, CA). Some examples of work using each of these programs is then presented.

Mr. Denton currently teaches at Moore College of Art, Philadelphia College of Art, and Hussian School of Art. He is currently working for Forte' Interactive Design on an animation project for the IBM compatible computers.

Introduction

Paint programs are perhaps the most common artists tool for small computer systems. A paint program on a computer gives the artist a freedom not found with conventional painting. One can "undo" a brush stroke, change the shape of the brush, and make copies of the piece at various stages in its development so that a different course may be taken to its final composition. Problems with paint programs lie mostly in two areas: the artist must learn a new set of tools and techniques and the realization of the work is generally not on canvas but rather a picture tube, printer output, or photograph.

The Joy of Paint Programs

One of the features of paint programs which I use in my work is the abilty to design your own patterns and then fill and/or paint with them. Both MacPaint and PC Paint have this capbility. MacPaint's is more flexible. PC Paint allows color painting which the Macintosh does not but pattern design is limited to two colors. MacPaints lack of color though is made up in its sophisticated drawing features like outline and expansion and contraction of designs.

Show Me the Whole Picture...

When I used to use a Koala Pad (a low cost touch tablet) and my Commodore 64, I got used to the fact that what I saw on the screen was my entire design. Moving onwards and upwards to more expensive computers and programs, I was disappointed to learn that much of my design was off the screen somewhere. It is frustrating not to be able to look at the picture as a whole. Almost all of the sophisticated (?) paint programs for small computers show only part of the picture. This makes it difficult to do something as simple as drawing a boarder.

One Tool at a Time Please....

One distracting feature of most of the paint programs is that the tools are always displayed on the screen. When I paint, I want to select the tool I need and do my job on an uncuttered screen and I don't want to see the rest. Having all the other tools on the screen is a distraction which is not necessary. Again referring to the low cost Koala Pad, you pressed a key which brought up a screen full of tools for selection. Once selected, you returned to your work area which filled the entire screen.

CH2218-6/85/0000/0101$01.00 © 1985 IEEE

Show Me Some Tricks...

MacPaint generally had more sophisticated drawing techniques but one feature I found useful in my particular work, I found only on PC Paint. This was the ability to draw a line and then "curve" it by pulling on a section of it. Apart from this feature, I found PC Paint lacked some of the more creative tools such as mirrors, expansion and contraction, outlining, and flexible pattern design. The expansion and contraction was very useful for drawing oversize and reducing to fit the design. I liked the differnt fonts of MacPaint. I would like to see tilting added to these programs (though an additional program, Clip Art Effects, provides this for the Macintosh). I could also use a larger box for pattern design.

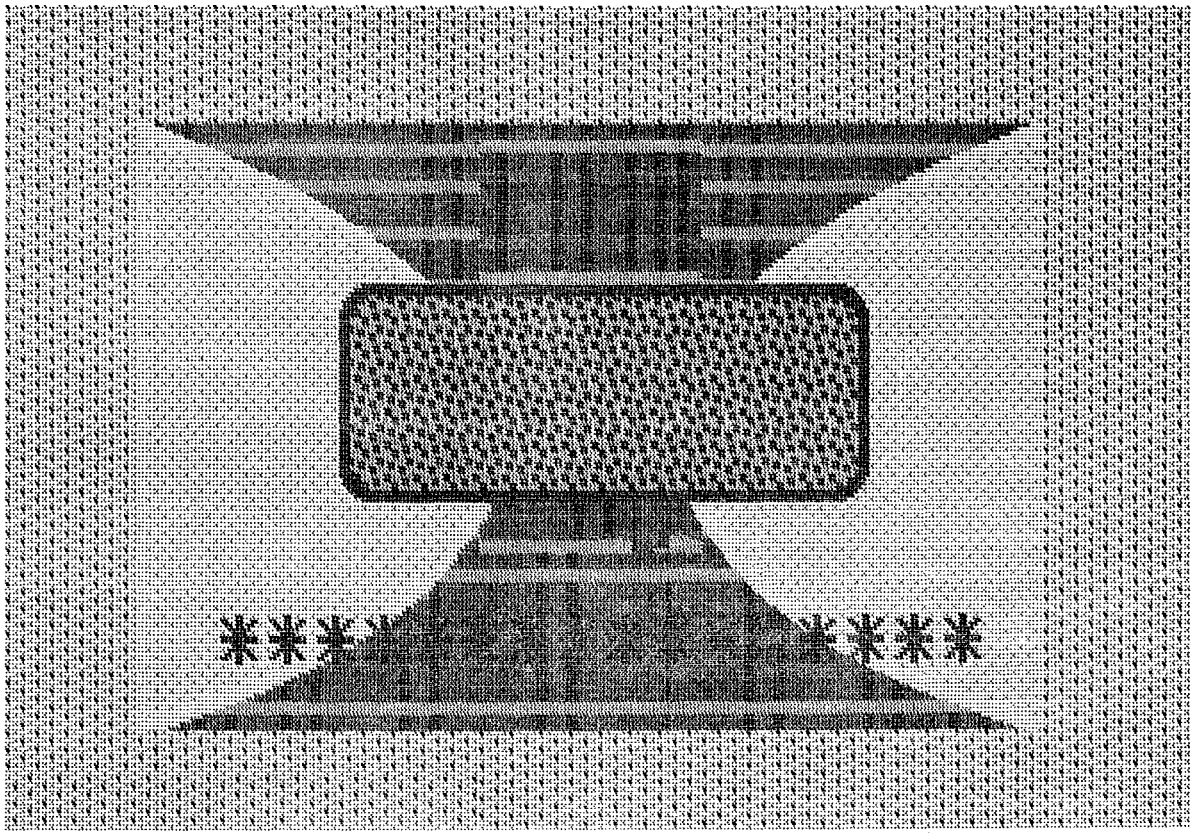
No More Bugs!

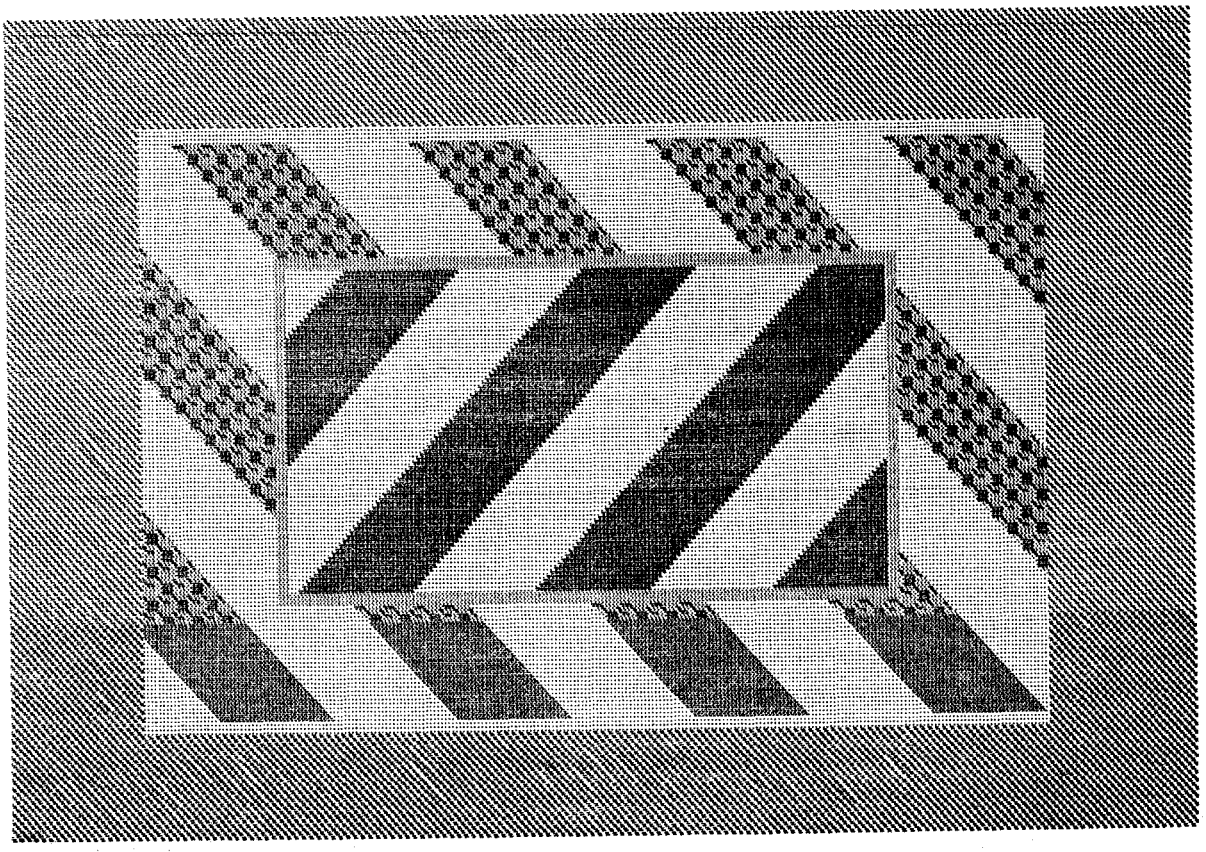
Bugs were found in both programs. MacPaint's ocurred when a picture was moved off the screen in Show Page mode. It returns destroyed on untouched depending on which side you move it to. A more frustrating bug was found in PC Paint whereby designs in the upper left corner could not be magnified. It must be moved to another location and then magnified.

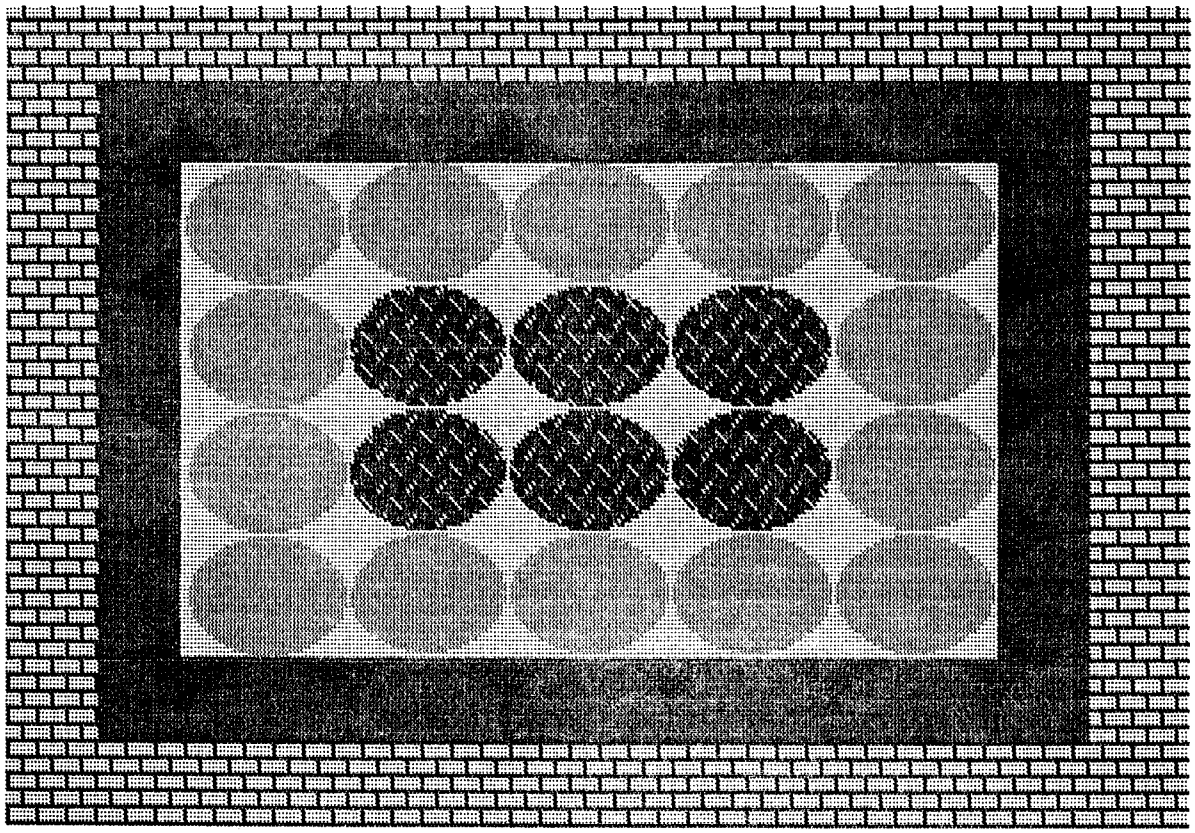
Video

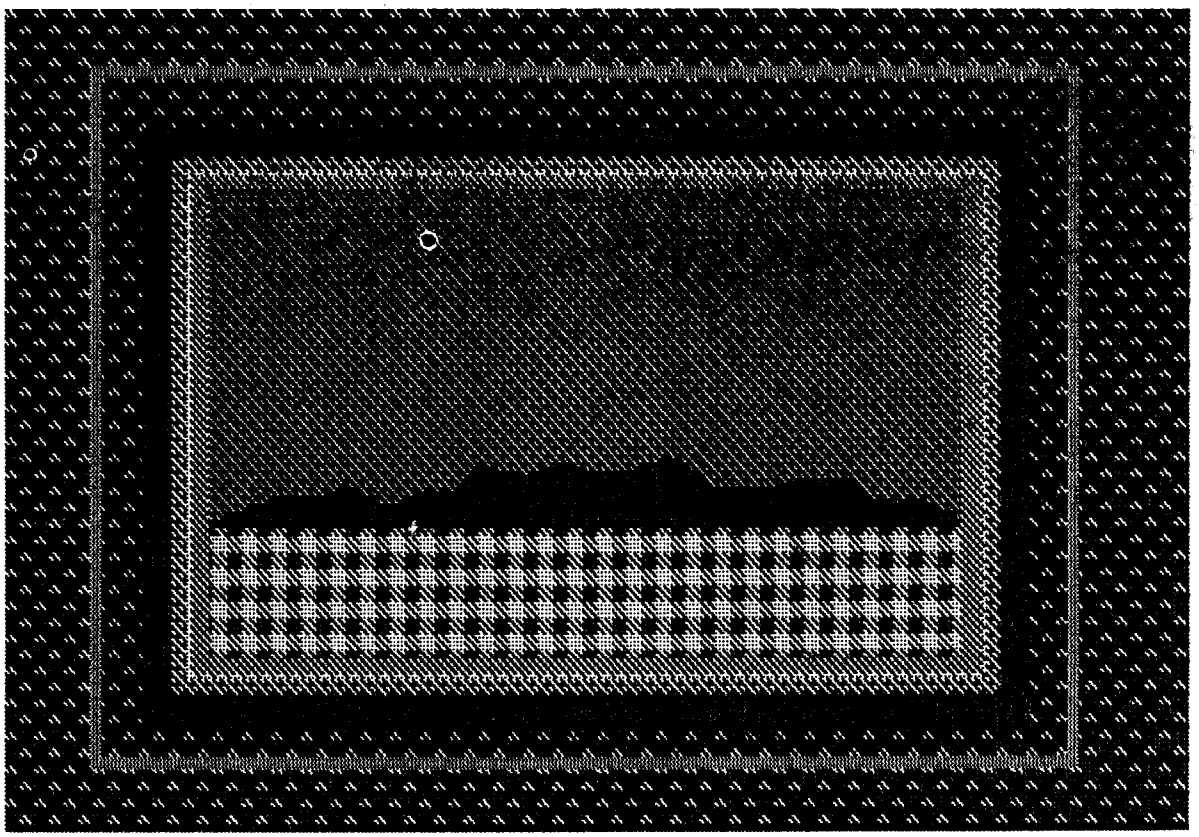
The Inovion (by Inovion, Layton, Utah) user interface is more difficult to use than the previously described paint programs but it allows video capture and re-painting. It has more colors but no pattern fill, enlarge and contract, and other nice tools. It does have highly sophisticated color changing features. You can modify one color, a whole area, or one pixel.

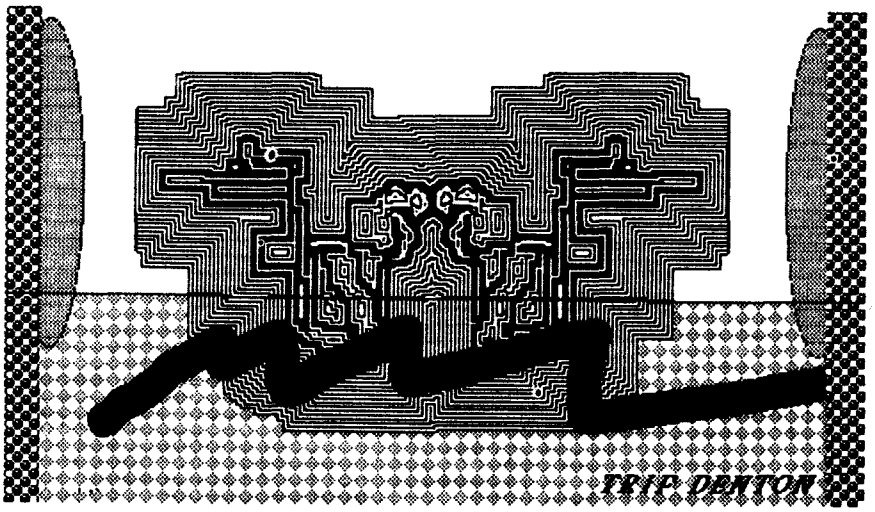
Some samples of my work using PC Paint and MacPaint follow. The first two pages are done with PC Paint followed by some MacPaint examples.

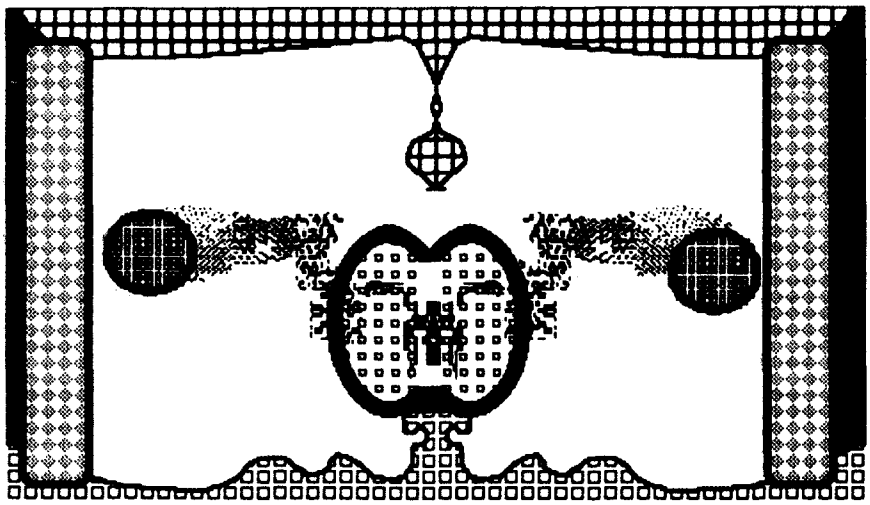









TRIP DENTON



CONCEPTUAL GUIDELINES FOR CHOOSING A GRAPHIC WORKSTATION

THOMAS PORETT

DIRECTOR OF COMPUTER STUDIES
PHILADELPHIA COLLEGE OF ART

ABSTRACT

This paper discusses the problems confronting the artist or designer when faced with the question of incorporating digital imaging systems in the studio. It begins with a conceptual analysis of the strengths inherent in computer graphic systems, and implications of how such systems can influence the creative process. The paper concentrates on micro-computer based workstations, configured by the user or packaged by various vendors. Guidelines for choosing a system the offers optimal integration is stressed.

COMPUTER AS CATALYST

The computer configured as an artist's workstation is unquestionably a powerful tool, and catalyst for visual thinking. Within the confines of current technology it is a flexible, multipurpose image making device, whose capabilities range from relatively simple flatwork, to animated three dimensional sequences. Although it is viewed most often as a tool, it is important to recognize that as a system, it can affect significant influence in the conceptual creative process. This is possible primarily because the adaptive nature of the computer allows the 'personality' of the tool to be molded or radically altered.

Problem solving is altered because visual material is not fixed, but exists as changeable numeric data. This data can be acted upon by both hardware and software, and until a piece is fixed on paper or film, it is subject to change, yielding a flexibility that is unparalleled in any other medium. This malleable state offers the artist an opportunity to examine the creative act as a multidimensional process, in which decisions can be interrelated in such a way that more solutions emerge than would if a conventional approach were chosen.

A common example of this is seen in the operation of a typical electronic 'paint' program. Such a program seeks to emulate the tools common to the conventional drawing studio, with such functions as 'brushes', palettes, and various line drawing modes. However, a major addition to these common tools is the editing function that allows a picture to be cut and pasted, much like text is manipulated in a typical word processor. This offers significant control and flexibility to the artist, but represents only part of the real potential available.

Far more fascinating possibilities begin to emerge as both tool makers and artists better understand this new creative environment. The

CH2218-6/85/0000/0107$01.00

critical concept that distinguishes digital systems from analog and material processes, is the concept of integration. The potential of communicating elements of one process with those of another, is not easily possible through conventional means. This holistic approach to problem solving has gained great sophistication in development of integrated office systems. The ability to share data amongst disparate tasks represents a powerful implementation of a dynamic desktop business tool.

The future of graphic systems will follow this lead, as it utilizes the power of the computer in a way that synergistically opens creative avenues to serve the demands of artists working in technological media. This synergy occurs by what might be called "process mediation", through which a solutions to problems are influenced by the openness and flexibility of the tool used. The artist's conception is enhanced through the freedom a system can offer in allowing an idea to be realized through multiple possibilities.

POTENTIALS FOR INTEGRATION

For example, a system designed to create three dimensional objects should integrate features of an interactive 'paint' program, allowing a virtual solid object to be shaped, then painted and further altered either dimensionally or graphically. A simpler example would integrate a 'paint' program along with the vector drawing capabilities of a computer aided drafting and design (CADD), system. This would permit two dimensional objects, stored in a 'library' disk, to be modeled by the paint system. Such capabilities are unique to digital systems. They engage a baseline of conceptualization previously unavailable to the artist, extending creative alternatives by inviting integrative problem solving.

Clearly, many aspects of such a system already exist in some expensive installations, but even those are only partially integrated, and out of reach of the individual artist or designer. However, there are encouraging signs that micro computer based workstations are beginning to emerge that will allow the concept of integration to be embodied as a working system. Most of these systems are designed for two dimensional image creation, but in some cases have a data structures that can be mapped onto three dimensional shapes. The artist is faced with the somewhat perplexing task of accommodating their immediate needs, while choosing a graphics system that offers a degree extensibility.

As a consequence, there are added burdens placed upon the artist, as he/she must acquire both depth in a specialized visual discipline, and a general understanding of digital systems. Conventional art education has functioned well in providing the requisite visual skills, but has little experience in the latter. Accordingly, it is currently in the hands of the individual to attempt to acquire these skills independently, while the mainstream educational institutions come to grips with these issues.

CONCEPTUAL FRAMEWORK

For the individual that has little experience with computers some steps can help make this process somewhat less traumatic, and yield additional skills in the interim. There is probably no better way to begin to sense the power of the computer that to experience the use of a word processor. Most typical programs will give the individual an understanding of those essential ingredients, such as cut and paste, that make

word processors, such powerful tools that enhance written expression. Through the experience of using the editor of a word processor, they will better understand the intrinsic rules that shape events within the computer's memory, and perceive the meaning of what is termed an 'environment', created by a program be it a word processor or graphic system.

Additionally, the experience of storing data to disk, printing to paper, and re-editing this material are elements that have direct counterparts in digital imaging processes. The important issue for an inexperienced user, is to establish an conceptual framework for utilizing the computer in a creative endeavor.

GRAPHICS LITERACY

Once such familiarity is established, the next phase should be directed toward gaining a graphics 'literacy', beginning with the use of a raster oriented 'paint' program. Many features found in advanced graphics systems have been incorporated in micro-computer graphic programs, and a good deal can be learned by using one of simple to use, but elegant ones available. It will begin to give one a sense of those features that an optimal system should possess. Further steps should include some level of familiarization with aspects of a vector drawing CADD process, and/or a three dimensional design package. Both raster and vector oriented approaches will continue to flourish, as each have distinct strengths for particular applications.

Once the familiarization process has been carried out, one can begin to approach the final stage of choosing whether to buy a prepackaged graphic workstation, or configure one from separate subsystems. The first option will very likely cost more, but will provide a usable system without requiring much more specialized knowledge of digital processes. Should the second option be chosen, the individual must be rather familiar with the fundamentals of hardware and software, in order that they are able to differentiate capabilities among the variety of devices available. The alternative is of course to purchase an off the shelf system from a vendor, and accept the inherent capabilities or limitations that accompany such a system. In some instances this may be the best solution particularly if the vendor is committed to user support and system upgrading.

While such an alternative is acceptable for some, it subjects the user to limitations determined by the hardware and software supported by that vendor. Further, these systems tend to be rather expensive, and may be out of reach of the individual or small design firm. There are currently some systems that are beginning to emerge that are priced in the ten thousand dollar and under market, and provide many of the essential features of a usable system. The following information should help those who are interested in purchasing an existing system, or configuring a system of their own.

GUIDELINES

Primary among the many factors that face the artist is the question of upgradability of a system. Currently a relatively low cost system provides at least 16 colors and a resolution of 512 x 480 lines. It should be able to expand its color palette to at least 256 simultaneous colors, and possible a range of 4096 or greater total number of colors.

It must have an analog RGB output that will allow this range of colors, and permit the system to be interfaced with a film recorder. It should be expandable to a higher resolution if needed.

At the very low end of expandable system configurations is the Apple IIe equipped with extended memory (128k). Although it does not attain the desired resolution mentioned above, it is capable to of 560 x 192 lines in black and white, and 140 x 192 with 16 discrete colors, (mixes of resolution exist between these extremes).

The strongest argument in favor of the Apple is its cost, upgradability, and support by numerous vendors, including very acceptable paint functions, and sophisticated CADD environments. The availability of seven I/O slots allows several functions to coexist and often be integrated. For example, support of a high resolution RGB monitor and film recorder that can photograph any screen image yielding a 35mm slide, takes just one slot. A video digitizer may be added that supports the double high resolution (560h x 192v) capabilities will take another slot. Another slot can be used to interface a digitizing device such as a tablet, mouse or lightpen. A CADD system can be added that interfaces through the joystick port instead of an I/O slot. An additional slot can be dedicated to supporting a color ink jet printer or conventional dot matrix device. Finally, a slot can be dedicated to hosting a plotter, resulting in a rather broadly based graphics workstation, providing great flexibility, lacking high resolution for raster graphics.

UPGRADING GRAPHICS

Even this limitation may be upgraded by introducing a graphics processor either as an internal board, or as an interface to an external system. Typically such an upgrading will yield a resolution of 512 x 512 with at least 16 colors, and is usually capable of accessing a palette of several thousand colors. Some of the upgrades have special features that allow the output to be used directly as a video source, a factor often not paid much attention to until the occasion arises. As many individuals have discovered, the rich RGB signal that appears on a monitor does not necessarily transfer to a video environment.

VIDEO GRAPHICS

To make this transition, the processor must either be designed to provide the requisite NTSC signal, or be processed by an RGB to composite video device. Should the signal be destined for use in the demanding environment of broadcast video, it must comply with a set of exacting standards that many composite sources fail to pass. It is wise to do some thorough investigation before committing one's resources to a system if broadcast video usage is foreseen.

APPLE LIMITATIONS

There are limiting factors with the Apple II based system that do present some problems. There is no single source for a system configured as an integrated package, leaving a good deal of work to the user. Also, the 8 bit Apple processor is relatively slow, particularly with some extended resolution graphics processors. This can be mitigated somewhat by the use of a high speed accessory processor interface, but it will suffer in comparison to some newer systems. More importantly, there has been a slowdown of new software and hardware intended for use with the Apple. Nevertheless, it still represents a viable and cost effective system particularly when it is assumed that it will eventually act as a host to a more advanced graphic processor. Additionally, most devices such as printers, plotters, film recorder and monitors can be used with a more advanced configuration at some time in the future.

IBM PC BASED SYSTEMS

The next step up from the Apple system is one that is based upon using the IBM PC architecture. Although not primarily a graphics machine in its own right, the PC has been given a good deal of support by

outside vendors. In order to have any graphics capability at all, one must choose to support a graphics processor interface, and the market offers a plentiful number of options from which to select. The field actually narrows when it is understood that most graphics cards support a limited graphic capability, and only a few are dedicated to doing more than non-demanding business graphics.

Those cards intended for more demanding graphics devices are a very attractive alternative, and can be configured into a powerful system with the proper choices of software and hardware. The cost of such a system will be about twice that of a comparable Apple based system, but will offer greater speed and upgradability in the tradeoff. This market is being actively developed by OEM's, software firms and system vendors, offering the potential user a selection of high performance and expandable systems. The costs are modest to steep, ranging from a low of about four thousand to a high of about thirty thousand dollars. Interestingly enough, some of external graphics processors can be interfaced to either an Apple or IBM, providing a transportability of operation should an individual change their host CPU.

SOFTWARE STANDARDS

One major step that has developed along with the advances in hardware is movement toward utilization of graphic software standards. Although such standards are by no means universally accepted by the industry, some software systems such as Dr. Halo and GKS have enabled a common link of command structures to function effectively with several graphic boards for the PC. This movement may begin to lead to truly integrated micro-computer graphic systems that will rival large scale installations, and may even exceed the degree of integration, particularly with bit mapped color raster graphics.

NEEDS ASSESSMENT

An assessment of need is required to justify both financial commitment and perhaps more importantly, the time needed to make this investment worthwhile. This point should not be underestimated, as it will take a considerable length of time before one is able to effectively use even some relatively straight forward graphic programs. Further, the artist or designer must carefully weigh an actual need for making the transition to computer graphics, as it must be realized that many visual problems are as easily solved in conventional ways, and may actually be hindered by the presence of the machine. The difficulty stems as much from aesthetic issues as operational ones.

The presence of glowing electrons can seemingly enhance nearly any mundane image, and it requires a healthy degree of disciplined distance to accurately judge a work in progress. In some ways it is similar to the satisfaction experienced by the novice photographer in watching a print develop in the darkroom. One is seduced by the operation of process, and in doing so loses sight of the primary aesthetic problem at hand.

CAVEATS

Another issue that must be weighed carefully is what might be termed a balance between freedoms and tyrannies. As has been mentioned in the beginning of this paper, an undreamed of flexibility and freedom to create is offered with these new tools. The graphic workstation can be thought of as a self contained studio, that invites multidimensional conceptual problem solving. It can provide

a rich creative environment to those who adapt their creative skills to the demands of these new tools. However, there are some important caveats that must be heeded.

Each system however well conceived has strict limitations that inhibit performance in matters ranging from resolution, to style of hardcopy supported, and the effectiveness of its software. These factors can negatively influence the artist because they create some rather absolute boundaries beyond which one cannot wander. Although any medium has limitations, digital systems, particularly those that are closed and bound to the whims of one vendor, can be very frustrating. There are no easy solutions, except care in choosing a system that permits open expansion of hardware and software. It may not entirely alleviate the tyranny of limitations, but it can help avoid unnecessary aggravation.

The tyranny of unreasonable expectations is by far the most important factor that can diminish or even negate the creative potentials of digital processes. The belief that by utilizing a wondrous graphic workstation one's creative output will grow in quantum leaps is clearly nonsense.

Except for those commercial graphic concerns that mass produce business graphics of dreary pie charts, huge gains in productivity cannot be assumed. In a worst case scenario, one can envision the hapless artist of the electronic age chained to their monitor by unrealistic demands to produce more because computers, " can do things faster."

Such expectations are clearly unrealistic no matter how well the machine is configured because intelligent thought, the heart of any successful image is ultimately limited by the mind. The qualitative aspects of an image rely on careful thought, and there are no enhancements to any graphics hardware or software that can replace this process. However, a well designed system can be a valuable aid in the visual thinking process, and can help provide a qualitatively better image.

The interaction between this flexible tool and the artist can lead to multidimensional thinking in which solutions emerge that were impossible to conceive of previously. The ideational process is profoundly altered, implying that the term 'tool' is too limited when attempting judge the influence that computer graphics systems will have in the future of visual thinking. Those artists that are prepared to create in these new environments will shape and define the value of computers to the arts. The question remains open and tantalizing.

©Thomas Porett

DOUBLE HIRES PRINTOUT FROM AN APPLE IIe COMPUTER

©Thomas Porett

DOT MATRIX PRINT FROM A MACINTOSH COMPUTER

©Thomas Porett

INK JET PRINT FROM THE OUTPUT OF A SYMTEC PGS III
GRAPHIC PROCESSOR WITH 512 X 460 RESOLUTION
(ORIGINAL COLOR PRINT MEASURES 11 X 11 INCHES)

ALGORITHMIC COMPOSITION

Richard Kram

Temple University
College of Music
Philadelphia, Pa. 19122

ABSTRACT

"Musicians make good programmers," I'm told time and time again by computer professionals. I contend that there is a fundamental relationship between the two disciplines and that an in-depth knowledge of one of these fields can be used to reinforce the creative aspects of the other. I will elaborate on this idea and present a compositional methodology which I term "algorithmic composition". Its intent is to start formalizing a new field of musical study which applies the thought processes and programming techniques of the computer scientist to composition, hopefully resulting in fresh musical forms and innovative pitch and rhythmic combinations.

INTRODUCTION - COMPOSER VS PROGRAMMER

Many of the basic formalities of design found in programming have been used by musicians for hundreds of years. A short (albeit oversimplified) comparision of a typical piece of music and a computer program should begin to clarify this assertion.

First, both music and computer programs are normally written in a language which represents an abstraction for a linear process which progresses with time. A composer must always be cognizant of where any particular note stands in relation to its neighbors. One misplaced note could ruin a phrase and possibly destroy an effect which has been building from the beginning of the piece. A misplaced computer instruction can in turn render a block of code useless. By the same analogy, a slight change in a composition or a program can often solidify an entire work.

Both the composer and programmer must become adept at integrating a myriad of events so that the result produces an harmonious interaction. Compositions and programs are both composed of indivisible units (notes and rhythms vs. opcodes and operands) which are combined to form various patterns conveying a minimal amount of meaning (motives vs. instructions). These patterns can in turn be developed into more meaningful units (phrases vs. statements). This growth process continues into larger sections (periods vs. procedures) until a self-contained, unified whole emerges. There are even relationships concerning the simultaneity of events (harmony and counterpoint vs. multiprocessing and interprocess synchronization).

Both composers and programmers must deal with highly subjective measurements which can only be judged in relationship to other works in a particular style. When does a piece of music start to become trite, redundant or boring? When does a computer program start to become inefficient, unstructured or too long to be maintainable? Every composer has a particular style of composition and manuscript. Likewise, no two programmers code in the same manner.

Both composers and programmers are meticulous about finding and correcting errors. Composers have been in the "debugging" business since the birth of musical notation. How many composers can boast of writing a substantial score of more than twenty pages without finding numerous errors in the manuscript? Many of these errors are never caught, just as many computer bugs remain undetected. In short, composers are subject to the same fits of frustration and moments of inspired creativity known all too well to computer programmers.

ALGORITHMS AND LANGUAGES

One of the major differences between writing music and code is that programming is by necessity more functionally goal oriented. Once a major goal has been set, the programmer must formulate a specific series of instructions which when executed will produce the desired result (an algorithm). Large tasks are

broken into smaller ones until the job can be described in a series of descending levels, each one dealing with more and more of the complexities of the project. With music, it is desirable to get a grasp of the overall form of a piece and its various sections before churning out notes, however, music is often composed more from the bottom up than from the top down.

The technique of composing algorithmically is not without precedents. Canons and fugues have always followed a strict set of rules. In the last thirty years, a substantial body of music has been written which can be described in algorithmic terms. The critical point to be made here is that a computer is not necessary to compose algorithmically, but a high-level algorithmic computer music language would greatly facilitate the endeavor.

Many computer music languages exist for composing traditional music, but few make an attempt to incorporate elements used to manipulate abstract data structures like records, linked lists, trees and graphs. Even fewer contain elements which allow the composer to experiment with music in a multiprocessing environment. At present, composers must be willing to learn a high-level computer language capable of handling these data structures and their associated algorithms if music is to be created which relies on them. AML (Algorithmic Music Language) is a computer music language I am currently formulating which is intended to help fill this gap. AML will provide data structures geared for musical storage upon which traditional computer operations can be performed.

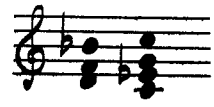
Algorithmic composition is a broad field of study. It embodies more than "composition through a set of rules" as its name might suggest. Any element of computer science is fair game for musical use. In this regard, the microcomputer can be just as valuable a tool as a mainframe. Computer science can be used as a source of ideas. Set theory has been invaluable in the composition of serial music. There is every reason to believe that elements of programming can make similar contributions to algorithmic composition.

EXAMPLE APPLICATION - TREES

During the last fifty years, composers have investigated numerous methods of deriving thematic material through pitch set permutation. An enormous body of music is based on the manipulation of tone rows using transposition, retrograde and inversion as the primary derivations. Computer science provides numerous data structures which can be used to generate new thematic possibilities when applied to pitch sets. As an example of algorithmic composition, a set of variations will be generated by applying different traversals to a binary tree containing a short motive (and to a series of trees derived from the original tree).

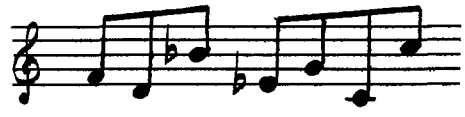
Lets begin by selecting a rather trivial musical example from which a variety of melodic patterns can be generated:

Example 1



This is nothing more than two alternating triads a step apart. What can be done with so simple a fragment? A linear representation must first be created so the tones can be manipulated in sequence:

Example 2



The next step in this exercise is to store the motive in a binary tree. The binary tree is a linked data structure consisting of a root element (parent node) from which at most two descending nodes (right and left children) can be attached. Each of these child nodes can in turn act as a parent to further subtrees. The tree can thus be described recursively (in terms of itself).

Each node will normally contain a key. Our example uses pitches as keys. The key of the left child is normally less than the parent's key and the key of the right child is greater then that of the parent. Rhythms might also be used as keys, but this example will confine itself to pitch manipulation. The following algorithm can be used to build the tree:

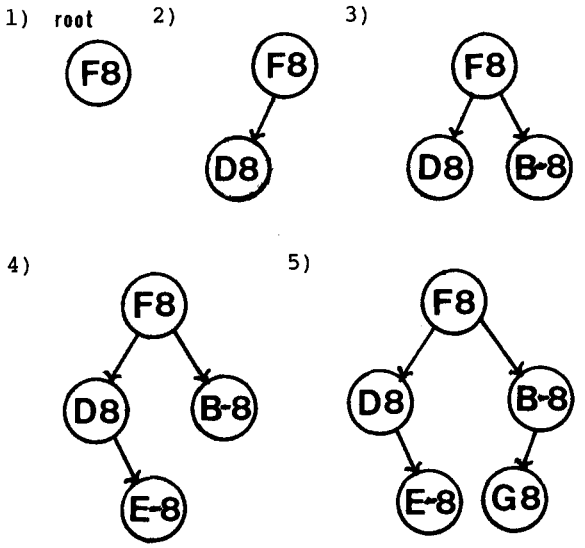
Algorithm 1

1) Get the first note (F8) and install it as the root of the binary tree.
2) While there are still more notes to process do the following:
   a) Get the next note in the motive.
   b) Starting at the root, if the pitch of the note is lower than that of the current key being examined, traverse down left else traverse right. (This assumes no repetition of keys.)

c) Continue traversing the tree until a free space has been found and the note has been installed as a terminal (leaf) node.

The tree for the above motive should be be built in the following seven steps:

Figure 1



This tree has three levels counting the root, each of which is perfectly full or balanced. If you try to generate trees from random themes, you will find that a perfectly balanced tree rarely occurs. Try making a binary tree from a major scale. A totally unbalanced tree should result containing seven levels. A relatively balanced tree should be used to generate a variety of interesting patterns. The optimum configuration has no leaf node more than two levels away from any other leaf node.

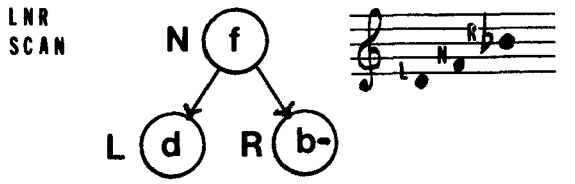
There are numerous limitations inherent in our tree representation. No means is provided for storing note repetitions. The tree also makes no attempt to retain the original note ordering. Extra node fields and links (threads) can be used for this purpose, although other data structures such as arrays and linked lists are better suited for linear storage. There is no limit to the complexity which can be imposed upon our model. This highlights an important art of programming -- knowing what data structures are best suited to manipulate a specific set of data. These relationships are well defined for numeric and string (text) data. Composers must experiment to establish the optimal storage methods for handling musical data.

Now that the original motive is stored in the binary tree, tree traversals can be used to generate different orderings of the original tones. There are three major types of tree traversals (scans): inorder, preorder, and postorder. These names refer to where the parent node examination of a sub-tree falls in the traversal.

INORDER TRAVERSAL (LNR and RNL)

In the inorder traversals, either the left or right sub-tree of the parent node is scanned, followed by the parent node, followed by the remaining sub-tree. This procedure continues recursively until the entire tree has been traversed. It is easy to visualize this process using a tree containing only three nodes:

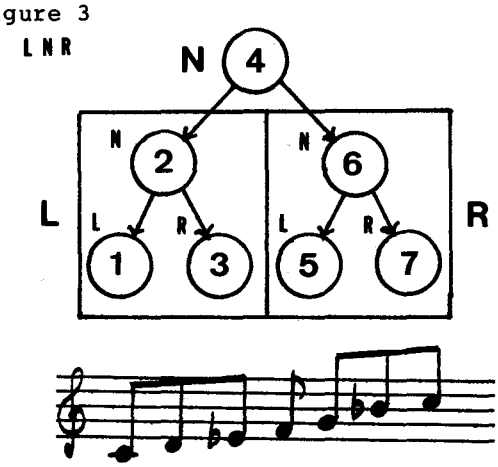
Figure 2



The LNR (left-node-right) traversal simply visits the left node followed by the parent node followed by the right node and produces the sequence D8, F8, B-8. ('-' indicates flat and '#' sharp). Now lets apply the LNR scan to the entire tree. The leftmost sub-tree must be processed in LNR order followed by the next leftmost subtree (etc.) followed by the root node, followed by the sub-trees to the right of the root, which also must be

processed in LNR order. The sub-trees will be boxed in the following examples to help illustrate this procedure.

Figure 3

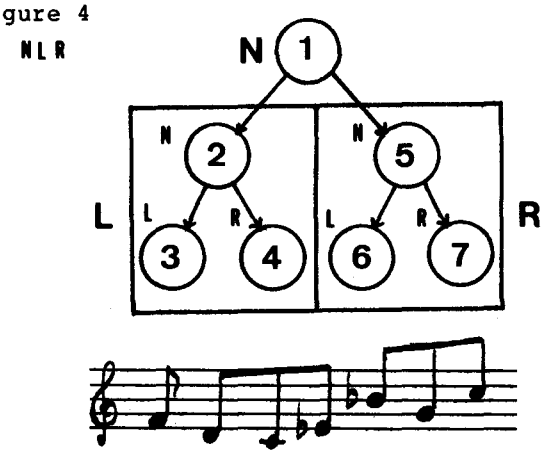


An inorder traversal will produce the low-high or high-low ordering of the keys. These traversals can be used when the most linear permutations of a theme stored in a tree are desired.

PREORDER TRAVERSAL (NLR and NRL)

All of the remaining traversals follow the same recursive scheme, differing only in the order of node visitation. The pre-order traversals visit the root node followed by the left and right (or right and left) sub-trees. Parent nodes are thus examined first, followed by children, if they exist.

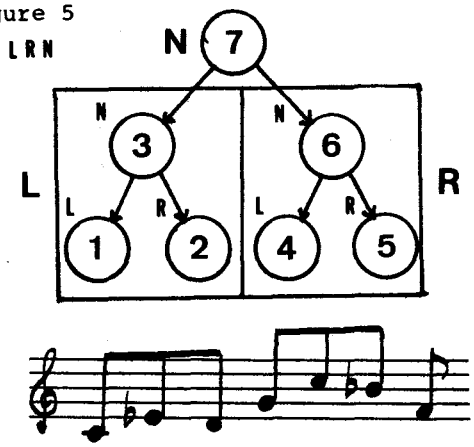
Figure 4



POSTORDER TRAVERSAL (LRN and RLN)

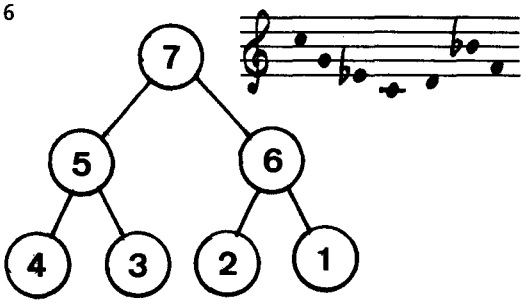
Postorder traversals visit parent nodes last in the scanning process. Either the right or left sub-tree is scanned first followed by the remaining sub-tree followed by the parent.

Figure 5



There are other nonstandard methods of traversing a tree which may produce some interesting patterns. Try winding from the terminal nodes in an alternating left-right-left (etc.) fashion up to the tree's root. This would require special threads to program, but is easier to visualize than the three traversals presented above because it is a nonrecursive linear process on paper.

Figure 6

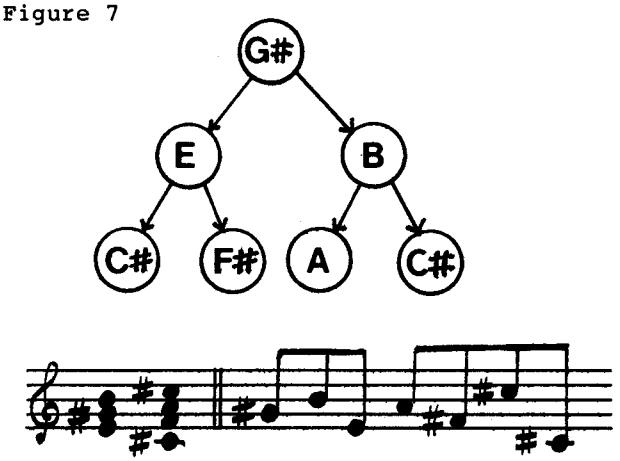


Try inventing some different scans. You may well create some interesting traversals never used by computer programmers due to the nature of the information normally stored in trees. This is one area where musicians may well add to algorithmic design - using data structures in a manner never envisioned when initially developed.

All of the motives thus far produced will have a similar modal sound because none of the permutations have been transposed, and they use only six diatonic tones. A separate tree could be created storing a transposition scheme which in turn could be traversed to create harmonic variations. Another possibility, if more tonal complexity is desired, is to incorporate the remaining six tones of the chromatic scale into our thematic model. It turns out that they can be stored in a similar tree derived from

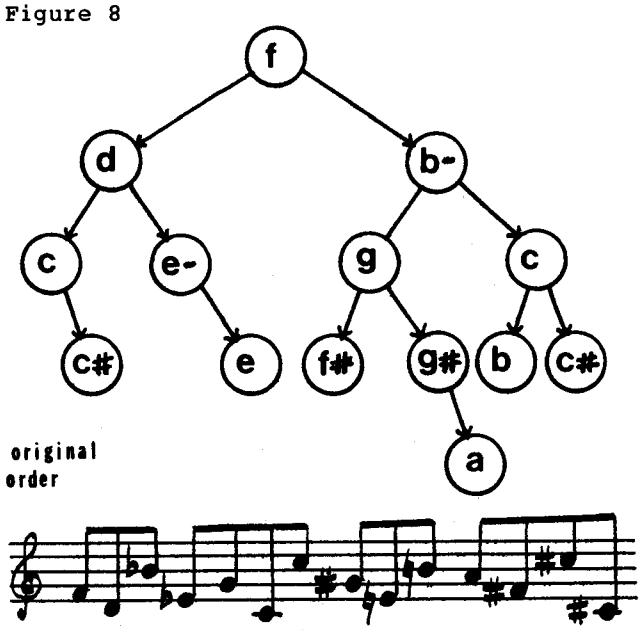
the same stepwise triadic alternation.

Figure 7



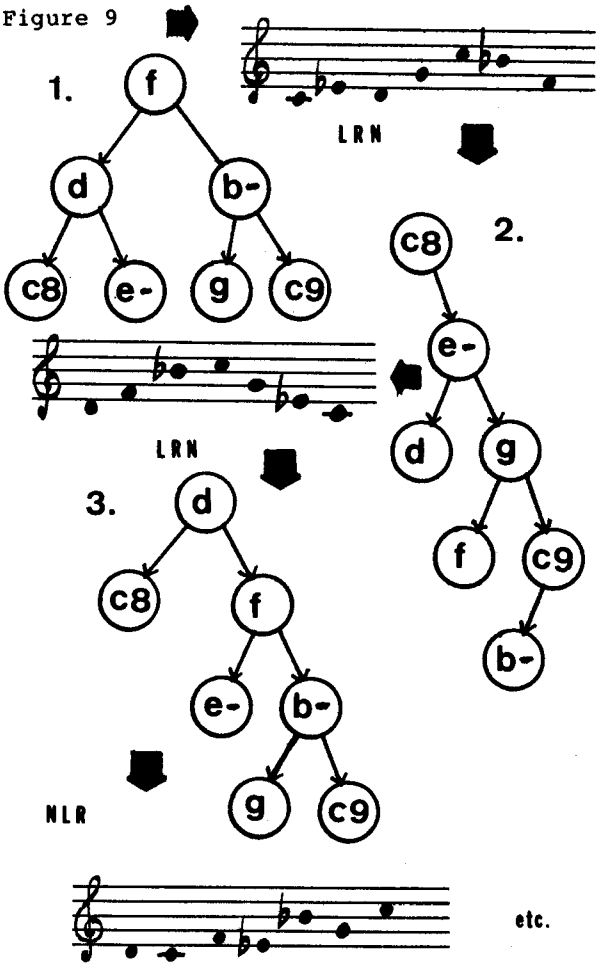
Patterns created through the traversal of this tree may be alternated or interweaved with those produced from the original motive or the dissonance level may be further increased by building a tree containing both of the motives. The resultant traversals contain a strange combination of chromaticism and modality.

Figure 8



Even more patterns can be generated by channeling the output from one traversal into a new tree and traversing that. This process can be repeated until the pattern starts to repeat. Clearly the possibilities are endless.

Figure 9



OVERVIEW

All of the motivic variations created through tree manipulation are highly unified through a well defined series of operations. The composer is left to use this thematic material as the need sees fit. Entire pieces can be generated through setting up some complex scheme of traversal interdependancies or maybe only a few of the motives derived from the original tree need be used to compose a piece in a more traditional manner.

Trees provide only one means of storing and manipulating pitch and rhythmic material. Arrays, linked lists, graphs, stacks, queues, random access files and a host of other data structures can be used to generate interesting patterns along with other techniques such as looping and conditional branching which can be used to introduce numerous formal relationships into a piece.

This paper has focused on pitch generation, but algorithmic composition can also address large scale structural

considerations as well. A series of algorithms can be used to both select musical material and to form it into a unified piece. The goal of algorithmic composition, however, is not to try and get a computer to compose "by itself". I leave this area of composition to those interested in the area of artificial intelligence. Algorithmic composition uses computer science for ideas and inspiration. The computer itself is incidental. A pencil and piece of paper can always be used to simulate any algorithm or data structure which may be applied to music.

I hope that this paper has inspired some musicians to explore the world of computer science as a source for alternatives to traditional composition. Languages like AML should make it easier for the novice to experiment in algorithmic composition by making the intricacies of programming transparent to the user and thereby allowing composers to concentrate on creative matters. I am not suggesting all composers become computer gurus, nor am I encouraging computer programmers to run out and become composers. I am hoping that composers will use computer programming and its related skills to widen their conception of musical composition.

REFERENCES

[1] D. Knuth, "The Art of Computer Programming" - Vol. I - "Fundamental Algorithms" Addison-Wesley, 1973.

[2] R.Sedgewick, "Algorithms" Addison-Wesley, 1983.

[3] T. Standish, "Data Structure Techniques" Addison-Wesley, 1980.

[4] N. Wirth, "Algorithms + Data Structures = Programs" Prentice-Hall,Inc., 1976.

AESTHETICS AND COMPUTER GRAPHICS

Walter Wright
School of the Arts
Virginia Commonwealth University
325 N Harrison St
Richmond, VA 23284-0001

THE ARTISTS' WORKSTATION

"You can't teach an artist to be a computer, so we taught our... computer to be an artist." a recent magazine advertisement tells us. This is how not to design an artists' work station.

Let's consider the advertisers point of view. An artist inside each computer. What a sales gimmick, a truly expert system, instant art, a replacement for those irresponsible and unpredictable types who call themselves artists and designers. Incidentally, the advertiser has solved a problem that has plagued philosophers for centuries. He knows what Art is !

From the artists' point of view most workstations are disasters. There is something very wrong with the notion that a standard "paintbox" can turn anyone into an artist or designer. Do artists really use standard paintboxes ? None that I know. Each artist's paintbox is very personal and highly individual. They choose paints, brushes, papers, and other tools to suit themselves. My paintbox and my choice of materials is unlike anyone elses. On the other hand most artists would love to use a computer graphics system, preferably after hours. The challenge of a new medium, the opportunity to experiment, the chance to create images never before possible--- what artist wouldn't be turned on ?

As an artist the workstation is a miserable failure. It promises much but delivers little. There is no real challenge, no means of experimenting, and the images are copies of "the same old thing". What happened ? In order to find out let's examine some of the basic concepts of art and design.

ART AND INFORMATION

"The map is not the territory". This quote contains an important clue. Who are the mapmakers ? Who creates the symbol systems necessary to describe new and uncharted territory ? Artists and designers are mapmakers, along with other creative individuals they chart out new applications for science and technology. Computer graphics is new territory, it should be explored with an open mind, without any preconceived notions. However the current crop of artists' workstations indicate that just the reverse is happenning. We are speeding into the future with foot to the floor and eyes glued to the rear view mirror, to paraphrase Marshal McLuhan.

An example of this self-destructive urge is the brush algorithm. I used to refer to the computer as an electronic paintbrush, it seemed like a good metaphor at the time. Unfortunately the electronic paintbrush metaphor is being interpreted literally. Making a computer into an expensive paintbrush is counter productive. It wastes everyones time and energy on something the computer was not designed to do, and ignores the things the computer was designed to do. I once taught a course in systems and painting. The brush, paint and paper systems developed were very complex. I can't imagine simulating any of these systems on anything less than a giant mainframe. So why bother ?

What does a computer do well ? Posterization is a good example. This is a technique used in photography to add color to a b&w image. The image is seperated into a series of high contrast b&w masks corresponding to grey levels in the

original. The masks are placed one by one in a color enlarger, a color is chosen for each mask, and a single sheet of color print paper is exposed. Each mask must be carefully registered, the process is exacting and time consuming, sometimes taking days to get it right. The computer can digitize an image and posterize it in seconds not days. It can cycle through thousands of color variations in minutes not weeks. And this is not the only thing a computer can do well.

I can cite more examples of things it does well--- various arithmetic and logic transformations, spatial transformations, code transformations, collaging, combining systems, and so on.

Art is not a paintbox, design is not cut and paste, both are communication systems. When art and design fail to communicate they fail period. A communication system is concerned with one thing, information. Information theory defines a successful communication system as an open-ended system with structure and at the same time a high degree of unpredictability. It defines an unsuccessful system as a closed system, rigid and predictable. A successful artists' workstation must encorporate the features of the former. The current crop of systems, however, are prime examples of the latter.

ELEMENTS OF COMPUTER GRAPHICS

MOVEMENT--- Movement is the element that distinguishes film, video and computer graphics from the traditional visual arts such as painting, printmaking and even sculpture. In order to arrive at an aesthetic for computer graphics we should look closely at film and video. Movement in film and video is the result of time plus energy. Images are projected in rapid succession, from 10 to 24 fps. The eye doesn't detect single images or frames, rather, it sees a continuous image in which objects can appear, disappear, move, change shapes, change into one another and so on. Besides the illusion of motion, film and video create an illusion of 3-d space.

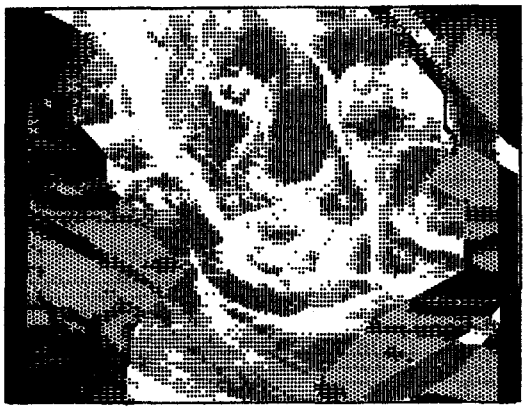
TIME PLUS ENERGY--- In particular, video is a medium in which both time and energy are under the control of the artist. Film animation allows the artist control over each frame but it's a long tedious process. Video offers instant gratification. It's almost like having a visual instrument, a real color organ, John Whitney's animations in real time.

WAVEFORMS--- The video image is transmitted, recorded and played back as a complex waveform. Waveforms are a primary element in any video system--- high frequency waves can be seen as objects in the image, low frequency waves can be used to control fades, wipes, color transformations, keying or any other voltage controllable parameters of the video signal. Electronic music uses waveforms in the same way, in fact, the first video synthesizers were like electronic music synthesizers. There are sine, triangle and pulse waves. Waves can be synchronized, triggered, summed, differenced, multiplied together, continuous, one shot, transformed from one shape to another.

IMAGE AND SYNCHRONIZATION--- The video signal is a composite signal containing image and sync information. Sync pulses mark the beginning of each frame and the beginning of each line in the frame. Playing around with the sync signal can cause the image to wraparound vertically and horizonally, or break up completely.

ANALOG VS DIGITAL--- Video is changing from analog electronics to digital electronics. In analog electronics the waveforms are represented by continuously varying voltages, in digital electronics waveforms are represented by numbers. Color cameras have computer controlled registration and color balance. Special effects systems contain digital frame buffers, analog to digital and digital to analog converters are used in video digitizers and sound digitizers. All this can be controlled by a computer.

PIXELS--- Computer graphics and video are becoming one in the same thing. The various elements of video seem pretty basic, but wait. What is the basic unit in digital video and computer graphics ? My vote is for the pixel as the basic unit of information. The pixel is described by screen location, line number and horizontal offset, and by color. As in many communication systems, the pixel is transparent, meaning is carried by the interaction of many pixels not by an individual pixel. Think about painting--- the basic unit is the brushstroke, the picture is composed of many brushstrokes. An artist understands that he/she is not painting trees and houses but is making brushstrokes on canvas. Somehow this fact has been overlooked in an alarming number of computer graphics systems, systems which paint only trees, houses, and rocket ships in space. These systems use 3-D shapes as the basic units, they use a display file. There are, however, raster based systems using pixels as the basic unit (see RASTER below). As artists' workstations, the raster based systems offer much more potential. Again, think about painting--- complex 3-D calculations are not necessary to create the illusion of space in a painting, proper perspective isn't necessary, neither is shape for that matter. I have the feeling that history is repeating itself and that the concern with pictorial representation in computer graphics is just about as necessary as in painting after the invention of photography. As computer artists we should be interested in more than representation.

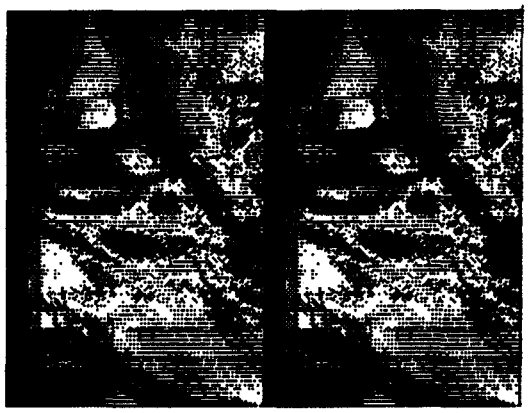


CINEMATIC CODES--- Film, video and computer graphics share principles of design such as point, line, plane, value, color, texture, pattern, contrast, balance, proportion, focal point, positive/negative space and so on. But unlike design these principles operate in both time and space, over a sequence of images and like music or dance additional principles such as repetition, movement, rhythm, harmony, counterpoint and metamorphosis become important. Further, film, video and computer graphics share a unique set of "cinematic codes" including lens angle, camera angle, camera speed, focus and depth of field, camera movement, object movement and so on.

THE FRAME--- The frame is an important element on its own, especially when a work is shown. Film theorists see it as a picture frame, a determining factor in the compostion of an image, or as a window unto another world, or as a mirror showing the audience things they may or may not want to know about themselves. All are useful metaphors.

CUTS--- A film, video or computer animation is broken into a series of "shots" (a shot is a consistent sequence of images). Cuts occur when two shots are "butted" together--- from one shot to another different or related shot, a wide angle to a closeup or detail. "Cutting on the action" means the shot begins or ends before during an action (leaving residual movement which carries into the next shot and forces the viewer to complete the action thus making the cut transparent). Cuts are like the spaces between notes in a musical phrase, cuts determine the relationships between shots and therefore determine such things as rhythm, harmony, counterpoint, etc. Cuts are the intangible glue that hold the work together.

FADES--- Fades occur when two shots are run together, unlike cuts the shots are not "butted" together rather the first shot fades out while the second shot fades in. There are short or long fades, continuous fades, multiple fades, fades to black, fades to color and so on. Video special effects generators can produce a variety of fades and wipes.



COLLAGING--- 16mm film cameras and some 8mm film cameras allow multiple exposures, images can be combined, layered or "collaged" together in a controlled fashion (counting frames and controlling object movement) or in a more organic way (controlling composition on seperate passes). It's possible but expensive in video. It's possible and practical in computer graphics, various images can be combined in all sorts of ways--- addition, subtraction, averaging, multiplying, selective keying.

KEY OR MATTE--- A key or matte combines images by removing a piece of the first image and replacing the portion removed with another image. It can be used to build special effects such as placing an object over a background (the newscaster appears over images of fresh disasters) or to collage different images together (surrealism). There are static and moving mattes, b&w keys (video) drop black out of the first image and replace the black with a second image (text over an image), chroma keys (video) drop out a specific color (usually blue), soft or hard edge keys, etc. Wipes are just moving mattes, the matte or key source is often a third image, the first image appears inside the matte and the second outside the matte. Things can become rather involved in short order, multiple mattes, overlaying mattes, setting priorities within a frame or over a sequence of frames. In computer graphics the artist can key on any color or series of colors.

COLOR MAPS--- A number of computer graphics systems display a given number of colors simultaneously, these colors are numbered and their actual red, green and blue components are stored in a table. This table is the color map and the entries in the map can be changed by the computer frame by frame. Colors can be assigned randomly, shaded, rotated by color number or by color component, components can be set independently, etc.

RASTER--- Not a jamacian religion, this is the scanning pattern used to produce a video image. It is driven by the sync pulses which determine each frame and each line. Computer displays are most often raster based. On some systems the timing can be changed giving different resolutions. A digitized image is stored sequentially in a frame buffer and divided up in a manner consistent with the raster. Each pixel can be assigned a position in memory based on line number and horizontal offset. The raster based system allows the artist direct control over each pixel each frame. Groups of pixels may be looked at in order to determine a single pixel in a new frame (pixel averaging or convolution). The raster can be subdivided into areas and these areas can be moved, copied, added, subtracted, multiplied, averaged, keyed, XORed into a new frame or into the existing frame. On some frame buffers areas of the buffer can be be set to grab and hold an image while other areas pass a live signal. The starting addresses for beginning of frame and beginning of line can be offset to produce wraparound vertically and horizontally.

LOGIC FUNCTIONS--- Not available to filmmakers or to most video artists, these functions are familiar to programmers. A digitized image is stored as a sequence of numbers in a memory, these numbers can be manipulated like any other number in computer memory, they can be added and subtracted, they can also be ANDed, ORed or XORed together. A great deal number of possibilities are available if color transformation and logic functions are combined to produce keys and mattes.

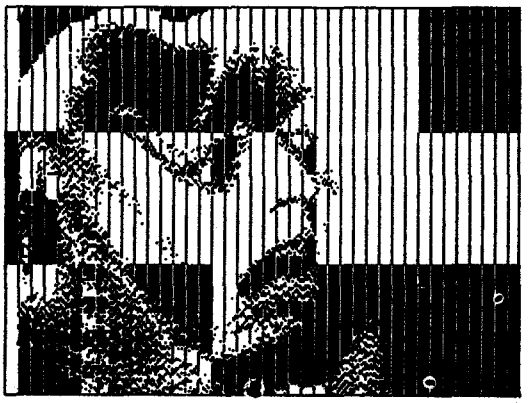
PROGRAMMING--- There is a continuing debate amongst artists and educators as to whether computer artists need to know programming. Can they get along just fine with graphics systems written by "real programmers" and not bother with the details of how the computer actually works ? I don't think so. Usually the preprogrammed systems do only a small part of everything the artist wants to do. He/she is limited by the software. Sometimes several graphics systems can be combined to produce interesting results. Soon, however, the artist discovers something he/she wants to do that no system will do. The only cousre left is to learn to program, preferably in machine language. I think every serious computer artist should be a programmer. This means he/she understands the computer and all the steps necessary to create an image and it means the artists can control every step in the process. Without this understanding and control the artist cannot transcend the medium, or at least the technology, a problem understood by filmmakers and musicians. That is, the computer artist should not have to think about the machine while creating, it should disappear from his/her consciousness. The artist should be thinking only about the product not the process. So why do artists continue to use preprogrammed systems ? Because they're lazy and they're afraid to learn programming.

ALGORITHMS--- When the artist learns programming he/she discovers a whole new world of possibilities. Besides logic functions, programs offer a variety of structures that can be applied to creating single images or sequencing groups of images. Loops, branches, conditional branches allow the artist to describe and control various situations--- if a pixel is yellow or blue make it red, if a pixel is black replace it with a pixel from another image, if this frame has too much red in it replace the red with random colors, and so on. The program can be made to respond to external conditions, various control devices, to a clock.

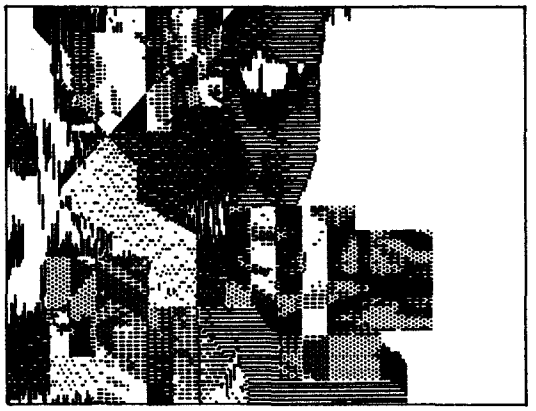
RANDOMNESS- Often a program is too repetitive, the changes become too predictable. A random number generator can add a measure of unpredictability. Sometimes too much. There are white noise, brownian motion and fractal algorithms for producing random numbers. The fractal algorithms produce naturalistic (not necessarily realistic) results and should be used more.

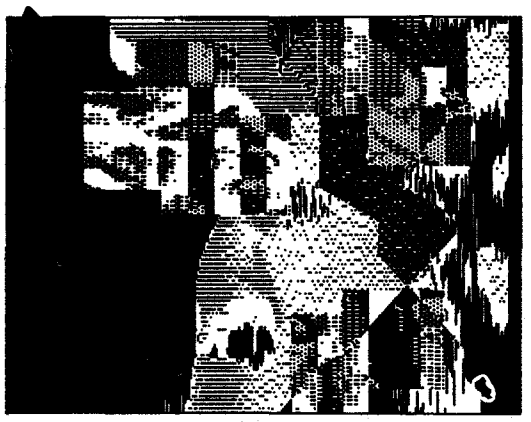
BACK TO AESTHETICS

Aesthetics describe the ways in which the elements are combined in a work of art. I propose that computer graphics elements exist and that we should understand these elements of the medium before we try to determine the aesthetic value of any of its products. An easier course would be to adopt rules for determining aesthetic value from another medium, painting or commercial film for example. Another possibility would be to use some sort of objective standard--- "Is it realistic ?" However these approaches will lead us astray, computer graphics doesn't need to look like painting, it doesn't need to be realistic. It needs to look like computer graphics and it needs to produce an emotional response in the viewer, to be expressive. The means of combining these elements in aesthetic and expressive ways were first alluded to by Naum Gabo and Antoine Pevsner in the Realist Manifesto, 1920---



"We renounce the thousand-year old delusion in art that held the static rhythms as the only elements of plastic and pictorial art. We affirm in these arts a new element, KINETIC RHYTHMS as the basic forms of our perception of real time."





Author Index

Beebe, C. 29
Berkowitz, S. 97
Denton, T. 101
Glassmire, C. 19
Holynski, M. 9
Karpinski, G.S. 69
Kerlow, I.V. 57
Kirsch, J.L. 65
Kirsch, R.A. 65
Kolomyjec, W.J. 3
Kram, R. 117
Lewis, E. 9
Mercuri, R. 73
Moldauer, W.S. 1
Ozmon, M.M. 47
Palyka, D.M. 39
Porett, T. 107
Righter, D. 73
Shafran, J.K. 91
Shortess, G.K. 13
Spiaggia, C. 29
Wright, W. 123

PROCEEDINGS

5th symposium on small computers in the arts

ISBN 0-8186-0689-4
IEEE CATALOG NO. 85CH2218-6
LIBRARY OF CONGRESS NO. 85-62328
IEEE COMPUTER SOCIETY ORDER NO. 689