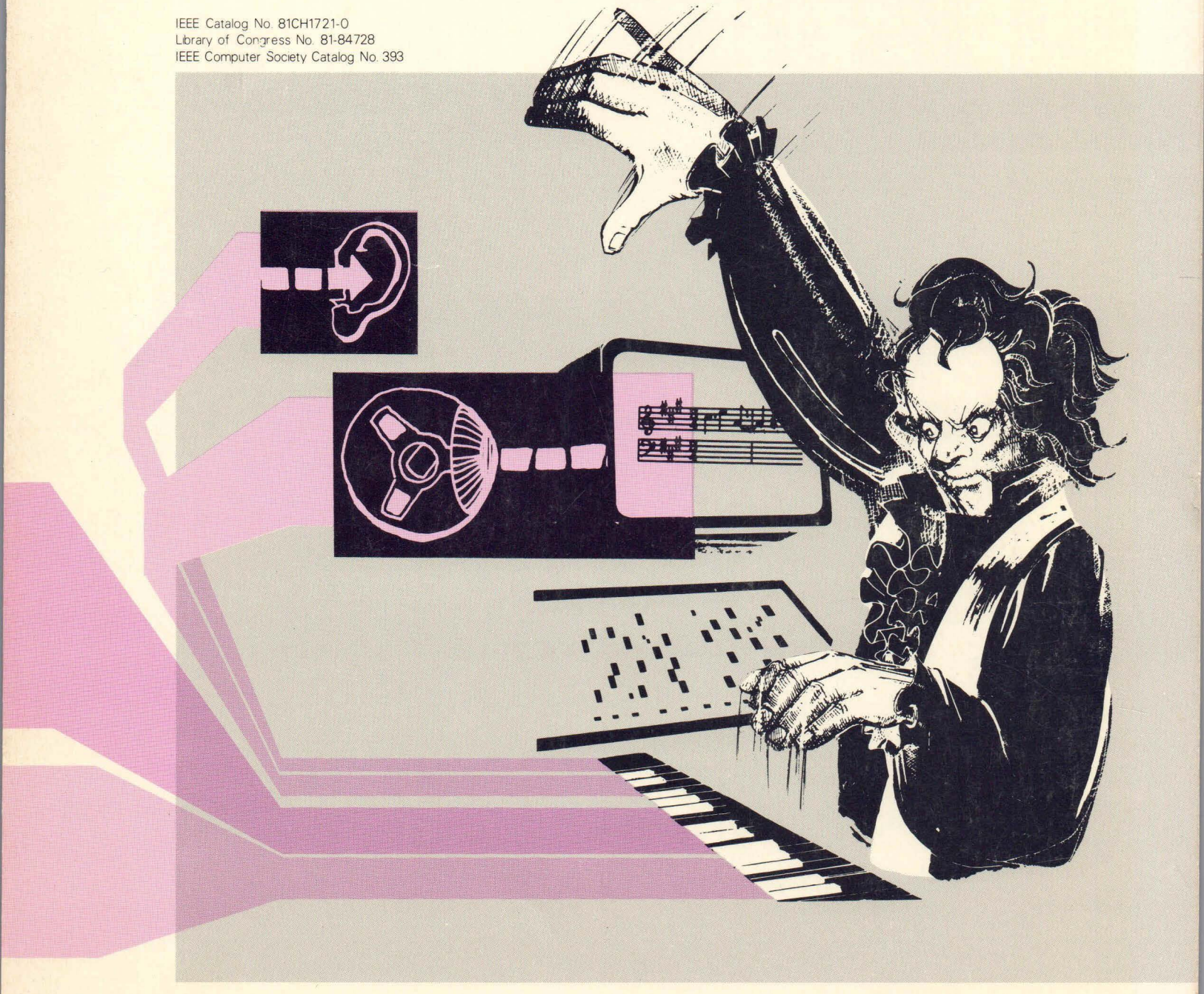
PROCEEDINGS

symposium on small computers in the arts

NOVEMBER 20-22, 1981
PHILADELPHIA, PENNSYLVANIA

SPONSORED BY:
IEEE COMPUTER SOCIETY and IEEE PHILADELPHIA SECTION

IEEE Catalog No. 81CH1721-0
Library of Congress No. 81-84728
IEEE Computer Society Catalog No. 393



THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

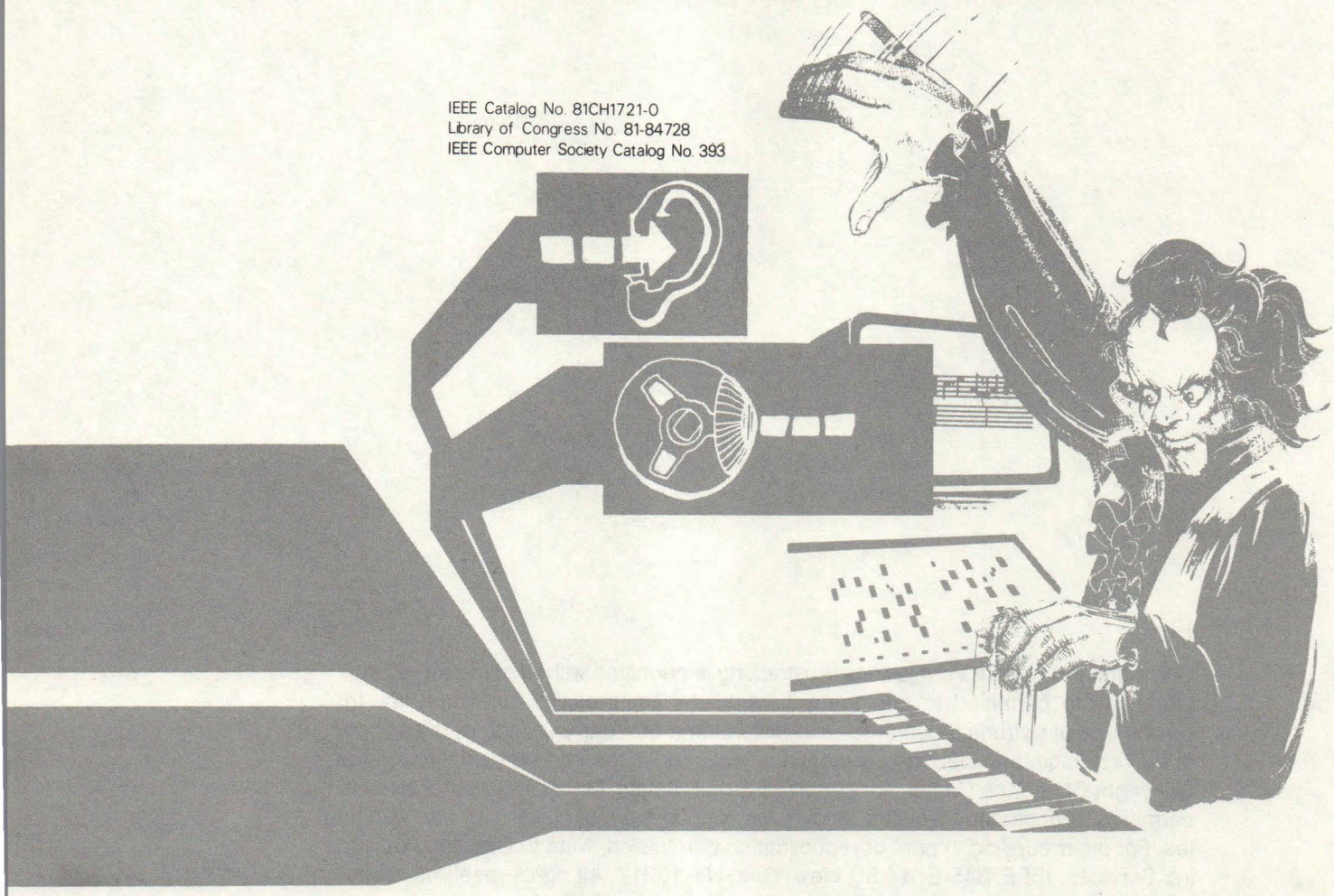
IEEE COMPUTER SOCIETY PRESS

PROCEEDINGS
symposium on small computers in the arts

NOVEMBER 20-22, 1981
PHILADELPHIA, PENNSYLVANIA

SPONSORED BY:
IEEE COMPUTER SOCIETY and IEEE PHILADELPHIA SECTION

IEEE Catalog No. 81CH1721-0
Library of Congress No. 81-84728
IEEE Computer Society Catalog No. 393



THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

IEEE COMPUTER SOCIETY PRESS

IEEE Catalog No. 81CH1721-0
Library of Congress No. 81-84728
IEEE Computer Society Order No. 393

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.

SYMPOSIUM ON SMALL COMPUTERS IN THE ARTS

NOVEMBER 20-22, 1981
PHILADELPHIA

SPONSORED BY:

IEEE COMPUTER SOCIETY
IEEE PHILADELPHIA SECTION

ORGANIZING COMMITTEE:

DICK MOBERG
BILL MAUCHLY
STEVE LEVINE
ERIC PODIETZ

PREFACE

THE ENTRY OF THE SMALL COMPUTER IN TO THE HOME CREATED A NEW GROUP OF COMPUTER USERS. THESE PEOPLE DO NOT HAVE A LARGE COMPANY OR UNIVERSITY WATCHING OVER THEIR SHOULDERS, MONITORING THEIR WORK. THEY USE COMPUTERS FOR FUN, NOT PROFIT. ALL OVER THE WORLD WE ARE DISCOVERING INDIVIDUALS WHO ARE USING SMALL COMPUTERS TO HELP THEM HAVE FUN, AND THAT INCLUDES, NOT SURPRISINGLY, MAKING ART. WE USE THIS TERM BROADLY, MEANING MOVIES, GRAPHICS, MUSIC, SCULPTURE. OF COURSE, MANY PRODUCTION TASKS MAY MAKE USE OF A COMPUTER TO ASSIST IN SOME ROUTINE LABOR. FOR INSTANCE, THE POET MAY USE A WORD-PROCESSOR TO EDIT AND TYPE HIS POEMS. BUT THE REAL PURPOSE OF THIS SYMPOSIUM IS TO BRING OUT THE APPLICATIONS WHICH RELY ON COMPUTERS; THOSE THAT INVOLVE THE COMPUTER DIRECTLY IN THE CREATIVE TASK, AND INVOLVE THE ARTIST WITH THE MEDIA IN A NEW WAY. IT IS HERE THAT WE ARE FINDING NEW IDEAS, NEW WAYS OF THINKING ABOUT ART, NEW WAYS OF REPRESENTING IDEAS, AND NEW PALETTES OF RAW MATERIALS.

THE PERSONAL COMPUTER IS ONE MACHINE UNDER THE CONTROL OF ONE INDIVIDUAL. THAT PERSON IS DIFFERENT FROM THE TYPICAL LARGE COMPUTER USER IN THAT HE OR SHE DOES NOT HAVE CONSULTANTS, PROGRAMMERS, OR DEPARTMENTS SET UP FOR ASSISTANCE. A PERSONAL COMPUTER USER MAY OPERATE SOLO. THIS IS ESPECIALLY TRUE IN THE MUSIC AND ARTS APPLICATIONS. FOR THESE REASONS, IT IS IMPORTANT TO HAVE THIS GATHERING OF PEOPLE INTERESTED IN THE ARTS, WHERE THEY CAN COMMUNICATE WITH OTHERS AND GAIN INSIGHTS INTO THE TECHNIQUES AND PHILOSOPHIES OF THEIR COHORTS.

HISTORICAL NOTES

THIS SYMPOSIUM GREW OUT OF A COMPUTER MUSIC CONCERT HELD IN DOWNTOWN PHILADELPHIA IN 1978. IT WAS PLANNED AS PART OF THE PERSONAL COMPUTING '78 SHOW HELD AT THE CIVIC CENTER. JOHN DILKS, THE FOUNDER OF THE SHOW, GRACIOUSLY BACKED THE IDEA AND PROVIDED A HOTEL BALLROOM FOR THE EVENT. AS WORD OF THE UPCOMING CONCERT SPREAD, WE RECEIVED CALLS FROM PEOPLE AS FAR AWAY AS THE WEST COAST ASKING IF THEY COULD PARTICIPATE. ONE MUSICIAN FROM NEW YORK ACTUALLY ARRANGED A PIECE FOR COMPUTER AND CLARINET ESPECIALLY FOR THIS CONCERT.

THE EVENING OF THE CONCERT OVER 500 PERSONS SHOWED UP AND TRIED TO SQUEEZE INTO A ROOM THAT ONLY HELD 300. THE CONCERT WAS RECORDED, AN ALBUM WAS MADE, AND IT IS NOW SOLD BY CREATIVE COMPUTING.

THE SUCCESS OF THAT CONCERT LED THE ORGANIZERS TO FORM AN INFORMAL PERSONAL COMPUTER ARTS GROUP TO PRODUCE SIMILAR EVENTS AND TO ACT AS A CLEARINGHOUSE FOR THOSE INTERESTED IN COMPUTER APPLICATIONS IN THE ARTS. THE 1979 PERSONAL COMPUTER MUSIC FESTIVAL, SPONSORED BY THE GROUP, INCLUDED TALKS AND DEMONSTRATIONS DURING THE DAY IN ADDITION TO THE EVENING CONCERT. IN 1980, A SEPARATE DAY OF COMPUTER GRAPHICS TALKS AND DEMONSTRATIONS WAS ADDED TO MAKE THE PERSONAL COMPUTER ARTS FESTIVAL. ALL THESE EVENTS WERE HELD AT THE PERSONAL COMPUTING SHOWS IN PHILADELPHIA AND WERE BACKED BY JOHN DILKS. A FEW OTHER EVENTS PROMOTING COMPUTERS IN THE ARTS WERE HELD DURING THE LAST YEAR.

IT HAD ALWAYS BEEN OUR DESIRE TO SOME DAY ORGANIZE A MAJOR MEETING SOLELY DEDICATED TO THE USE OF SMALL COMPUTERS IN THE ARTS. THIS DREAM BECAME A REALITY WITH THIS SYMPOSIUM THANKS TO THE SUPPORT OF THE IEEE COMPUTER SOCIETY AND THE IEEE PHILADELPHIA SECTION.

THE PERSONAL COMPUTER ARTS GROUP, AS A VOLUNTEER NOT-FOR-PROFIT GROUP, CONTINUES TO PROMOTE THE USE OF COMPUTERS IN THE ARTS THROUGH ITS NEWSLETTERS, CONCERTS, AND OTHER EVENTS. TO CONTACT THE GROUP, WRITE TO: PERSONAL COMPUTER ARTS GROUP, BOX 1954, PHILADELPHA, PA, 19105.

SYMPOSIUM ON SMALL COMPUTERS IN THE ARTS

TABLE OF CONTENTS

PART I

SMALL COMPUTERS IN MUSIC PERFORMANCE COMPOSITION AND SYNTHESIS

INTRODUCTION

1981 MARKED THE INTRODUCTION OF SEVERAL NEW COMMERCIAL SYSTEMS FOR MAKING MUSIC ON PERSONAL COMPUTERS. THE READY AVAILABILITY OF RELIABLE SYSTEMS HAS BROUGHT COMPUTER MUSIC TO A NEW AUDIENCE: THE COMPOSERS AND MUSICIANS. IN THE SAME WAY THAT "APPLIANCE" HOME COMPUTERS SAVED THE USER FROM THE UNNECESSARY COMPLEXITY OF BUILDING A COMPUTER SYSTEM FROM SCRATCH, NOW MUSIC PROGRAMS AND PERIPHERALS FREE THE MUSICIAN FROM LOW-LEVEL HARDWARE AND SOFTWARE INCONVENIENCE. THIS IN TURN HAS SPAWNED INDEPENDENT SOFTWARE PROJECTS. WE ARE SEEING THE FIRST WAVE OF CREATIVE, INDIVIDUALIZED MUSIC SYSTEMS.

THE SYMPOSIUM POSES SOME SPECIFIC ANSWERS TO THE PROBLEMS OF MUSIC COMPOSITION BY COMPUTER. PAPERS ON PATTERN MANIPULATIONS, STOCHASTIC PROCESSES, AND CANON WRITING, SHOW HOW THE COMPUTER CAN HELP THE COMPOSER GENERATE MUSICAL MATERIAL. OTHER WORK TREATS THE PROBLEM OF REAL-TIME PERFORMANCE, WHERE THE COMPUTER MAY ASSIST THE PERFORMER IN CONTROLLING AN INSTRUMENT.

THE FINAL AND MOST TECHNICAL SECTION PRESENTS SOME NEW WORK IN SOUND SYNTHESIS. MANY FUNCTIONS WHICH HAVE BEEN DEVELOPED ON LARGE SYSTEMS ARE BEING IMPLEMENTED THROUGH CREATIVE HARDWARE AND SOFTWARE DESIGNS FOR SMALL SYSTEMS.

THE PIPE ORGAN AND THE MICROCOMPUTER

Dr. Robert Suding

Problems facing a microprocessor assisted pipe organ

Earliest versions of "pipe organs" date back many centuries to Pre-Christian times. The pipe organ as generally known today in churches was well developed by the end of the 17th century except for electrical keying and wind pressure blower systems. These latter two were accomplished almost 100 years ago. The pipe organ is indeed a very old and mature technology, and most innovations have traditionally met with a great deal of resistance.

Even such a common idea as coupling the pipe system(s) to the keyboard with electricity is discarded by some, claiming that without the mechanical linkage they lost a tactile sense and a playing style when only electrical contacts are made.

Another fear is what will occur under a failure condition. The pipe organ is a mass of parallel matrix switches, the proper selection resulting in a pipe "speaking". Two failure modes are typical. The most common failure is that a pipe does not speak when ordered. The other more serious failure is that a pipe speaks without being selected, referred to as a cipher. The first problem is temporarily overcome by not using the group of pipes where the silent one resides, or using another similar rank to fill in for the missing note. The cipher is cured by removing the offender from its pipe rack and treating the problem as a missing note.

Now visualize the position of yourself trying to introduce a new fangled microcomputer to a pipe organ. This pipe organ can have from 1000 to 10,000 pipes. Can you imagine the sound that would occur if a microprocessor failure turned on every pipe?

The typical pipe organ wiring system is conceptually relatively simple, but works with thousands of wires and contacts in parallel. Some have suggested that since the pipes take a certain amount of time to react to the keying of the wind pressure, about 1/10th of a second "refresh" time would be adequate to sample the key, calculate the pipes required, and turn on the required valves. Unfortunately this produces an effect that nullifies the "personality" of a performer. In the real world, chords are not simultaneously struck, but are "rippled on", so that lower (or upper) parts of the chord come on ahead of the rest of the chord. While difficult to pinpoint the cause of the realism lack, experiments indicate a key/calculate/pipe refresh cycle requirement of 4 milliseconds or less.

A pipe organ typically enjoys a very long life, often measured in centuries. Electronic technology however is not so stable. Obsolescence is measured in years or even months. While defects may occur on the pipe organ mechanical parts, replacement parts are readily available a century later. Which electronic part do you think will be available for easy replacement in 2081?

Advantages of a microprocessor assisted pipe organ

The microprocessor does offer an outstanding advantage: Flexibility. The typical cabling system for an organ consists of several large cables of thin wires going through a number of mechanical relay banks to tables of pipes. Several miles of wire are involved and many thousands of contact points, even in a small pipe organ system. The stop tabs and couplers on the console are then engraved to reflect the wiring architecture. End of job for 20-30 years. But tastes change. Pipes hardwired to one division often are captive. Experimental ensemble combinations are discouraged by the permanence of the wiring.

The microprocessor driven pipe organ gives a degree of freedom and experimentation to practice sessions. The performance may be digitally logged and replayed. But the degree of replay may be customized. The replay of notes can be accompanied by a new combination of stops and couplers. The performer is able to eliminate muddy passages, or trim the ensemble effect to achieve the composer and performer's interpretation of each passage. Initial practice sessions may record a single hand on a difficult passage. Replay of the prelogged performance may then be "ORed" with the next hand, creating the ensemble effect lost when only a single (or foot) is utilized initially.

The last (and least desireable) possibility is the complete performance by the computer. Insistence on this obvious possibility will guarantee failure. The organists of the world will

CH1721-0/81/0000/0005$00.75

PROBLEM	COMMENT
● New systems not accepted.	● Show as an aid to the organist, not a replacement.
● The tactile feel of a tracker action is lost.	● "Solid State tracter action" would introduce a new and highly marketable system.
● Failure may result in a catestrophic sound, or total crippling during a performance.	● Redundant system architecture needed, and fail safe anti-cipher designs are mandatory.
● "Inhuman effect" caused by computers' inflexible timing subsystems.	● Update system cycles in less than 4ms. Randomizing programming required if "player organ" implemented.
● Technology obsolescence	● Extremely difficult to avoid. Use only the commonist parts. Provide easily redefinable subsystems to allow substitution. Provide spare parts. Far overrate. Overprotect. Provide standard connectors.

GAIN	COMMENT
● Flexibility	● Minimize matrix wiring subsystems. Do not skimp on driver system. Do not try to take input shortcuts.
● Practice assistance	● Provide alterrate real time input and output systems. Spend the majority of the design time at the critical interchange points.
● Performance aid	● Auto-detect ciphers and disconnect if possible, otherwise notify which rank has the cipher.

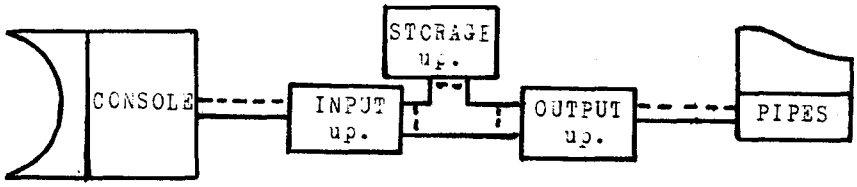
rise enmasse and stomp on your IC's.

The architecture of the pipe organ computer system must be designed to overcome the problems, while achieving the gains. The table above lists the problems and gains with comments.

System Alternatives

Many architectures are possible, but the need for flexibility and speed eliminate most. The key item to remember is that over 200 inputs myst be scanned and over 1000 pipes must be potentially actuated at a rate of greater than 250 times a second (4 ms each time). A complete scan, calculate, and output in 4 ms is far beyond present economical technology.

However, voice and video technology are both pointing towards various reduced bandwidth systems. Perhaps the delta modulation systems may provide an acceptable solution. Investigations of how pipe organs are actually played show that over the desired 4 ms time slice comparatively small amounts of change occur when compatred with the previous 4 ms time slice. The solution then resolves to building a work list of changes to be made. General Housekeeping is performed during "No Change" time slices, Failure Prevention Housekeeping is accomplished during "No Key Input" time slices, etc. Over a dozen time slice job list functions have been defined. Some are difficult, some easy. A Failure Prevention subroutine is simple; but coupler tab actuation with a sustained chord severely taxes the 4 ms limit.



PIPE ORGAN MICROPROCESSOR SYSTEM

The figure above illustrates a simplified block view of my selected architecture. Three microprocessor subsystems, consisting of Z-80B based modules proved 1) Input and delta worklist maintenance, 2) Output, calculate, and failure bypass, and 3) Storage, retrieval, and auto-calibrate.

The input requirement consists of gathering the 200 plus input lines into memory and checking for changes. Either parallel or parallel load-serial transfer systems are feasible. I have chosen the serial transfer system since this reduces the interconnect lines and provides a greatly reduced number of lines on the input microprocessor board (read that improved reliability!). The inputs consist of a 4021 CMOS I C for each set of 8 inputs. Although there are over 200 inputs, a cable of only 8 wires (+, -, Load, Clock, and Data) connects the console to a remotely located input microprocessor. One hundred percent redundancy is another 5 wire cable. The change analysis subroutines require a small amount of dual buffering. Keys, Stops, and Couplers are separately maintained in logically partitioned areas of the buffers. The work list buffering is Fifo-like in architecture, pouring its preliminarily processed data into the output microprocessor's workspace.

The output microprocessor also may be a parallell or serial/parallel based subsystem. In my case, I again selected serial transfer over a 5 conductor cable plus a redundant cable. A 4094 CMOS 8 bit shift register and latch is used for each 8 pipe magnets.

2N6724's at 20 cents apiece provide the magnet drivers with a current capacity of 2 amps, although only 150 ma is typically required.

The actual driver cards are separate from the output microprocessor and contain enough drivers for one or two ranks (each rank contains 32-97 pipes). The output microprocessor is in charge of these output cards and handling the partially processed data from the input microprocessor and any input from the storage microprocessor. Should the output microprocessor fail to send new data to any rank for more than 10 ms, the rank drivers will go to a "silent pipes" failsafe condition. The output microprocessor has the capacity to reassign pipe ranks to other divisions as well as make unique stops and couplings appropriate to certain musical combinations, (e.g. a soft or loud mixture rank formed to support a division).

The storage microprocessor is the most flexible, and many commercially available microprocessor systems are viable here, but the redundancy requirements would eliminate most systems. Off line storage could take many forms, but the Winchester disk offers speed/capacity that permits a rapid "simultaneous" read/write capability. "Simultaneous" read/write allows the perfomer's input from the console to be logged while the storage subsystem is also providing input to pipe driver in "play along with the computer" applications. Computer assisted composing would also utilize the "simultaneous" read/write storage system.

The costs will vary widely, but my rule of thumb for pure component costs is approximately one dolloar per input or output bit (a bit may be a key, stop, pipe or magnet) for the medium sized input and output subsystems. The organ of 3 manuals and 60 stops along with 24 ranks (1500 pipes) which I have will have abjout $1200 in parts. Redundancy will add about 50 percent. The offline storage doubles the previous costs. If the system were to be made into a commercially available package, $10,000 might be a reasonable price to expect. But you are greatly enhancing the capabilities of a $100,000 pipe organ. The on site wiring costs of this sized pipe organ could exceed the cost of the microcomputer system. The music hobbiest should be aware that used pipe organs of 40-60 years old vintage are often sold by churches and theaters. The cost of these "old relics" can be surprisingly low, frequently in the range of $2-$4 per pipe for a repairable organ system. Visitors to my house tend to bring their wives along to see my 1500 pipe organ. The wives generally conclude that their husband's hobby isn't so bad.

If only a simple input and/or playback function is desired, the ports of a microprocessor can be only connected to the keyboard of an organ. While simpler, flexibility suffers. Electronic organs (ugh!) may also be coupled to a microprocessor. Even toy organs can use a microprocessor's output ports, but the system sinks to a lowly "player organ".

The computerized pipe organ produces an effect truly worthy of the King of Instruments. It is infinitely superior to any musical experience you have ever had.

MANUSCRIPT: Music Notation for the Apple][

Rebecca T. Mercuri

2007 Jenkintown Road, Glenside, Pa. 19038

MANUSCRIPT is a high-resolution music editor which produces pages closely resembling printed notation. The use of special compilers makes MANUSCRIPT files compatible with all music peripherals for the Apple][system. This paper illustrates the decision-making process involved in construction of an editor which is visually pleasing, easy to operate, and does not require any additional hardware.

The ability to place notes, stems and beams in any location on the musical staff, to write dynamics and phrasing markings, and the inclusion of song lyrics have been seen as desirable features in music writing. To date, music editors which incorporated these items (such as those by Leyland Smith of Stanford University, and Donald Byrd of Indiana University) could only be found on minicomputers or larger systems. The major goal of MANUSCRIPT is to provide the microcomputer user with a seemingly infinite piece of blank manuscript paper, upon which can be written any item found in conventional musical notation.

The proliferation of musical hardware for the Apple][(ALF, Mountain Computer, MMI, AlphaSyntauri, to name a few) poses a problem of incompatibility among their editors. Imagine the dismay in discovering that the file you spent two weeks perfecting on your MMI system can not be played on your friend's Mountain Computer boards. The development of individual, highly specific music editors by hardware manufacturers is analogous to the use of interpreted computer languages like Applesoft which save development time at the sacrifice of portability and speed. A secondary goal of MANUSCRIPT, therefore, is to produce files which can be compiled differently depending upon the hardware used. A side benefit is that MANUSCRIPT can become compatible with any future music systems merely by writing new compilers. This issue is complicated, though, by the fact that the number of simultaneous voices currently ranges from one (Apple speaker) to sixteen (Mountain Computer), necessitating the use of various coding methods (stem direction, color) to separate the melodic lines.

The creation of any music editor involves numerous compromises imposed both by hardware and software considerations. For example, my selection of Applesoft BASIC (rather than Pascal) and paddle controllers (instead of a joystick or light pen) was intended to facilitate wide usage, since these items are readily accessable to Apple][and][plus owners. Most difficult has been my constraint to 48k of memory, although space problems can be resolved somewhat through segmentation of the program into different functional groups (such as text writing, dynamics marking...).

Prior to use of the MANUSCRIPT program, numerous options may be selected. These include: automatic measure checking for correct rhythms, display of note name corresponding to cursor position, audio feedback of note insertion using the Apple speaker, and specification of distinct voice lines. After, or during creation of a note file, it is possible to change into dynamics

marking or lyric writing modes. A post-processor is also available for automatic beaming (connection of eighth notes etc.) and note alignment (spacing within measures according to note value).

MANUSCRIPT's high-resolution display is designed for simplicity and ease of use. The black figures on a white background provide close similarity to paper notation. The basic configuration displays four 5-line staves, although the user may select 1, 2, or 3 staves and 4-lines or 6-lines for use in banjo or guitar tablature. The bottom tenth of the screen contains a menu of musical symbols. This is a multi-page menu organized according to function as follows:

PAGE 1: Treble, bass and four C clefs (SATB), user defined meters from 0-99 over 1, 2, 4, 8, 16 or 32.

PAGE 2: Note and rest values from whole to 64th, measure bars, dotting, ties, accidentals, 8va, 15va.

PAGE 3: Repeats, 1st and 2nd endings, da capo, coda, dal segno, voice selection.

PAGE 4: Tablature notation

PAGE 5: Dynamics markings

PAGE 6: Ornamentation (trills, slurs...)

Certain of the menu items, such as voice selection and tablature notation, may only be active when those options are in effect.

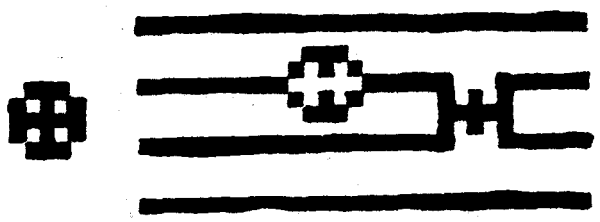


FIGURE A

Cursor design and appearance on lines or spaces.

A special cursor (see figure A) was designed to assist in correct note placement. When the cursor is on a staff line, it appears differently than when it is on a space, thus reducing entry problems. The paddles independently control x and y motions of the cursor. Menu items are selected by moving the cursor to the desired symbol and then pushing the paddle button. The chosen item will now appear shaded on the menu, and when the cursor is moved to any area on the manuscript paper, pushing the paddle button again will deposit the character at this location. The cursor may then be repositioned on the page without having to return to the menu. In the audio feedback mode, notes are heard as they are installed onto the page, and various click sounds are also available to indicate menu selections, rest entries and cursor movement.

Special care was taken in the design of the note shapes. Printed music has evolved over many hundreds of years to the point where it has become aesthetically pleasing. For a graphic music editor to be truly effective, the subtle curvatures, proportionalities and shadings of printed music must be replicated. The direct transfer from a graph paper representation to pixels is, unfortunatly, not possible because of the peculiarities of the video medium. A note head that appears almost square on paper is perceived as round when viewed on a monitor. One or two pixels, properly

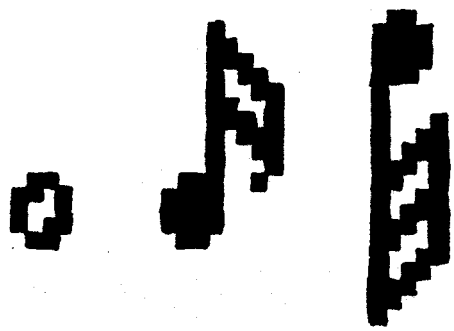


FIGURE B

Examples of whole, 16th and 32nd note formations.

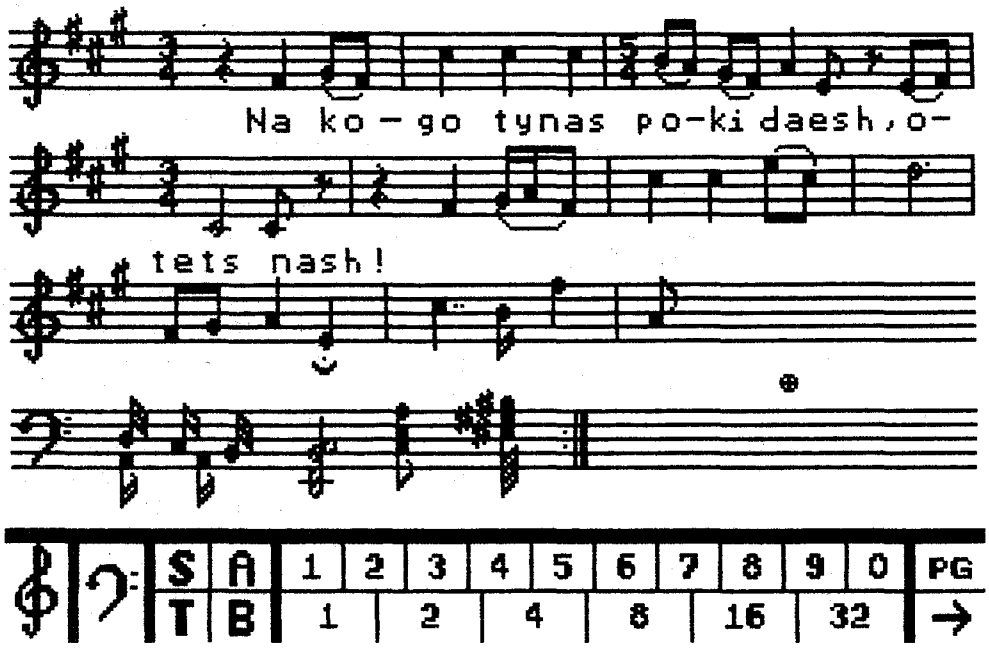


FIGURE C

MANUSCRIPT display showing excerpt from Boris Godunov and examples of chord construction. Clef and time signature menu appear beneath the music.

placed, can provide the illusion of smoothing out the jagged lines that are problematic in graphics systems without gray scale levels. Universal visual perception problems with parallel lines can cause well-formed objects to look peculiar when superimposed upon the staff; tests had to be performed in order to insure that a note would look well on either the lines or the spaces. Figure B illustrates the design of some of the characters created.

A current MANUSCRIPT display can be seen in Figure C. The program is still under development - future plans include the addition of a special notation for digitally produced sounds as well as commands for non-diatonic tunings. Further information about MANUSCRIPT can be obtained by writing to the author.

REFERENCES:

Byrd, Donald, An Integrated Computer Music Software System, Computer Music Journal, April 1977.

Grout, Donald, The History of Western Music, rev.ed., W.W. Norton and Co., Inc, 1973.

Ross, Ted, The Art of Music Engraving and Processing, Charles H. Hanson Music and Books, Inc., 1970.

NCC: A Program That Composes Canons

Michael Keith
D46 Abbington Drive
Hightstown, N.J. 08520

Abstract

This paper describes a program written for a microcomputer (the Apple II) that automatically composes musical pieces of a simple form. Specifically, the program (called NCC) composes canons, a musical piece in which all the voices play different transformations of a single theme. The algorithm used to generate the canons is based on a combinatorial property of the twelve-tone musical scale. The algorithm is explained in detail, and several examples of the types of canons which the program has composed are presented.

Introduction

The program described in this paper is called NCC, for Numerical Canon Composition. The purpose of the program is, with minimal user input, to automatically compose canons. A canon is a simple polyphonic musical form consisting of a single tune played (or sung) simultaneously with multiple copies of itself. Each of these copies is transformed in some way from the original; thus the illusion is created of a complicated, multi-voice piece of music, even though the actual information content is relatively low.

A simple example of a canon, familiar to almost everyone, is the "round". In this type of canon, the different voices sing the same melody but begin singing at different times. An example of a round is the familiar "Row, row, row your boat" (Figure 1). If we call the main melody voice 0, then we can see that voice n is merely voice 0 delayed by n measures.

One of the masters of the canon was Johann Sebastian Bach. Among his works can be found canons with all kinds of transformations; for example, compression or expansion (in time) of the melody, inversion (in which the transformed melody goes down (in pitch) where the original goes up, and vice versa), backwards, upside down, transposition (in which the whole melody is transposed up or down in pitch), and so on. Although this program cannot claim to be Bach's equal musically, it can generate all of these same types of transformations, as well as some that even master Bach probably never tried (such as "shift measure i of the original melody by p*i(mod n)measures, where p is a prime number and n is the total number of measures in the piece").

Composing Strategy

The basic problem in composing a canon, as opposed to a single-voice melody, is preventing clashes. That is, each note in the melody functions on two levels - both as melody and harmony. We must consider both of these aspects so that the final multi-voice canon will sound reasonable both melodically and harmonically. Thus, we are led to a consideration of the harmonic, i.e., chord structure of the final canon.

A simple strategy which I have adopted is the following: First, choose an underlying chord structure for the base melody so that the copies of the melody (after appropriate transformation) have the same chord structure. After determining the chord structure for the main melody, the actual melody is then computed.

Let me illustrate with an example: Suppose we want a six-measure, two-voice canon with the second voice being the first voice played backwards. Then, if we pick any palindromic sequence of chords, for example,

/ C / Fm / G / G / Fm / C /

and then compute the melody based on these chords, the complete canon is guaranteed to have no clashes at the measure level. It may have momentary clashes at the note level, but in fact this is not undesirable since it adds a little "tension" to the piece and makes it sound more interesting.

Thus our canon composer consists of two parts; a chord sequence generator, and the actual melody composer. I will describe each of these briefly.

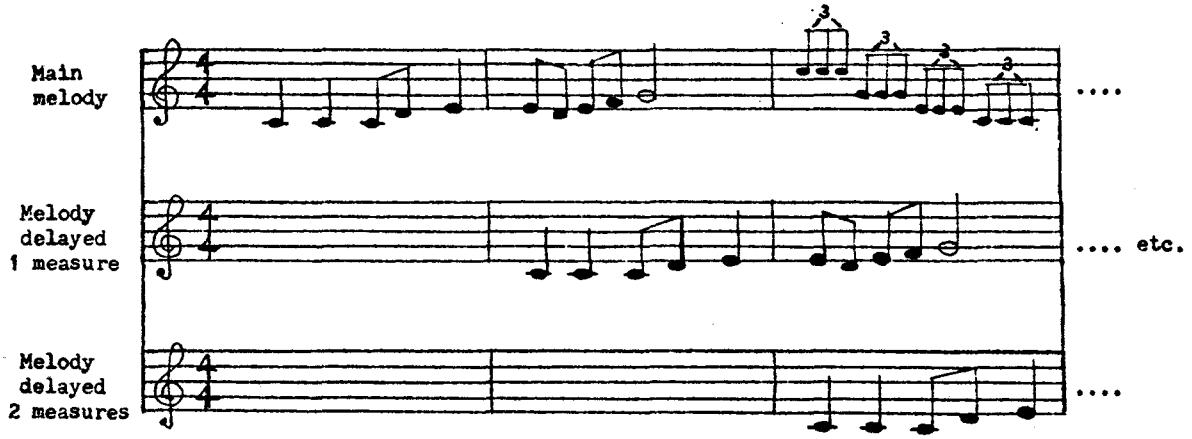


Figure 1

Sample Canon (round).

Chord Sequence Generator

The NCC program uses a simple but powerful method of generating the chords. It is based on the following combinatorial property of the twelve-tone scale. If we define a chord as two or more notes of the scale (with adjacent notes ("clashes") excluded), then it turns out that there are only 30 distinct chords. In other words, every chord is either one of these 30 or a transpose of one. See reference 1 for more details. We can consider the entire chord space, then, as 30 x 12 = 360 points, each identified by a tonic (C, C♯, D,...) and a chord type (1 - 30). This is represented schematically in Figure 2.

We will describe the chord structure of the canon probabilistically, so that our program will generate different canons on multiple runs (even with the same set of inputs). This is done by considering Figure 2 as the graph $K_{12} \times K_{30}$ (the direct product of the complete graphs on 12 and 30 points) and by assigning a probability to each arc in the graph, with the restriction that the sum of the probabilities eminating from each node = 1. These probabilities will be interpreted as the probabilities of changing chords from measure to measure. By convention, the canon always begins at the point corresponding to the major chord with tonic equal to the key of the piece. After each measure, the graph, which I call the Transition Graph, is consulted to determine the chord for the next measure. The sequence of chords in the piece corresponds to tracing out a path in the Transition Graph.

Of course, the graph of Figure 2 is very large, having roughly 100,000 edges. Thus in practice, only a few of the edges are given non-zero probabilities; this corresponds to restricting our random walk to a small subgraph of the transition graph. As each chord is generated, its transformed copies are also generated. This process continues until a chord has been generated for each measure of the piece.

Melody Generator

Once the sequence of chords if generated, the melody can be generated by any single-voice melody algorithm. The algorithm I use is very simple: A list of measure note patterns (compatible with the current time signature) is supplied by the user. For example, for 4/4 time, the list might include:

/whole/, /half half/, /half quarter quarter/,..

Each of these is assigned a probability. For each measure, one of these patterns is chosen, then the pitches of the notes are chosen by choosing from the notes of the underlying chord, with possibly a random perturbation up or down a few half-steps. This results in a melody that sounds reasonable within the harmonic setting of each measure.

Results

The program which implements this algorithm is approximately 8K of Applesoft BASIC code. On each run of the program, the user can specify the

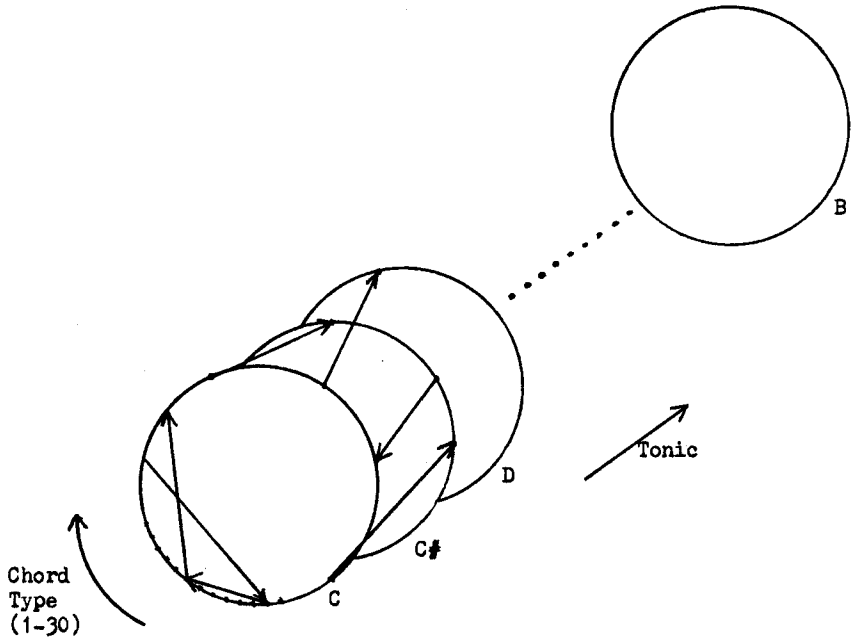


Figure 2

Chord Transition Graph (K_{12} x K_{30})

(A few representative arcs are shown)

---

following parameters:

1. Length of canon,
2. Time signature,
3. Transition graph probabilities,
4. Note patterns & probabilities,
5. Pitch distribution.

The output of the program is a file compatible with a homebrew music system called "BACH" (see refs. 2 and 3) and can be viewed in standard musical notation. Two sample canons composed by this program are shown in Figure 3. Both are upside-down canons (they read the same when the music is turned upside down). The input parameters for these two examples differed only in items 1, 4, and 5 above, but even this slight change in the input parameters is readily obvious in the music (the Crab1 canon, for example, has no accidentals, a consequence of the pitch distribution used). Each canon required less than a minute to compute, which is quite fast considering that the program is in interpretive BASIC.

Musically speaking, the canons generated by the NCC program, while not masterpieces, are generally pleasant, and occasionally surprisingly good. One modification I plan to add to the program is the addition of a countermelody to the completed canon (using the same underlying harmonic structure). This should make the final piece even more musically interesting. Other additions to the program could, undoubtedly, produce even better results.

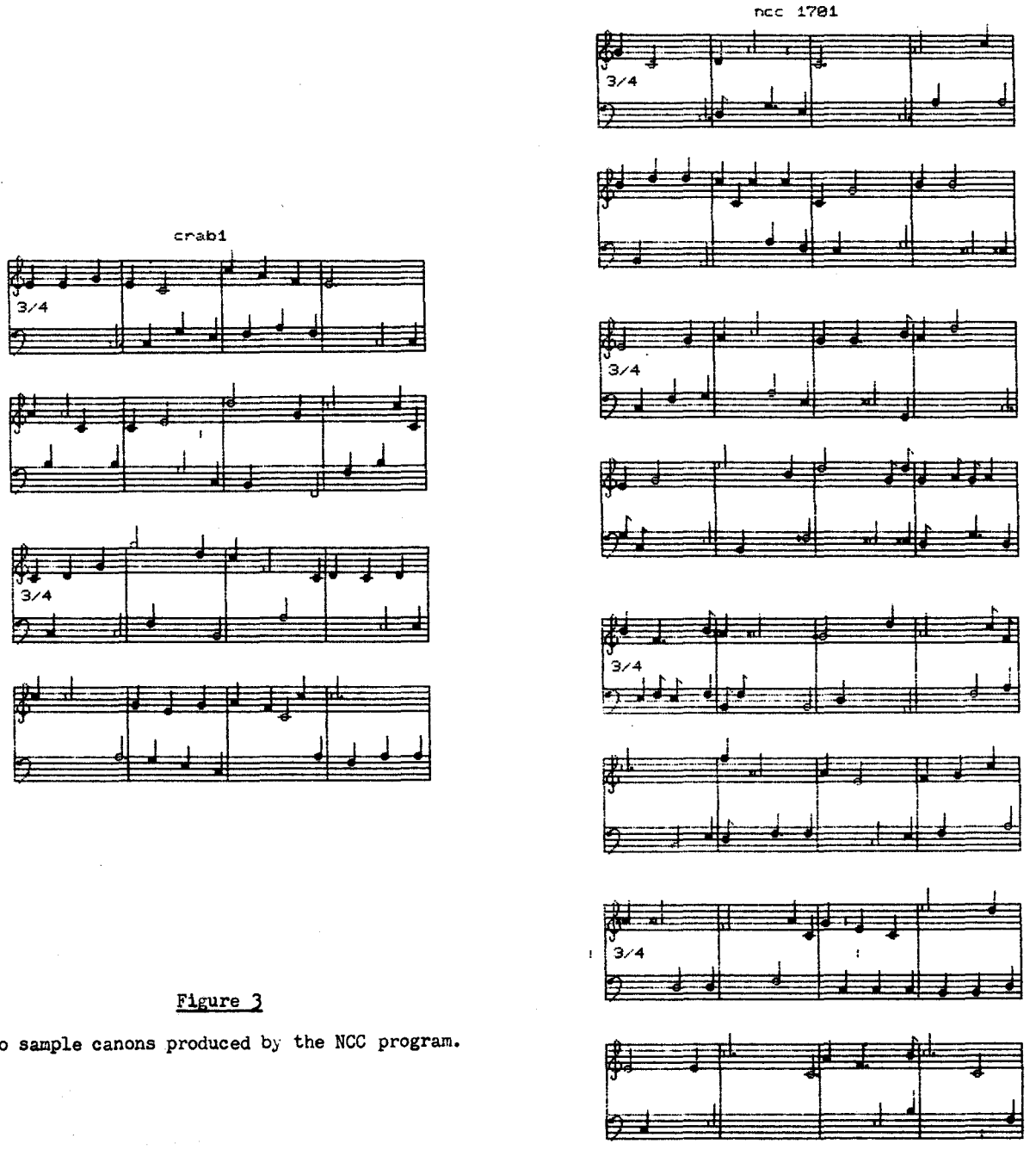


Figure 3

Two sample canons produced by the NCC program.

References:

(1) "Some Combinatorial Aspects of Computer Music Composition", M. Keith, Personal Computing Festival Proceedings, 1979.

(2) "BACH: A System for music composition and display on a microcomputer", M. Keith, Personal Computing Festival Digest, 1980.

(3) "BACH", M. Keith, Creative Computing, Feb. 1981.

COMPUTER-AIDED COMPOSITION - AN EXAMPLE

Małgorzata Mikulska

Department of Mathematics
State University of New York at Buffalo
Buffalo, New York 14214

Abstract

In this paper, an example of computer-aided composition is presented: "Amazonia" for flute,oboe, clarinet and percussion. The piece described is an attempt to formalize certain stages of the compositional process: organization of rhythm and pitch structure as well as some more general parameters, e.g., instrumentation and the global development of the texture. The method used is a combination of formal procedures and arbitrary artistic decisions. In formalized stages of the compositional process a "semi-stochastic music language" is used, combining deterministic and stochastic elements: deterministic general principles and stochastically determined details.

Formalization of the process of musical composition is a necessary starting point for computer simulation of this process. Various stages of the compositional process and various parameters are suitable for formalization to a different degree. Usually, such stages as arganization of rhythm and pitch structure are easier to formalize than is the general structure of a piece or its form. In fact, all too often the latter comes out as an unforeseen and random result of all remaining parameters.

Another problem worth investigating in music (although not necessarily connected with the use of computers) is the controversy between determinacy and randomness, and the role of random events in musical composition. In fact, randomness as an active factor of musical composition has been hardly ever considered consciuosly and seriously till the middle of our century. Most, if not all, music of our culture seems to aim at determinacy. It tourned out, however, that in certain cases strict determinacy and free randomness may produce similar results. This is, for example, often the case with total serialism where stricly deterministic uniform distribution of all parameters can give almost the same result as random distribution of these parameters with uniform probability distribution (see Xenakis[2]).

This paper contains some remarks concerning both topics: formalization of the compositional process and the role of random events in musical composition. Observations have been based on the author's own composition "Amazonia" for flute, oboe, clarinet and percussion (1980-81).(A performance of the piece was scheduled for fall 1981 at SUNY at Buffalo.) This composition is an attempt to formalize certain stages of the compositional process; the formalization, however, does not embrace all compositional decisions. In fact, the method used is a free combination of formal procedures and informal, musical decisions. It can be described as a "semi-stochastic musical language" (see Myhill[1]), with deterministic general "guidelines" and stochastically determined details. Furthermore, not only the rhythm and pitch structure were formalized, but attempts were made to formalize certain other parameters, concerning more general structure of the piece.

The development of the general structure of "Amazonia"was determined by two factors: 1) its instrumentation; and 2) changes of rhythmic componenents of the texture. Given in advance were two groups of three instruments each: winds (flute, oboe and clarinet) and percussion ("skin", "wood" and "metal"). (NB. In case of percussion each instrument consists in fact of the whole group of instruments.) Deciding which of six instruments was playing at every particular moment of the piece was partially left to random, according to the following algorithm, shown in Figure 1.

In this algorithm the whole length of the piece, approximately thirteen minutes, was divided, for purely technical reasons, into segments with the average length AVE = 27 seconds; the actual length of every segment was calculated from Gauss distribution (This was done instead of using regular steps of given length.). Each of these segments was assigned a pair of numbers (WIND,PERC), denoting the number of wind, or percussion, instruments, occuring in this segment. These numbers were chosen randomly (step 2 in the algorithm) , with probability distribution given by a table (Figure 2). Particular values in this table were based on musical ideas and preferences--note predominance of combinations "one wind instrument and percussion" and "percussion only" over others and avoidance of combinations with two or three winds and percussion. Step 3 in the algorithm is a restraint condition put on the pairs: to avoid sudden changes of instrumentation, no pair "winds only" could be followed by "percussion only" or vice versa. If the restraint

CH1721-0/81/0000/0015$00.75

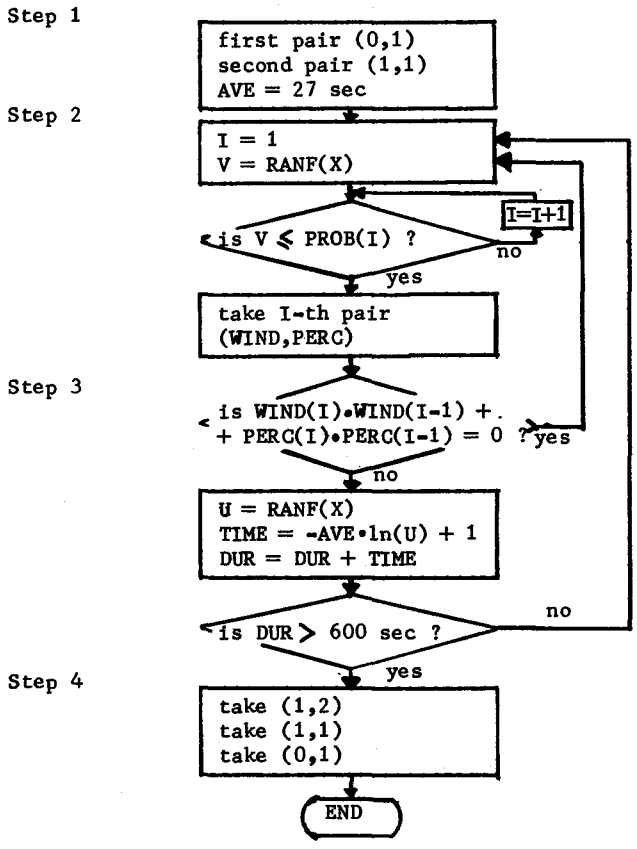


Figure 1

I-th pair (WIND,PERC)	PROB(I-th pair)
(0,0)	.7 %
(0,1)	8.3 %
(0,2)	10.4 %
(0,3)	12.5 %
(1,0)	2.0 %
(1,1)	10.4 %
(1,2)	8.3 %
(1,3)	6.2 %
(2,0)	6.2 %
(2,1)	6.8 %
(2,2)	3.5 %
(2,3)	2.0 %
(3,0)	8.1 %
(3,1)	6.2 %
(3,2)	4.1 %
(3,3)	4.1 %

Figure 2

condition was satisfied, the next pair was being chosen. The first and last steps of the algorithm again include arbitrary decisions: arbitrary choice of the first two and the last three pairs, based on purely musical ideas.

Listings obtaines with this algorithm gave number of instruments only, without specifying what instruments they are. This next step - deciding what should "two wind instruments" or "three percussion instruments" be specifically - was made "by hand", on the basis of some musical ideas. The general idea here was to preserve maximal continuity in orchestration and to avoid sudden changes of it. In general, the same instrument occured in several consecutive segments, as long as the corresponding number of instruments in pairs (WIND,PERC) was different from zero. As a result, a diagram of instrumentation was obtained. During further work on the piece, however, several arbitrary corrections were made, not depending on any formal principle. The final version of this diagram is shown in Figure 3.

Another factor determining the general shape of the piece was the development of the rhythmic structure. Two parameters of the rhythmic structure were taken into account: density and (degree of) randomness. "Density" is the number of attacks per time unit; "degree of randomness" is a new notion and more words of explanation are here needed (see Myhill[1], where this notion--as "degree of periodicity"--was introduced and described for the first time). The general idea underlying this notion is that of control of deviations of rhythmical values from an average value. If an average duration of notes is given, a small degree of randomness will result in small deviations from the average of actual durations; a greater degree of randomness will result in considerable differences among actual durations. For randomness equal to zero, all dura-

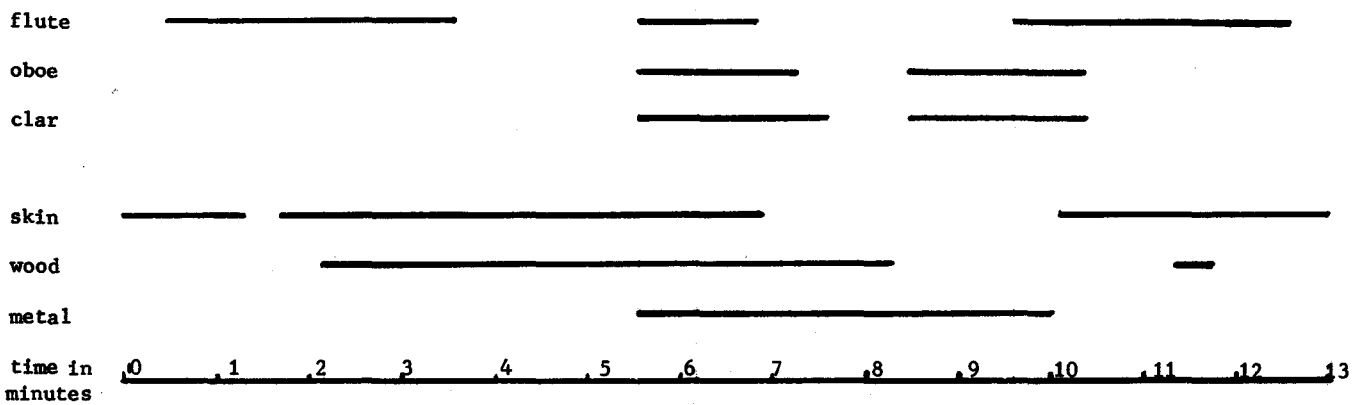


Figure 3

tions are actually equal to the given average and a regular, periodic rhythmic pattern is obtained. On the other hand, randomness equal to five or six gives already an impression of completely random choice of durations from among all possible durations.

In "Amazonia" not only changes of these parameters themselves were used, but various combinations of these changes as well, e.g., simultaneous increasing of density (or randomness) in one instrument and decreasing of the same parameter in another one. For calculations of particular values of density and randomness the following functions were used:

```
     flute (first section):
DEN1 = 2.25 * (T/60.)**2. + 0.25
DEN2 = 3.0 * (T-60.)**2./5000. - 3.0 * (T-60.)/50.
                                          + 2.5
RAN = 2. * EXP(T*ALOG(2.)/180.)

     flute (middle section):
DEN = 1.8 * SIN(PI*(T+25.)/30.) + 2.
RAN = 2.

     oboe (middle section):
DEN = 1.8 * SIN(PI*(T+15.)/30.) + 2.
RAN = 2.

     clarinet (middle section):
DEN = 1.8 * SIN(PI*(T+45.)/30.) + 2.
RAN = 2.

     oboe (last section):
DEN = 4./(1.+7.*(T/75.-1.)**2.)
RAN = 5. - 5./(1.+49.*(T/75.-1.)**2.)

     clarinet (last section):
DEN = 4./(1.+7.*(T/75.-1.)**2.)
RAN = 5. - 5./(1.+49.*(T/75.-1.)**2.)

     flute (last section):
DEN = 4./(1.+(2.*31.**2./25.)*(155.-T)**2./155.**2.)
RAN = 6. * EXP(-T*ALOG(12.)/180.)

     skin (from 0 to 83 sec):
DEN = EXP(T*ALOG(3.)/83.)
RAN = EXP(T*ALOG(3.)/83.)

     skin (from 102 to 204 sec):
DEN = 3.
RAN = (T/102.) + 2.

     skin (from 204 to 336 sec):
DEN = 3.
RAN = -3. * SIN(PI*T/33.) + 3.

     skin (from 336 to 416 sec):
DEN = 2.8 * SIN(PI*(T+55.)/30.) + 3.
RAN = 2.4

     skin (from 610 to 780 sec):
DEN = 6. * EXP(-T*ALOG(7.)/90.) *
                    * (0.8*SIN(PI*(2.*T+45.)/30.)+1.)
RAN = 3. * EXP(-T*ALOG(3.*10.**3.)/180.) *
                   * (SIN(PI*(2.*T+15.)/30.)+2.)

     wood (from 132 to 204 sec):
DEN = EXP(T*ALOG(3.)/72.)
RAN = (T/72.) + 2.

     wood (from 204 to 336 sec):
DEN = 3.
RAN = 3. * SIN(PI*T/33.) + 3.

     wood (from 336 to 416 sec):
DEN = 1.8 * SIN(PI*(T+35.)/30.) + 2.
RAN = 2.4

     wood (from 416 to 500 sec):
DEN = -(T*0.6/90.) + 1.2
RAN = -(T*1.4/90.) + 2.4

     wood (from 680 to 704 sec):
DEN = 6, * EXP(-T*ALOG(12.)/24.)
RAN = -1. * EXP(T*ALOG(10.)/24.)

     metal (from 336 to 416 sec):
DEN = 0.8 * SIN(PI*(T+15.)/30.) + 1.
RAN = 2.4

     metal (from 416 to 500 sec):
DEN = 0.6
RAN = T*2.6/90. + 2.4

     metal (from 500 to 600 sec):
DEN = -(T*0.3/100.) + 0.6
RAN = 5.
```

For given values of density DEN (or average duration AVE) and randomness RAN the actual durations were computed from the following simple algorithm (see Myhill[1]):

```
     AVE = 1./DEN
     IF (RAN.LT.0.05) 1, 2
   1 DUR = AVE
     GOTO 3
   2 ALFA = EXP(RAN)
     X = ALFA**(1./(1.-ALFA))
     Y = X/ALFA
     V = RANF(X)
     Z = Y + V*(X-Y)
     DUR = -AVE * ALOG(Z)
   3 RETURN
     END
```

In this way the whole rhythmic structure of all six instruments was determined. Since in the case of percussion, however, each instrument consisted in fact of the whole group, additional decisions were made, concerning the distribution of durations among all instruments of the group (done "by hand").

The last parameter to be determined in all details was pitch. Here, two principles were used: in sections with one woodwind instrument the horizontal structure (melodic intervals and their distribution) was of primary importance; while in sections with two or three woodwinds vertical (harmonic) structure was the deciding factor. In both cases, the basic principle used was probability distribution of certain intervals or chords, resp., and their gradual change in time.

For the alto flute part in the first section, a probability distribution with predominance of sevenths and seconds (both major and minor) was used, and a transition matrix, favoring such transitions as change of direction of intervals (up-down or vice versa), and minimalizing the probability of two intervals in the same direction, resulting in a triad (e.g., major third-minor third or vice versa). A similar type of change was used for the flute in the last section.

Sections with two and three instruments required more attention to the vertical structure. Here a few "harmonic zones" were chosen (chords FBE,

F#B♭D and FGAB) where the probability of the occurrence of the notes from the pivotal chords was the greatest, and the probability of other small or null. This probability distribution underwent gradual changes, resulting in a new harmonic zone. The probability distribution of harmonic zones is shown in Figure 4.

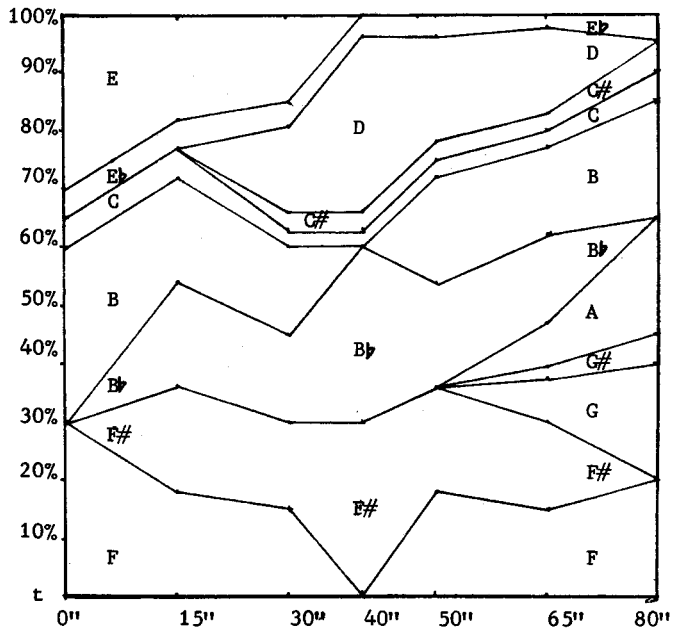


Figure 4

Other parameters: dynamics and articulation were determined for all instruments without any formal principles.

All programs for "Amazonia", based on described above algorithms, were written in FORTRAN IV Extended, and all calculations made on a CYBER 174 computer at SUNY/Buffalo.

References

1. Myhill,J., Controlled Indeterminacy. A First Step Towards a Semi-Stochastic Music Language, Computer Music Journal 3, p.12-14.
2. Xenakis,I., Formalized Music, Bloomington 1971, Indiana University Press.

MANIPULATIONS OF MUSICAL PATTERNS

by Laurie Spiegel
September, 1981

175 Duane Street, New York City, New York, 10013

A lot of attention in music system design is being given to data entry and score storage formats, timbral synthesis techniques, and to user interface refinements. Such considerations often say more about the problems of computer implementation of music than they do about problems or structures inherent in musical acitivy itself.

I've discussed in other writings the idea that music, which is all too often thought of in terms of tiny sonic entities called "notes," effectively consists of larger configurations. Such musical patterns (including chords, motifs, melodies, rhythms, meters, harmonic progressions, etc. right up to sonatas and symphonies) can be created, described, stored, encoded, orchestrated, and interpreted in a wide variety of ways. The best choice within each of these option fields depends greatly on the nature of the actual material used. This in turn depends on the nature - the structure and function - of the larger musical configuration in progress, and on how the specific materials in question are to fit into it.

That is, the ultimate form we want a composition to take, and the developmental and transformational processes we intend to use in creating it, are important design considerations in developing musical patterns to be used in a piece. (The design of a good fugue subject is a large part of the process of writing a fugue.)

In general, musical patterns of information must be designed to be recurrent, recombinant, and subjectable to selected transformations. (This doesn't mean composing is necessarily a "top down" activity. It's at least as common to come up with a motive and say "How can this be developed into a piece?" as it is to pick a process first, and then ask "Well, now that I've decided to write a fugue, what would make a good subject?")

The process of creating music involves not only the ability to design such patterns of sound, but a working knowledge of all the processes of transformation which can aesthetically be applied to them. Beyond these there needs to be a practised awareness of how such materials and operations, and the specific characteristics of each, relate to and influense each others' potentials. By which I mean both individual processes (e.g. transposition) and complex combinations (e.g. the use of transposition as part of a complex of processes such as fugue).

Making a start in this direction, then it seems like a good idea at this stage of computer music's evolution to look at plain old fashioned non-electronic music and to try to extract a basic "library" consisting of the most elemental transformations which have consistently been successfully used on musical patterns, a basic group of "tried-and-true" musical manipulations.

I do this with the following hopes:

1: That such operations may find themselves incorporated into standard compositional tools (programs) of the future, along with such already common text-editor warhorses as insertion, deletion, and global search-and-replace (this last requiring more sophisticated pattern recognition techniques for music than for word processing, of course).

2: That such a roughly drawn initial library, by its content (elemental processes) and structure (recombinable modules) could be of use in developing a more process- (versus entity-) oriented approach to computer music without abandoning principles and practices which have successfully generated music in past eras, as has sometimes happened with process-oriented systems.

3. That this might provide a slightly different model or view which may be useful in increasing our understanding of music, in developing the visual temporal art which is just beginning to evolve, and hopefully also in gaining further insight into human perception (which is, afterall, what music is designed to interface with).

4: That the description of music in terms of a transformationally oriented conceptual vocabulary will help evolve a more appropriate vocabulary and syntax for the description, understanding, and creation of experiences in time, for all self-referential temporal arts, of which music is a very pure example.

On these presumptuous notes of high aspiration, here is a starter's group of basic effective processes of musical pattern manipulation, some minimal modules of transformation, the rules for selection and combination of which, unfortunately, will have to wait.

CH1721-0/81/0000/0019$00.75

1. TRANSPOSITION

Adding an offset of fixed magnitude throughout a pattern. (In graphics this operation might be found under the name "translation.") This technique has been most noticably applied to pitch patterns in music, but can be applied to other aspects, such as amplitude, harmonic richness, or tempo. "Parallel" (simultaneous) motion of two musical "voices" (such as in medieval organum) might be described as a pattern of change set in counterpoint to its own transposition.

While repetition and delay (offset) could be described as transposition along the axis of time, these will be dealt with separately here, so as to better reflect how these sub-processes are thought of in music, and to allow the circumstances of their actual use to be better clarified.

2. REVERSAL

Along any axis, as in retrograde (temporal reversal) or inversion (usually along the pitch axis), with no change to the content, order, magnitude or proportion of the pattern's internal structure.

A distinction might be made between positional reversal (of 2 patterns, such as in invertable counterpoint or of antecedent and consequent phrases), and internal reversals, involving uniform change of direction within a single pattern (inversion of a melodic line, where each perfect fifth down becomes a perfect fifth up, etc.).

Reversal can be thought of as implying the concepts of directionality, sequencial order (linearity), and of a center point around which a reversal can occur (a pivot point).

The synchronous reversals of sub-elements or of multiple aspects of retrograded patterns (such as the reversal of the envelopes of notes within a retrograded melodic sequence) might best be described as nested or correlated (or paired or grouped) reversals, so as to distinguish among, and emphasize the independence of, musical parameters and architectural levels of patterning.

3. ROTATION

Moving something (such as an event, a location counter, one's own position...) from one end of an ordered group to the other end of the same group (in the manner of assembly language rotation instructions), or moving some unique entity through a cyclic entity. What musicians call "inversions" of a chord might better be described as rotations, as they are the movement of a unique discontinuity (an octave offset) through a cyclic group of fixed intervals.

4. PHASE OFFSET

Rotation relative to another cylcic pattern or another instance of the same pattern (for example, a canon or round). Phase can be (somewhat arbitrarily) defined as a relative (or context-dependent) realm of operation, whereas rotation, above, can be considered as an internal (self-contained) transformation (or as though against an absolute).

Different aspects (parameters) of a multi-dimensional theme (a "composite pattern," as long as we're coining terminology) may also "phase" each other. In the medieval isorhythmic motet, the pitch aspect ("color") and the rhythmic aspect ("talea") of a pattern were of different lengths, such that during the course of a piece, one of these aspects would "phase" the other (e.g. the pitch sequence might be three quarters the length of the rhythmic sequence and repeat 4 times while the rhythm would only repeat 3 times) before they end the piece together. Again, this category could be grouped with another (rotation), but is being kept separate here for reasons of musical understanding and usefulness.

5. RESCALING

Expansion or contraction of range of a set of relationships without alteration of the internal proportions. Distances are changed, but not ratios. For example, rhythmic augmentation or diminution, microtonal equal tempered scales, or playing a rhythmic pattern at a different tempo.

Reversal could be considered as a subprocess of rescaling (by a factor of -1), as scaling is really a form of multiplication. (Again, kept separate here to better reflect traditional musical perspectives.)

6. INTERPOLATION

Filling in between previously established points. Inserting a smooth ramp between discretely separated values, a fast-moving melody added over slow-moving chords, or additional chords put between given chords, embellishing with trills or other such ornamentation. The renaisance practise of "divisions playing" (improvising variations on a theme) was a method of extending shorter patterns into longer compositions by means of melodic interpolation (see also medieval trope and melisma).

7. EXTRAPOLATION

Extension beyond that which already exists in such a way as to preserve continuity with it, to project from it. This enters the realms of good intellect and/or sensative creative imagination. What is described as "free evolution" of musical material often consists largely of the performance of this operation on extant patterns.

8. FRAGMENTATION

The isolation, usually for purposes of separate manipulation, of a sub-pattern which has occurred (or "been stated") as a part of a larger configuration. (Haydn and Beethoven may be most famed for this type of "motivic development" but look at the way Bach's fugal episodes use fragments of his long fugue subjects, too.)

Generally, fragmentation has been done along the time axis, as in most examples by the above-cited composers, but it can also be applied through the separation of different parameters of a composite pattern (pitch, duration, articulation, orchestration, etc.), especially with the new conceptual freedom which electronics and mathematics provide (see below).

9. SUBSTITUTION

Of a particular within a group, a pattern within a pattern group, an event in a sequence which is other than what the listener has been led to expect (for example, a classical "deceptive cadence"), of a chord within or a melody against a chord progression, of different instrumentation in a restatement, etc. Substitutions can be made without rule or else by some orderly process, individually, or as part of a group of coordinated operations on material (an "exchange" could be described as a symmetrical or bidirectional substitution).

New technologies no longer isolate parameters of sound (or of image) from each other by method of description. Voltages and numbers represent patterns in far more general ways than staff notation (symbolic representation) or paint (the instance itself). This has made interparametric pattern substitution much easier to explore.

Note that substitution is only apparent if an original version of a pattern (or pattern group) has been sufficiently well established, either through repetition or by its striking design, to be clearly recognizable, and if enough of the original has been preserved after the substitution has been done to make the change noticable.

This brings up the important question of the extent to which the patterning in music must be consciously perceivable for it to provide the experiences humans want from music. (This is not a question with an answer.)

10. COMBINATION

Familiar terms include "mixing" and "overdubbing" and also "counterpoint" and "harmony."

The main unanswerable question for this operation is that of the degree to which each entity which is combined maintains a separately perceivable identity, as opposed to losing that individuality, becoming merged, blended with other component elements into a single unified texture (see Gestalt psychology).

It can be speculated that the power of Bach's music rests in large part on his ability to pivot on the balance point between 2 modes of perception, the older parallelistic (polyphonic, contrapuntal) and the newer group-sequential (homophonic, harmonic) mode. We have a great challenge in the creation of technological tools which could permit us to determine the balance on other perceptual axes as well, to move freely between discrete and merged perception in the domains of figure/ground, harmony/timbre, succession/continuation, et cetera.

If humans could develope sufficient self-understanding, there would be no reason why such high level powerful variables as just mentioned could not be available to composers to manipulate directly, replacing myriad weaker ones, focussing the act of composition on a much higher level than is generally practised. This would involve a changeover in the information to be specified by artists, and dealt with via our creative tools, from terminology based on the parameters of art's materials to language designed to reflect the pure processes of thought.

11. SEQUENCING

Couched in such familiar terms as "append" or "splice" and "delete" or "edit out," this is really the temporal dimension of "combination," above. Again, this process is being described separately from its more general form out of deference to musical practise. Hearing, and therefore music, seem to embody greater refinement of sensativity toward transition over time than do the visual sense and its art, which exhibit more refined sensativity toward distinctions and blendings of simultaneities.

Sequencial transitions involve the construction of paths along axes we might define as "disjunct/conjunct/overlapped", or "continuous/discrete." Whenever there is a continuous transition, its rate and curvature are highly expressive musical dimensions.

12. REPETITION

Many powerful musical forms have been based on it (canon, fugue, passacaglia, sonata, rondo, variations, strophic song, etc., etc.).

Important considerations pertinent here involve:

the balance between redundency and new information, (see information theory),

the absolute density of new information over time, including how the human ability to absorb a given density changes (see perceptual and/or cognitive psychology),

the use of repetition (listener recognition), versus continuity (listener extrapolation) in creating predictability, in leading the listener to hypothesize, to expect,

how specific material relates to listeners' pattern recognition abilities (including the ability to recognize originals after they've been put through various transformations), and the role of learning in pattern recognition ability (musical style enters here with the question of the types of patterns and manipulations with which listeners have developed the greatest facility),

the use, and composition into music, of the above-referenced ratios, to manipulate (among what other things?) expectation, emotion, physiology, consciousness, and thought.

the role of process-oriented and language-based technologies in exploring such aesthetic questions, and in providing those who wish to create with tools the syntax and variables of which are operant on the levels these questions address.

13. THE GREAT UNKNOWN

The dominant aesthetic transformational processes of the future, which could be to those of the present and past as only their own vocabulary may be able to describe, may more closely approximate or express the complex and delicate processes of the mind than those above.

They may, however, first be begin to be described in the aesthetic languages of technologies we now know.

Laurie Spiegel

New York, 1981

Controlling the Digital Synthesis Process

by G. Kevin Doren

Music Technology, Incorporated
Garden City Park, New York

Abstract

Digital music synthesis offers precise control over the synthesis process, but such control is worthless unless it is put to good use. This paper discusses methods which can be used to take advantage of this precision to create musically interesting sounds. Some of these techniques could be applied to computer-controlled analog synthesizers as well.

Introduction

Music is a process that has traditionally been completely within the analog domain, from the generation of sound waves at the instrument to their perception in the human ear.

Almost any analog process can be imitated digitally, with varying costs and benefits; sometimes the results are worth the effort, sometimes not. Digital computers and digital wristwatches offer real improvements over the analog processes they replaced. Some analog processes will probably never be replaced digitally. For example, while it is technically possible to broadcast radio and television signals digitally, the costs in terms of radio bandwidth could never be justified; a small improvement in transmission quality would be paid for by having far fewer stations.

It is technically possible to synthesize sound digitally, but what are the benefits? Are there any real advantages over analog techniques?

One potential advantage is that digital systems are better suited to large-scale integration than analog systems, so the manufacturing cost of a digital synthesizer could ultimately be lower. One manufacturer currently sells a digital synthesizer that is also a calculator for less than eighty dollars.

Digital systems also tend to be more reliable than analog systems because they are not sensitive to the drifting of component values over time or temperature.

The other major advantage is that digital techniques give extremely precise control over the synthesis process. However, this precision does us no good at all unless we know what to do with it, and since more precision costs more money, it is important to know how much precision is musically necessary or useful. This precision can even turn into a hindrance, because if the musician is forced to control too many parameters, the creation of music can become a very tedious process.

Synthesis Parameters

The most pleasing musical sounds are not boring to listen to. They vary dynamically with time in interesting ways which may range from very subtle to very obvious. They vary from one second to the next, from one note to the next, from one pitch to the next, and especially, they reflect the expressiveness of the musician.

If synthetic sounds are to have these qualities, some thought must be given as to what parameters should be controlled and how they should be controlled. Mechanisms must be provided that allow these parameters to be controlled precisely, expressively, or randomly, or even with some combination of all three.

Static Frequency Control

One of the first things to be considered when designing a system for digital music synthesis is the frequency resolution of the oscillators. At first glance, it might seem that resolution of about 1/2 Hz is sufficient, since this is the approximate frequency resolution of the human ear in the lower range of musical frequencies. This number is especially convenient because it can be achieved with a 16-bit two's complement frequency term in a system with a 32 KHz sampling rate. This is because a 32 KHz system produces frequencies from 0 to 16 KHz, and a 16-bit two's complement number gives 2**15 different magnitudes. Therefore a 16 KHz band is divided into 32,768 different frequencies, with approximately 1/2 Hz between them.

This estimate turns out to be mistaken because it does not consider the effects of slightly detuned frequencies beating with each other, preventing the instrument from being either in tune or out of tune to the desired degree. If an instrument is tuned to the equally tempered scale, the harmonics of notes played at certain intervals should beat at specific rates. With insufficient frequency resolution, the instrument can never be properly in tune.

More importantly, it is often desirable to use controlled detuning of oscillators to produce the kind of richness that occurs naturally in some

CH1721-0/81/0000/0023$00.75

instruments, or when two or more instruments are played at the same time. For example, when a key is struck on a piano, the hammer hits three strings. The resulting sound is not that of one very loud string because the three strings are not playing precisely the same pitch, and this produces beating between the component frequencies of the three sounds. If the frequency resolution of the system were 1/2 Hz, then the slowest beating that could be achieved would be 2 seconds per beat, when in reality the ear can perceive beats as long as 30 seconds. This requires frequency resolution of 1/30 Hz, or a 20-bit two's complement frequency term in a 32 KHz system.

Even though most instruments have great resolution of pitch, it is important to realize that they also have some amount of uncertainty associated with pitch. For instance, a string player will play each note out of tune by some small random amount that differs from note to note. This is imperceptable when a soloist is playing, but when a group of strings plays it creates a "chorus" effect due to the beating of frequency components. In order for a digital system to create this kind of sound, it must have the ability to detune oscillators with a controllable amount of randomness.

Our implementation allows a detuning factor to be specified for each oscillator. This can either be an absolute amount in 1/30 Hz increments, or one of five ranges of random detuning. If random detuning is chosen, a random number in the desired range is calculated at the time the oscillator is started.

It should be noted that analog synthesizers, which use a control voltage to set the frequency of a voltage controlled oscillator, achieve this detuning "for free" because their oscillators drift as the temperature changes. While at first this may appear to be a negative aspect of analog synthesis because it cannot be accurately controlled, such detuning is crucial to creating some of the most popular sounds associated with these instruments.

Dynamic Frequency Control

If a sound is to be musically interesting, it cannot be static - it must vary dynamically with time in an interesting way. A single note played on a violin by a virtuoso is much more pleasing than the same note played by a beginner, because the virtuoso plays with vibrato, the periodic variation of pitch. In trying to create such an effect electronicly, the usual approach is to vary the pitch in a very regular (sine wave) fashion. While this is better than no vibrato at all, the violinist's vibrato is much more interesting because the violinist introduces subtle fluctuations in vibrato rate and depth that are not present in the typical electronic implementation. Furthurmore, the vibrato rate and depth tend to vary inversely with each other, because the violinist increases vibrato depth by moving the string farther, which takes more time and therefore slows down the rate. Also, higher notes (violin) vibrato faster than lower ones ('cello).

In a computerized implementation, aperiodic fluctuations of this kind are relatively easy to accomplish, and they greatly enhance the quality of the sound.

There are also some less subtle frequency controls that can be very useful. These include pitch bend, which allows the musician to move the frequency of a note up and down while it is sounding, and portamento, the ability to cause a note to glide to another pitch.

These dynamic frequency controls can be accomplished in software, without any special hardware support, if an updated frequency term for each oscillator can be calculated every 10 to 15 milliseconds. When the frequencies are changed at this rate or faster, it is normally impossible to hear that they are not changing continuously. If the update rate is slower than this, the pitch will be heard to move in quantized steps during fast pitch bends and portamentos.

Envelope Control

In order to produce complex and interesting sounds, complex and interesting amplitude and frequency envelopes are required - simple attack and decay functions are insufficient. Although the computer can, in theory, control envelopes of arbitrary complexity, in practice this is not necessary for most purposes.

Our implementation provides a method for the musician to "program" amplitude and frequency envelopes for each oscillator in a simple fashion by using up to sixteen linear segments to construct a complete envelope. For example, if an exponential decay is desired, it can be approximated using a number of linear segments.

Other envelope control functions are also provided. An "acceleration factor" can be specified, which will cause the envelope to speed up by a certain rate when the key is released, simulating, say, the damper on a piano string. A "sustain point" can be selected, which will cause the envelope to stop at that point while the key remains depressed, and then continue when the key is released. A "loop point" may be specified, which will cause the envelope to repeat a certain section over and over while the key remains depressed, and then continue when it is released. Finally, there is the "retrigger point", which is like a loop point except that the looping of all oscillators is synchronized.

In our system, amplitude and frequency ramps are performed by hardware. This is particularly important in the case of amplitude ramps, because if the amplitude update rate is too slow, very fast attacks will cause a "chirp" due to discontinuities in the waveform when the amplitude is changed by too large an amount. The required update rate seems to be in the neigborhood of 100 microseconds or faster if sharp attacks are desired.

Expressiveness

The most important characterisic of any musical intrument is its ability to act as the vehicle for the artistic expression of the musician, and the computer should assist in this process whenever

possible. Most electronic keyboard instruments use their keys as simple switches, and repeated depressions of the same key always produce the same sound. In order to make the insrument more expressive, it is possible, using relatively simple hardware, to measure the velocity with which the keys are struck. This information can then be used to control one or more synthesis parameters, the most obvious being the amplitude of the note. However, most musical instruments change not just in amplitude but also in timbre as they are played more loudly, an example being the saxophone, which gets "raunchier" when blown harder. Percussive instruments such as the xylophone or piano produce sound for a longer period of time when struck harder. It is therefore desirable to develop a generalized technique for using key velocity to control many parameters of the sound in a simple way.

One such method, devised by Hal Alles [2], is to specify each sound twice, once to create the sound that will be produced for minimum key velocity and again for the sound that will be produced for maximum key velocity. We call these sounds the lower and upper bounds, and they are specified by programming the envelopes, both amplitude and frequency, for each oscillator used in making the sound. If the key is struck with some intermediate velocity, the resultant sound is arrived at by interpolating between the two specified sounds based on the key velocity. For example, if the lower bound was a mellow sound and the upper bound was a bright sound, then key velocity would control brightness. If the lower bound were a flute sound and the upper bound were a horn sound, then increasing the key velocity would gradually change the sound from a flute to a horn. It should be understood that a medium velocity in this case would not produce the sound of a horn and flute mixed together; it would produce a sound that is an interpolation between the two, possibly sounding completely different from either bound. There are 32 possible interpolations, including the two bounds.

In order to give the musician some control over the way in which the key velocity affects which interpolation is chosen, two controls were added: center and sensitivity. The sensitivity control specifies how a change in key velocity will change the interpolation; if the sensitivity is all the way off, all key velocities will produce the same interpolation, which is selected by the center control. If the sensitivity is gradually increased, small changes in key velocity will produce gradually increasing changes in the interpolation that is selected. If the sensitivity is all the way on, all key velocities below a certain point (selected by the center control) will produce the lower bound, and all key velocities above that point will produce the upper bound.

In our implementation, only timbre is normally controlled in this way. Velocity-controlled amplitude is such an often-used feature that it is accomplished through a separate mechanism, with its own center and sensitivity controls.

Variations Based On Pitch

Envelopes. Many instruments change their envelope characteristics as they change pitch, one example being the piano, where the higher notes decay much faster than the lower ones. In order to achieve such effects with relative ease, a mechanism was adopted which allows the timbre interpolation to be skewed toward either bound by some amount according to key number.

Consider the case of a piano-like sound, where high notes or notes struck lightly have short decays, and low notes or notes struck hard have long decays. First, the lower bound envelope is specified with a short decay and the upper bound envelope is specified with a long decay. This gives the note a velocity-controlled decay that is independent of pitch. Then, a special mode is entered whereby a skew factor can be entered for the various keys; this factor would be a large positive number for the lowest notes, changing gradually to zero for the middle notes, and a large negative number for the highest notes.

If a low note is struck lightly, this would normally produce the lower bound sound with the short decay, but since the low note has a large positive skew factor associated with it, the interpolation that is actually heard is closer to the upper bound, and therefore has a long decay.

Conversely, if a high note is struck hard, this would normally produce a long decay, but the negative skew factor specified for this note will cause the interpolation to be more toward the lower bound, and the sound will therefore have a shorter decay.

This approach to changing the interpolation based upon the pitch has the advantage of being fairly straightforward to implement using tables to hold the skew factors, and it is easily integrated into the envelope interpolation scheme described above.

There is one significant disadvantage, however, which is that the skew factors, if large, limit the number of different interpolations that can be selected using key velocity. For instance, in the previous example where low notes had a large positive skew factor, if a low note is hit with any velocity it will always have a long decay, although since velocity controlled amplitude is done with a different mechanism, the key velocity will still affect the loudness in the usual fashion.

Loudness. Many instruments are louder at some pitches than at others. In order to provide this effect, the software provides an equalization mode, whereby an equalization factor is specified for the various keys. This equalization factor serves to boost or cut the overall amplitude of the note.

Timbre. In addition to the equalization scheme described above, it is often desireable to control the amplitude of an individual oscillator based on pitch. If the oscillator is producing a sine wave, then the effect is the same as filtering the output of the oscillator. A mechanism was provided for specifying various "filters", which are actually merely tables containing the amounts by

which an oscillator should be boosted or cut at certain frequencies. These "filters" can then be assigned to the desired oscillators.

Using this system, it is possible to make the relative amplitudes of the oscillators, and therefore the timbre, vary according to pitch.

Conclusions

Using the techniques described previously, two products were developed. The first one, the General Development System (GDS), was designed to give the musician complete control of all synthesis parameters, for the purpose of developing various voices and software concepts.

The second product, the Synergy, is more commercially oriented, and can produce any voice designed on the GDS, but the musician is not given the control necessary to create new sounds "from scratch". Instead, the instrument is supplied with preset voices, which can be augmented by inserting a ROM cartridge containing additional voices, similar to the cartridges available for use in video games. The musician does, however, have the ability to alter many of the parameters of the sound, such as vibrato rate and depth, portamento rate and quantization, stereo channel assignment, and amplitude and timbre center and sensitivity.

Acknowledgements

This project could not have been completed without the efforts of many skillful individuals. Hal Alles was responsible for the conception and architecture; Jerry Kaplan for software implementation; Mercer Stockell for hardware and software support; Tom Piggott for musical guidance; Chris Scafidi for mechanical engineering; John Strawn for technical writing; Peter Nye for software support; Jeff Johnson, Jan Mattox, and Chris Chafe for voicing. The author was primarily responsible for hardware implementation.

References

[1] H. G. Alles, "An Inexpensive Digital Sound Synthesizer", Computer Music Journal, vol. 3, no. 3, Sept. 1979

[2] H. G. Alles, "Music Synthesis Using Real Time Digital Techniques", Proceedings of the IEEE, vol 68, no. 4, April 1980

[3] S. J. Kaplan, "Developing A Commercial Digital Sound Synthesizer", to appear in Computer Music Journal, vol. 5, no. 3, Fall 1981

[4] For an interesting historical perspective on music systhesis, see: Alan Douglas, "Electrical Synthesis of Musical Tones", parts 1-3, Electronic Engineering, July-Sept. 1953

DELAYED PLAYBACK MUSIC SYNTHESIS USING SMALL COMPUTERS

by Hal Chamberlin

Micro Technology Unlimited
Box 12106, Raleigh, North Carolina

ABSTRACT

Professional computer music synthesis generally involves the use of mainframe or large minicomputers in conjunction with sophisticated programs, such as MUSIC-IV and MUSIC-360, that make no attempt at real-time synthesis. Before the vast capabilities of these programs can be realized on inexpensive microcomputers, it is necessary to develop techniques for digital audio playback from standard floppy disk storage. A study of the physical characteristics of 8 inch double-density floppy disks reveals that long term sustained data transfer rates of 39 to 45K bytes per second are possible with specialized programming. Optimum usage of this data rate results in single channel audio playback through a 12 bit linear DAC with block gain control and a sample rate of 25 to 30KHz. An experimental system, which is now into its second generation, has been developed to prove the practicality of floppy disk based delayed playback synthesis on a 6502 microprocessor based computer.

Introduction

Until recently, music synthesis with small computers has been limited to one of two methods. One involves interfacing the computer to a conventional analog synthesizer or adding in self-contained analog or digital synthesizer boards. The other method uses highly efficient software to drive a digital-to-analog converter (DAC) with computed sound waveforms in real-time. The former is limited primarily by the flexibility built into the synthesizer hardware while the latter is limited by the speed available in standard small computers. Both have proven adequate for a variety of musical applications and both will continue to improve in utility as hardware costs (for synthesizers) decline and microprocessor speeds (for DAC systems) increase.

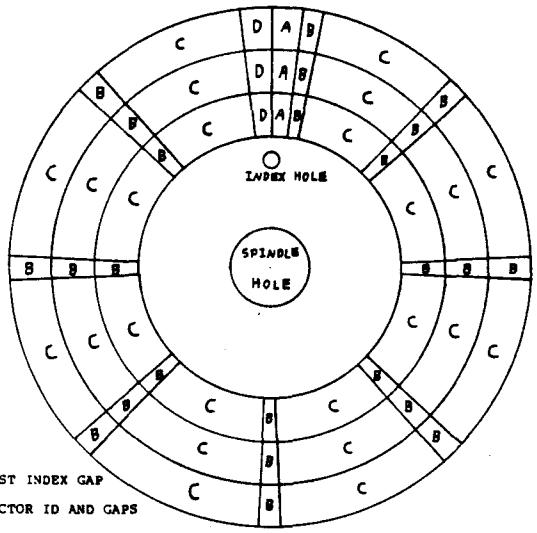
Historically, serious computer music performance and research has used a third method in conjunction with mainframes or large minicomputers. It most resembles the software driven DAC except that the waveform samples are stored on a mass storage device as they are computed and are then later played back at high speed through the DAC. In such a system, computation time between samples is of much less concern and therefore far more complex synthesis algorithms and musical scores may be accomodated. There is in fact no theorectical limit to the sound complexity (and quality) possible but there are of course practical limits. The main feature one looses when using such a delayed playback synthesis system is real-time (or nearly so) synthesis of the sound. Because of this these systems are not suitable for some applications (such as live performance) but are quite feasible for most programmed performance applications.

The primary hurdle to be cleared in implementing a delayed playback music synthesis system on a small computer is obtaining a low cost mass storage system with the requisite speed and capacity. A related obstacle is finding the required high resolution DAC with the necessary audio post-processing circuitry (deglitcher and lowpass filter). However, once these hardware requirements have been met, the full power of the technique becomes available and the remaining effort becomes purely a matter of software. The topic of this paper then is how these hardware needs have been met using standard floppy disk storage in an experimental system that should become commercially available shortly.

High Speed Data Transfer from Floppy Disks

The data transfer rate from a double-density 8 inch floppy disk is usually quoted as being 62.5K bytes per second. Yet most small computer disk operating systems achieve average transfer rates of only 5 to 10K bytes per second and even the very good ones only do about 20K. There are many reasons for the gap between promise and reality but if high fidelity digital audio playback from floppy disks is desired, it will be necessary to identify and overcome them. I will be talking exclusively about double-density 8 inch floppy disks here because the theorectical speed of 5 inch and standard density 8 inch disks is at most 1/2 that of double-density 8 inchers and therefore insufficient for high fidelity applications. Nevertheless the techniques that will be discussed are appropriate for getting the most out of 5 inch disks as well.

CH1721-0/81/0000/0027$00.75



BYTES/SECTOR	SECTORS/TRACK	BYTES/TRACK	AVERAGE TRANSFER RATE/TRACK
256	26	6,656	39,396
512	15	7,680	46,080
1,024	8	8,192	49,152

FIGURE 1. Overhead in Floppy Disk Format

Figure 1 shows a representation of data on a floppy diskette. For clarity only 3 tracks are shown but an 8 inch diskette usually has 77. As can be seen, a substantial amount of each track is taken up with sector ID fields and various kinds of gaps. The pre- and post-index gaps are fixed in size and are present mainly to take up slack caused by differences in disk rotation speed and index sensor location. The proportion of space taken by gaps between sectors however is a strong function of the number of sectors the track is divided into. Figure 1 also lists some standard sector sizes and useful data capacities per track when using them. Note that the capacity per sector is always a power of two. This is a limitation found in virtually all disk controllers and therefore using fewer than 8 sectors per track will not increase data capacity per track further.

At this point it is easy to calculate the data transfer rate averaged over a single track assuming the track can be read completely in one revolution. Since all 8 inch disk drives rotate at 360 RPM, a single revolution takes about 166.7 milliseconds (the tolerance is less than 1% for modern drives). The average transfer rate then is simply the track capacity divided by .1667 which yields the results shown in figure 1. While these figures are substantially below the promise of 62.5K per second, they are still respectable. They are also way above typical disk operating system performance so there must be other factors at work.

After a track has been read, it is necessary to move the head to the next track before it too can be read. Here again disk system specifications imply that this can be done more quickly than is the actual case. Quoted track-to-track seek speeds range from 3 to 10 milliseconds but unfortunately the head jiggles around considerably when movement stops. With many controllers, the head is also lifted while moving so time must be allowed for it to relower as well. The result is that about 20 to 40 milliseconds must elapse before data can be reliably read again. After realizing that the disk continues to rotate during the movement time, it is obvious that the first sector of the next track has already rotated past the head when the system is able to read again. Typically this would mean waiting for the sector to come around again before reading could resume. The net result is two diskette revolutions per track read and a long-term transfer rate 1/2 the figures mentioned earlier!

Fortunately most modern disk controllers look at the sector ID fields to identify sectors rather than counting from the index hole which means that the sector numbering sequence need not start right after the index hole. It is therefore possible to "stagger" the numbering from track to track so that reading of the next track can commence immediately after the head has moved and settled. Figure 2 shows 5 tracks which have been staggered by two sectors. This means that two "sector times" are allowed for the head to move and settle which in the case of 8 sectors per track is about 38 milliseconds. A little study of the figure will reveal that now only 5 disk revolutions are needed to read 4 tracks rather than the 8 needed earlier. The whole disk average transfer rates then are 4/5 of the figure 1 values or about 38.4K bytes per second for 1K byte sectors. A really fast settling drive may work with a 1 sector stagger and yield a 42.6K average rate.

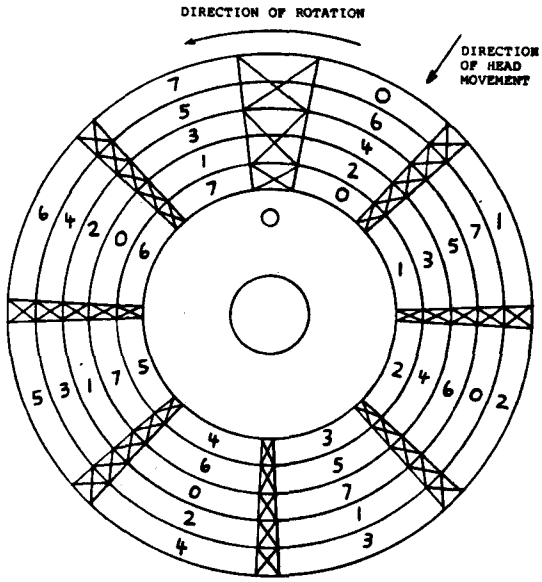


FIGURE 2. Sector Sequence for a Skew of 2

With double-sided drives and diskettes slightly higher average transfer rates are possible. Unlike head movement, head switching is instantaneous so both tracks in a "cylinder" can be read in two revolutions without staggering between sides. Stagger from cylinder to cylinder is still needed as before however. With two sector stagger, 42.6K bytes per second is possible while with one sector stagger, up to 45K may be possible.

Why then are disk operating systems up to an order of magnitude slower than the figures just calculated? Probably the main reason is that they can not read full tracks in a single revolution. Since a general purpose DOS is expected to perform random access, automatic space allocation, record blocking/deblocking and other functions, it takes much longer than the time between sectors to process the data within a sector. In a digital audio playback situation (also recording as will be seen later), none of these features are needed; the disk really just needs to act like a tape drive. Thus a custom routine for high speed disk data transfer is indicated.

Long Term Data Transfer from Floppy Disks

With the 38K to 45K bytes per second transfer rate we have been discussing, a single floppy diskette does not last very long. With 8 sectors of 1K each per track, a single-sided disk holds about 630K bytes while a double-sided disk holds about 1.26M bytes. At an average transfer rate of 37.5K bytes per second (the speed at which the experimental system runs), a single-sided disk will provide about 17 seconds of sound while a double-sided one gives 34 seconds. This actually is not bad and in fact is quite adequate for many potential applications in research, musical instrument analysis, and advertising jingles. For sound of unlimited duration one would want to be able to utilize two disk drives and switch back and forth while changing diskettes.

One factor that complicates writing a program for unlimited audio playback from floppy disk is the need for a large data buffer. Data transfer from the disk is in the form of 62.5KB/S bursts separated by gaps during which no data is transferred at all. The audio DAC (or ADC) on the other hand cannot tolerate even brief interruptions in its data flow; if interruptions are present, severe pops are heard in the reproduced sound. The first-in-first-out (FIFO) memory buffer required to smooth the data flow is much larger than might be expected.

The reason a large buffer is needed can be understood by considering what happens when the data rate to the DAC is slightly less than the average data rate from the disk. As time passes, the buffer will gradually fill up because data is going in faster than it is being withdrawn. Eventually the buffer will fill completely which means that reading must pause until the buffer empties some. However even a brief pause means that the next sector will be missed and a full revolution must elapse before there is another opportunity to read it. In the meantime the DAC continues to withdraw data from the buffer which must be large enough so it doesn't run dry during the wait. It follows then that a memory buffer at least as large as a track (8K) is needed when the DAC data rate is nearly as fast as the disk data rate. If the DAC rate is considerably less than this, the buffer can be smaller.

The buffer must be larger yet if a disk swapping scheme is used for unlimited playback duration. When switching from one drive to another, there is a random delay of as much as a full revolution before sector 0 of the first track comes under the head. This delay is due to the fact that rotation of the two disk drives is not synchronized. If this 1/6 second delay occurrs immediately after the revolution slip described in the previous paragraph, there can be a period of as much as 1/3 second during which only one sector is read into the buffer. The net result is a minimum buffer size of 15K bytes unless the DAC data rate is substantially less than the theoretical maximum. A side benefit of a large buffer is that it may often be possible to attempt a reread after a soft error provided the buffer is nearly full at the time.

The particular hardware implementation of the disk controller and digital-to-analog converter makes a big difference in the programming complexity necessary to simultaneously manage the disk, memory buffer, and DAC data flow. In fact, unless the computer is very fast, it is nearly impossible to perform these functions unless either the disk controller or the DAC is a direct memory access (DMA) device.

Nearly all disk controllers capable of handling 8 inch double-density disks use either DMA or an on-board sector buffer since the disk data rate itself (as little as 13uS between bytes) is usually too fast for direct programmed I/O. Controllers using on-board sector buffers are not suitable however unless the buffer is at least big enough to hold two sectors at once. If the capacity is only one sector, then it will be impossible to read consecutive sectors because the time needed to copy the buffer content into memory will exceed the gap time between sectors. There is at least one controller on the market however with a 16K on-board buffer which is large enough to serve as the FIFO buffer as well. With a true DMA disk controller, it is quite easy to manage a very large FIFO buffer in memory. In addition, The DAC will be able to utilize simple programmed I/O through a parallel I/O port as its interface method. Note however that DMA disk controllers must use either cycle steal or transparent DMA techniques. The old block DMA method where the CPU is halted during the entire sector transfer would leave 10MS gaps in the DAC data flow and too little CPU time between sectors to get anything accomplished.

There is one last complication in using disk swapping to attain unlimited playback durations. After a disk drive has been used to read a disk from track 0 through track 76, its head remains at the track 76 position. Before the same drive can

be used to read another disk (presumably after the other drive has exhausted its disk), its head must be repositioned back to track 0. This is not a trivial matter unless the memory buffer is very large since a 76 track seek requires from 230MS to 760MS depending on drive performance.

The logical thing to do is step the idle drive towards 0 a few tracks at a time while the other drive is reading a track. Unfortunately very few disk controllers allow stepping on one drive while simultaneously reading from another. Many controllers can step two or more drives at once however so one could instruct the idle drive to step one track towards 0 at the same time the active drive is stepping toward 76 for the next track to be read.

Even this won't work in all cases since some controllers cannot step a drive unless a disk is inserted and the door is closed. One solution is to sequence the data on pairwise alternate disks backwards, i.e., from track 76 towards track zero which eliminates any need to reseek to track 0. Another is to wait until the active disk is half read (and the idle disk has presumably been changed) and then step the idle drive two tracks toward zero for every one step of the active drive. A third possibility is to ground the disk Ready line so that stepping is possible while the door is open.

Returning to the subject of disk errors, the question is "What should be done when a read error does occur and the buffer is not full enough to attempt a re-read?" Actually, one would hope that transient read errors would not occur. Reading disk drive manufacturer's specifications would lead one to that conclusion since the usual quoted figure is "one in 10^9" bits which figures to one transient error per 125M bytes or about one per hour of sound playback. But they are also quick to point out that the promised figure "does not include media or data separator characteristics". As a result, the one-per-hour error rate is probably the best that can be expected and typical values are likely to be much worse. Thus top quality diskettes and a disk controller using an analog phase-locked-loop data separator (as opposed to a digital PLL or counter separators which have higher error rates due to time quantization effects) are preferred for sound playback applications.

There are actually three ways of dealing with a read error when retry is not possible. First, the sector could simply be skipped and the next one read into the buffer. This action would typically introduce an audible click, not unlike that of a surface defect on a record. It also would disturb the rhythm slightly since about 25 milliseconds of sound would be skipped over. Another possibility would be to repeat the previous sector's data in the buffer which would preserve rhythm without making the click any worse (on the average). Finally, whatever data was read could be used which means that the error is simply ignored. Besides a small risk that the error could be in a control field within the sector, a large proportion of transient errors are such that no data is read at all. In the latter case, the likely result is that sound data 300 to 400 milliseconds old would be left in that portion of the buffer which could cause a very annoying glitch. Thus the second method of error disposition is probably preferred.

The Audio D-to-A Converter

In the preceeding section the maximum sustained data transfer rate for a floppy disk system was determined to be between 38K and 45K bytes per second. This is actually somewhat less than what would be ideally desirable. For example, a professional stereophonic digital audio system may operate at speeds as high as 180K bytes per second based on two channels of 16 bit D-to-A conversion at a sample rate of 45KHz. For a workable system using floppy disk storage it is necessary to cut back somewhat from such state- of-the-art cost-is-no-object professional standards.

The most obvious cutback would be to single channel (monaural) operation which cuts the data rate in half. Reducing the DAC wordlength to 12 bits would make the system signal-to-noise ratio (about 72dB unweighted) comparable to conventional tape which after all will be the final recipient of the synthesis results. Reducing the sample rate to 30KHz then would bring the data rate down to 45KB/S while a 25KHz rate could be accomodated by a slower single-sided disk system. Before concluding that these compromises are too much to bear, let us consider the effect of each one individually.

If one is setting out to synthesize computer music for use in a concert situation or sale to the public on records, a stereophonic final result is certainly desirable. Standard practice in the recording industry however is to defer concern over spatial aspects of the recording until the end of the production sequence, i.e., the studio mixdown. The same concept is applicable here in that each channel could be played from disk separately and recorded on one track of the master tape. The computer and many modern "quartz locked" reel-to-reel tape recorders are quite accurate enough to do this without any special synchronization means.

Reduction to 12 bit samples merely brings the noise performance of the system down from the clouds to something comparable with most recording equipment. To achieve the theorectical 72dB S/N ratio of a 12 bit system however requires fairly careful attention to the signal levels being synthesized. The addition of a programmable gain control to the 12 bit DAC can greatly reduce this problem however while extending the dynamic range (but not S/N ratio) to that achieved with 16 bit samples.

Sample rate reduction to the 25KHz to 30KHz range is the most difficult to explain away. Using these rates yields a high frequency cutoff in the range of 11KHz to about 13.5KHz which seems far from hi-fi standards. Consider however that 13.5KHz is only two half-steps (1/6 octave) below the 15KHz legal cutoff of "high fidelity" FM radio broadcasting. Unless one has a very expensive turntable with a very light tracking tonearm,

records will loose their highs above 12KHz or so after only a few playings. The very popularity of hi-fi cassette tape, which really doesn't have any usable response above 12KHz, is a testament to the fact that most people would rather trade that last half-octave of response for convenience and lower cost. Digital audio at 25KHz actually sounds very good because the response really is flat right up to the cutoff point. Also there is no high frequency overload effect which is a perennial problem with other recording media.

Besides the usual deglitcher, anit-slewing circuit, and sharp cutoff low-pass filter, a 12 bit DAC used for audio playback needs at least a single level buffer register. The reason is that only a few dozen nanoseconds of jitter in the time between DAC updates can lead to a jitter distortion level that is greater than 12 bit quantization noise. The buffer in conjunction with a crystal controlled sample clock will smooth out the somewhat erratic data flow from the computer. Actually a short first-in-first-out buffer of perhaps 256 samples is even better, particularly if the DAC is connected to the computer through a standard parallel interface. With the short FIFO, data flow to the DAC can be interrupted for 2 or 3 milliseconds while the playback program attends to the disks between sectors. It should be noted that if a programmable gain control has been added to the DAC that the gain control bits must also be delivered through the FIFO, not through a separate register.

The Experimental System

The initial feasibility study and experimental delayed playback synthesis system was implemented on a 6502 based KIM-1 computer (manufactured by MOS Technology now a divison of Commodore International) equipped with a Micro Technology Unlimited disk controller and memory expansion. The disk drives were two Qume DataTrack/8 double-sided units. The 12 bit DAC with 256 sample internal FIFO buffer was constructed in a separate enclosure and interfaced through two 8 bit parallel I/O ports provided by the memory expansion board. The DAC itself was a hybrid module (DAC-349-12 from Hybrid Systems) to which was added an 8 step attenuator (3dB per step), a deglitcher made with an LM398 sample-and--hold module, and a 6 pole 0.5dB Chebyshev low-pass filter with cutoff at 10 KHz (end of flat response region, not -3dB point). The FIFO was made with conventional TTL logic and 256x4 bit MOS RAM's. The sample rate was determined by a crystal oscillator and programmable divider which pulled samples from the FIFO buffer at a constant rate when enabled. FIFO status back to the computer indicated how full it was in steps of 1/16 of its capacity.

This "first try" system worked satisfactorily but there were a few problems. The worst was that an LM398 makes a poor deglitcher; it introduced almost as much signal-dependent glitch energy as it removed from the DAC thus leading to excess distortion. Another difficulty was actually writing a sample playback program that successfully juggled the disk, large memory FIFO, and data transfer to the DAC all at once. The initial program worked but became unreliable when the data rate approached the theorectical maximum. As a result, the initial system was run at a 20KHz sample rate instead of the intended 25KHz.

The second generation experimental system is currently running on an MTU-130 computer (also made by Micro Technology Unlimited) using only standard features available with that computer except for the wire-wrapped 12 bit audio DAC board. Since the audio board is intended to become a product eventually, the original design was completely revamped for manufacturability plus A-to-D conversion, selectable filter cutoff, and stereo playback features were added. The interface method was changed from parallel port to direct system bus connection which greatly simplified the on- board FIFO buffer (the DAC sample clock could be synchronized with the CPU clock and thus eliminate FIFO buffer contention) and also simplified programming since access to internal registers is immediate.

Figure 3 shows a functional block diagram of the new 12 bit audio system. The three diagrams shown actually represent 3 possible operating modes of the board. Since these modes are achieved by reconfiguring the board's functional blocks, only one is available at a time. Mode switching is under software control however so it is possible to change modes quickly.

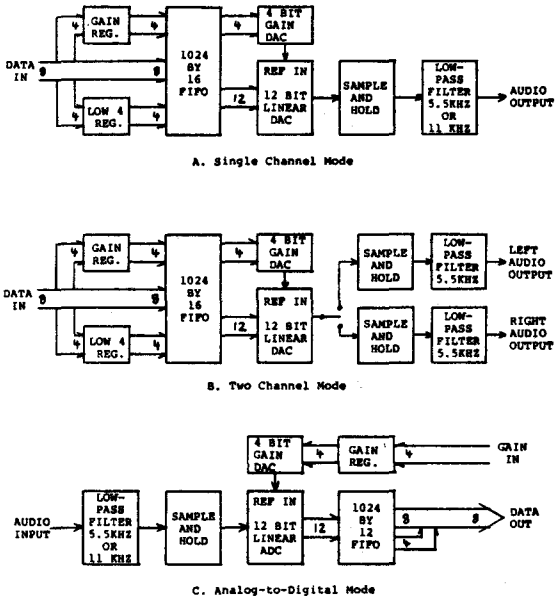


FIGURE 3. Experimental 12 Bit Audio System Board

Probably the most significant improvement over the previous unit is inclusion of the analog-to-digital conversion function shown in figure 3-C. Thus it becomes possible to digitize live instrument or other sounds for analysis or digital modification and playback. Everything discussed previously in relation to digital-to-analog playback also applies to analog-to-digital recording

thus it is possible to digitize audio material of unlimited duration onto floppy disks as well.

The stereo playback function shown in figure 3-B was added for two reasons. First, it was relatively easy to add and there are applications for reduced sample rate (12 to 15KHz) two-channel playback. After all, this rate gives response equivalent to AM radio broadcasts and there is considerable interest in converting that medium to stereo. The second reason is that rigid disks are coming down in price and perhaps within a few years capacities practical for audio (remember, the rigid disks are not removable) will also be reasonably priced. With a rigid disk, the continuous data rates necessary for high fidelity stereo can be achieved without much trouble. When in the stereo mode, the sample values read from the FIFO buffer are fed to the single DAC which alternately switches them between two distortion reducing sample-and-hold circuits.

Among detail changes, the on-board FIFO buffer was increased to 1024 samples. This was done not out of necessity but because 256x4 bit RAMs are becoming scarce! A monolithic CMOS DAC module (Analog Devices AD7541) was used because of its lower cost and lower power consumption compared to the hybrid module previously used. The original hand-trimmed gain control network was replaced by another AD7541 which gives better performance and is much easier to calibrate. A separate register for the 4 bit gain control value was added to hold a gain value until it needed to be changed. This reduces considerably the software overhead required previously to combine the gain value with the DAC data for every sample. Since the DAC is no longer housed in a separate enclosure (it is a plug-in board), the analog circuitry is fitted with a metal shield on both sides of the board to avoid noise pickup.

Figure 4 shows the disk data record format currently used which has proven to be both flexible and efficient. The fundamental unit is the sound block which consists of exactly 256 bytes. Since this is one "page" of memory in 8 bit microcomputer systems, indexing through a sound block can be highly efficient. Four of these sound blocks fit onto one disk sector using the disk format discussed earlier.

The first byte of a sound block has two 4 bit fileds. The upper nybble is termed the "ID" field and can be used for a variety of purposes. The present software system simply puts the "volume number" of the multi-disk "sound file" there and uses a zero value to signal end-of-file. The lower nybble gives the gain control value that applies to all of the samples in the sound block. The playback program simply stores this in the DAC's gain control register until the next block is processed. The remaining 255 bytes hold exactly 170 twelve-bit samples packed as shown.

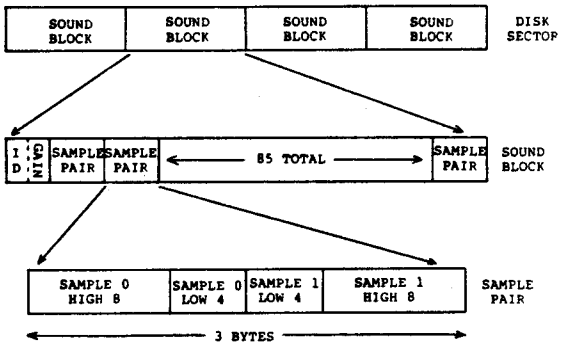


FIGURE 4. Sound File Record Format

Conclusion

The floppy disk based sound playback system that has been described has been in use for over a year and in the author's experience has proven its practicality. The companion digital synthesis program (which was not described due to lack of space) has proven to be flexible, reasonably easy to use, and actually runs faster than expected. Without any hardware arithmetic aids, compute-time/real-time ratios around 50-to-1 were realized with moderately complex music and 25KHz sample rate. The use of an AM9511 Arithmetic Processor IC in the future is expected to improve that figure at least 5-fold. The frequent disk swapping necessary during playback has not been a problem at all (provided the disks are kept in order), but swapping them while the piece is being computed can be a nuisance. The problem is that the computer requires attention every 15 minutes to an hour (depending on disk capacity and musical complexity) so one can't simply let it run overnight. As of this writing, the new audio board has not been available long enough to experiment with digital sound recording but no problems are expected.

ANALYSIS & GENERATION OF COMPLEX SOUNDS USING SMALL COMPUTERS

F.H. Covitz and A.C. Ashcraft

A B S T R A C T

A "mostly software" system is described which is capable of analyzing and generating complex sounds, including voice, musical instruments, as well as sound effects from any source. The system can run on any 65Ø2-based microcomputer such as PET, AIM, APPLE, or KIM. An audio signal is digitized and then converted into the frequency domain using a fast Fourier transform algorithm implemented in 65Ø2 machine language. The resulting Fourier coefficients are then displayed in a graphical form convenient for immediate study and for conversion into straight line segment approximations of the harmonic envelopes of complex sounds. All the sounds thus analysed and compressed become candidates for use in the authors' instrumental music synthesis program which is described in the paper.

SOUND SYNTHESIS

The digital music systhesis system to be described is a "mostly software" system which was developed by the authors for Microtechnology Unlimited and is presently commercially available from MTU along with the necessary 8-bit DAC and low-pass filter.

The software approach was based upon the waveform-table-scanning/DAC techniques described by Hal Chamberlin [1]. A program for a 65Ø2 based micro, the KIM-1, was published in the article. This program synthesized four voice music based upon repetitive scanning of an arbitrary waveform table containing 256 equally spaced amplitude values. A sampling frequency of 8.77 kHz permitted high frequency components of synthesized tones up to about 3.5 kHz. Alias frequencies above this limit (derived from the sampling frequency itself and the synthesized tones) were removed with a sharp cutoff low pass filter, also described in the BYTE article.

In the waveform-scanning method described, musical event durations and notes to be played within the event were loaded by the computer from a "song string" table previously transcribed from sheet music or created by the "composer" on the spot. Pitches were synthesized by maintaining (for each of the four voices) a double precision pointer into a pre-computed 256 byte waveform table. This allowed relatively high pitch accuracy to be obtained: approximately .13 Hz over the entire range of frequencies from 65.4Ø5 Hz (C2) up to 1Ø46.5 Hz (C6). An equally tempered scale was used.

The sounds produced by the program had rectangular amplitude envelopes with constant tone colour for the duration of the note, similar to the tones produced by an electronic organ. The program, note increment table, waveform table, and the note and duration data for a four-part arrangement of the "Star Spangled Banner" fit neatly into 1k of memory and ran in the KIM-1 !

At this point one of us (FHC) revised Chamberlin's program to run on an 8k PET. The music program itself was rewritten as a subroutine and a main program was written to execute a series of musical operations from a user constructed list called the "command string". Commands were provided which allowed:

* changes in number of active voices, 1-4
* changes in tempo
* changes in waveforms assigned to voices
* additive Fourier synthesis of complex waveforms (machine language - fast)
* scaling of existing waveforms
* execution of user defined subroutines
* playing of song segments

The increased memory space in the PET permitted several waveforms to be available at one time, so the voices could have contrasting timbres and longer song data files could be played. However, the tones were still like those of an electronic organ. If high speed multiplication hardware had been available to the authors, some increased realism of instrumental effects would have been possible by multiplication of the waveform amplitude values by a scaling factor derived from an amplitude envelope table prior to outputting amplitudes to the DAC. However, this technique would not have

CH1721-0/81/0000/0033$00.75

permitted timbre changes during the developement of the tone. Fortunately, another approach was possible which actually led to the present sound generation system.

The key to improved performance lay in the recognition that the page number of the waveform could be combined with the double precision pointer into the waveform table to give a triple precision pointer into a whole array of waveforms. The authors reasoned that if a large set of related waveforms were calculated a priori by additive Fourier synthesis and were stored in consecutive order, then a periodic increment of the page number part of the pointer could affect a gradual change in the tone colour and/or amplitude of the sounds being generated during the playing of a single note. Considering that many of the needed waveforms would be nearly identical, a waveform directory or "instrument page" approach was used, in which a full page of 256 bytes of memory was loaded with the page numbers of the waveforms needed for the desired sound effect, in the proper order, and repeated as often as was needed, for the proper time dependance of timbre and amplitude. During the playing of a note, an "instrument pointer" was incremented periodically by the program, and the waveform page number pointed to in the instrument page was stored into the highest order byte of the waveform table pointer. This approach allowed dramatic, surprisingly smooth, changes in timbre to be obtained with relatively small numbers of waveforms. Typically 6-8k of memory (24-32 waveforms) was needed to produce a smooth sounding exponential decay giving a very good impression of a percussive sound, see Fig.1. Even less memory, 2-4k, sufficed to give realistic attack, sustain, and release effects simulating simple wind instruments such as a pipe organ, or a woodwind instrument.

The overhead for maintaining the instrument table pointers was overcome by extensive reprogramming. The loop time of the play routine was reduced from 145 microseconds in the first "running" versions of the programs [2] down to 114 in the present version. This is coincidentally as fast as the original program and resulted in the same sampling rate of 8.77 kHz. Very time-efficient self-modifying code running in zero-page was necessary to achieve this. Time division multiplexing was also used in the new program, giving effectively the signal to noise ratio of a 9-bit system.

The increased capability of the sound synthesis routine was matched by more powerful command string options:

* Octave offset commands combined with a larger note increment table yielded a pitch range of eight octaves
* Absolute and relative key transposition commands were made available
* Flexible instrument page building routines allowed a variety of instrumental effects to be created from a single waveform set, giving extensive control of the attack, sustain and release phases of a note, while minimizing the total number of different waveforms required.
* Assignment of one of three different instruments (modes) to a given voice was made possible from within the note string.

Combining the latter two options gave opportunities for musical expression which we are just beginning to exploit now, eg phrasing, and dynamic effects within events.

A very crucial addition to the command string capability is the command which creates whole sets of waveforms by additive Fourier synthesis from straight line approximations to individual harmonic amplitude envelopes. Harmonics up to the 127th (although typically far fewer are used) can be entered as arbitrarily spaced waveform number/amplitude pairs. Although amplitude overflows are not flagged (but they are quite audible), the routine is self checking and self diagnostic with respect to memory constraints. Noise, with some control of its spectrum, can also be incorporated for special effects (chiff on wind instruments for example). Published data on analysis of musical instruments such as J.Moorer's contributions [3] to the Computer Music Journal could then be used directly to create the timbre changes required for realistic instruments, see Fig.2.

Unfortunately, published data are not available for all possible instruments of interest or for the entire musical range of any given instrument, nor for many other sounds which might be of interest, such as the enormous variety of speech and other naturally occuring sounds. Therefore, we sought some kind of sound analysis capability making use of the microcomputer tools already at hand.

In the following section we will describe the "mostly software" answer to our quest.

AUDIO ANALYSIS

The audio analysis program may be conveniently discussed in three separate sections. The first deals with the collection and display of the audio data, the second deals with the transformation of the data from the amplitude vs time domain into the frequency vs time domain

and the third is the user interface to allow the creation of straight line segment approximations of the original sound in a form suitable for the sound generation program.

1. DATA COLLECTION

The hardware requirements for the sound data collection are extraordinarily simple. Assuming that the main microprocessor has an 8-bit I/O port (such as a 6520 or 6522 PIA or VIA chip provides), the only other hardware needed is a moderately fast analog to digital converter (ADC. 20 microsec. or faster conversion time is adequate). In case the input to the ADC is to be a microphone, a suitable preamplifier can be used to raise the signal to the desired level. We have used a DATEL ADC-EH8B having a four microsec. conversion time in our system. Amplification of microphone signal was accomplished with a bifet operational amplifier chip configured as a follower--with-gain. The DAC hardware, used for convenience in the analysis program (ie, to listen to the analysed sounds) is the same as was used in the sound generation program.

Program Description A set of one letter commands, which in most cases may be preceded by a number, allows control of the data collection. In the following discussion, the symbol "n" refers to a numeric entry, up to 4 digits in length, which may require a sign.

nN (0<n<255) Set the sampling frequency index to n. The system default on entry is n=1. The sampling frequency is governed by software timing loops and is given by the folowing formula:

(n=0) f= 1000/43 kHz= 23.3kHz

(n>0) f= 1000/[48+7*(n-1)] kHz

Since we have chosen for convenience a time slice containing 256 samples, the fundamental frequency is f/(256). When n=0, the fundamental frequency is 90.8 Hz.

nL (0<n<255) Construct a sine wave of n periods, then scan the table cyclically at the previously selected sampling frequency, outputting the data through the DAC as long as the "L" key is depressed. This allows the user to listen to a multiple of the fundamental frequency. This is useful when analysing pitched sounds. It permits manual adjustment of the frequency of a sound source in order to pitch track (match the fundamental frequency or a multiple thereof) and permits use of a rectangular window for the FFT analysis. No other method of pitch tracking is presently incorporated into the software. When the "n" is omitted, the previous value (or n=1 as default) is used.

nT Set the audio amplitude threshold to n (0<n<256). This gives the effect of "squelch" or "VOX" hardware, i.e. digitization via the "D" command will not proceed until the sample amplitude exceeds n. The system defaults on entry to n=16.

D Digitize the sound source via the ADC into a 16k audio buffer using the previously selected sampling frequency. Wait until the threshold "T" is exceeded before digitizing.

P Play the data in the audio buffer through the DAC at the previously selected sample frequency.

B Play the audio buffer backwards through the DAC at the previously selected sampling frequency.

nW (0<n<16,064) Display consecutive waveform points in the audio buffer starting at the nth data point and using scale factor "S". Update the sample number and amplitude under the waveform cursor, see Fig.3.

+-nM Move the waveform window by n samples and display the waveform within the window, updating the status line display of sample number and amplitude. This command is similar to the "W" command where location of the window is specified relative to the current location.

+-nC (0<n<320) Move the waveform cursor +-n steps from its current position and update the status lines indicating the current sample number and its amplitude.

+-nS (0<n<7) Set the scale factor for the horizontal axis of the waveform display to 2**n. The number of samples displayed is equal to 320/(2**n). The system defaults on entry to n=0.

+-nF Construct a waveform set using straight line segment data, the amplitudes of which are pre-multiplied by (1+-n), 0<n<256, using additive Fourier synthesis. The results are stored in the audio buffer, and can therefore be listened to using the "P" or "B" commands, displayed using the "W" command or even re-analysed. The straight line segment data are normally the results

of FFT analysis, but can be loaded in or entered manually prior to running the audio analysis program.

X Exit the audio analysis program and return control to the calling system. Typically, return is to a machine language monitor.

2) ANALYSIS

R Use rectangular window during FFT analysis phase. This is useful only when the audio sample is closely pitch-tracked to the sample rate. The rectangular window should also be used when re-analyzing audio data generated by the additive Fourier routine (the "F" command).

H Use a "Hamming" window during the analysis. This is the default window on startup of the program.

nA (Ø=<n<1Ø, n is a user selected scale factor to preserve precision and avoid truncation in the 8-bit square root operation). Analyze the data in the 16k audio buffer as 64 consecutive "time slices" via the Fast Fourier Transform (FFT) algorithm, using either a Hamming or rectangular window with 256 samples per analysis (no overlap). The FFT of a given time slice of 256 samples results in the Fourier series of sine and cosine terms, up to the 128th harmonic. The amplitude of each harmonic is the square root of the sum of the squares of the sine and cosine terms (the phase is ignored). The harmonic amplitudes of each time slice are displayed in bar graph form, see Fig.4.

3) INTERPRETATION

The last operation of the audio analysis program is entered automatically after the last time slice has been transformed in the "nA" command. At the onset of this stage, the FFT results are again displayed, but at this point the display is one of amplitude vs time for the first harmonic, see Fig.5. In addition, a single-pixel graphic cursor is activated, to allow the user entry of straight line segment end points. As above, a set of single letter (or key) commands are available:

Cursor up,down,left,right - Move the pixel cursor in the indicated direction. The intent, of course, is to allow the user to "follow" manually the analysed data in a series of straight line segment approximations. However, there is really no constraint (within the limits of the cursor movement) and one could literally "imaginate" the amplitude vs. time behavior of the current harmonic.

A - Abort the current harmonic amplitude vs. time display, discarding any entered data for the current harmonic, and procede to the display of the next harmonic.

E - Enter the x,y data point at the pixel cursor and use to build the (waveform number, amplitude) straight line segment data array for thr current harmonic. An error message is displayed and the data is not entered if the pixel cursor is at or to the left of a previously entered point. As stated above, the pixel cursor doesn't have to be at or near the displayed analysis results, and thus a completely arbitrary set of straight line segment data can be created.

R - Reset the pixel cursor to Ø,Ø and discard any data previously entered for the current harmonic, but do not advance to the next harmonic. This is the only means at present to correct mistakes.

RETURN - Terminate the straight line segment data entered for the current harmonic. "RETURN" does not in itself enter any new straight line segment data; the last point must have been previously entered via the "E" command. If no or 1 x,y pair has been entered for the current harmonic, the "RETURN" performs only the function of the "R" (reset) command described above. The entered data at this point is in the form .. HN (PG,AM,...) FF , exactly as required by the sound generation program described above.

STOP - Stop the graphic entry phase by inserting a zero in the harmonic number position of the accumulated data, and return to the Data Collection Phase. The "STOP" action is automatically forced subsequent to the 79th harmonic display. Again note that the entered data is completely compatible with the requirements of the Fourier synthesis routine ("nF"), so that the waveform set described by the straight line segment data may be reconstructed, examined, listened to forward or backward at any allowed sample rate, and even re-analysed (for doubters - but please use the rectangular window).

Some Additional Comments-

Typically, the 16K audio data for sounds as complex as speech, for example, will have been "compressed" to fewer than 2ØØ bytes (much less for less complex sounds), using only moderate care in "following" the analysed data. The

synthesized sound for speech not only retains inteligibility under these conditions, but also typically retains most of the quality of the speaker's voice (but only when played forward and at the same sample rate used in the data collection phase!).

Strictly speaking, the analysis program as described, should be accurate only for monotonous sounds, i.e. the fundamental (and higher) partial frequencies should not vary with time. The utility of the analysis is not greatly impaired, however, for sounds which do not meet these requirements. For example, pitch glides and noise components are still usefully simulated by utilizing a relatively slow sampling rate. This, in effect, gives greater frequency resolution (at the expense of losing the highest frequency components). During the first part of the analysis phase, the results in this case appear as clusters of Gaussian shaped peaks, if the Hamming window was used. As an example, see figure 6. Pitch glides are then easily recognized since the center of the cluster will move toward higher or lower frequencies as the analysis proceeds. Also, noise is apparent under these conditions as relatively random amplitudes (or very broad bands if the noise has appreciable "color"); see figure 7 as an example.

Data collection, of course, is done in real time, about 1/2 seconds worth if the highest sample rate is used, and ca. 2-5 seconds for slower scan rates which still are capable of giving useful results at the higher harmonics. The FFT analysis (including graphical representation) takes ca one sec. per each 256 sample "slice". Thus a complete 64 slice analysis takes slightly over one minute. Reconstruction of waveform set via additive Fourier synthesis depends primarily on the total number of harmonics entered and the time extent covered by the waveforms; a fairly complex speech sound takes 10-20 sec. to re-synthesize. Thus, most of the time spent in a typical 10 min. session with the analysis program is spent within the straight line segments approximation portion of the analysis where the user is deciding where to position the pixel cursor for a sufficiently accurate representation.

Even gross approximations yield satisfactory results, if the objective is to get at the "essence" of the sound. There is no sharp cutoff; the closer the actual data is followed, the more closely will the synthesized sound match the input sound. In the limit, 8K(since the phase half of the analyzed data is discarded) would be required to resynthesize exactly.

It should be noted that "glitches" and low level noises can be easily seen and discriminated against by the user during the decision phase. Also, in essentially every case, the resulting synthesized sound is inherently musically usefull, since only harmonically related waveforms are involved. This holds true for essentially non-musical sound sources such as natural speech, wild even random sounds such as bubbling water.

The observant reader will have noticed the sparcity of pitch tracking capability. Pitch tracking is important for "clean" analysis. As far as the authors are aware, there is no completely satisfactory solution in either hardware or software. Simple zero-crossing detection could have been incorporated into the digitization software, but was considered not worth the effort, since it would considerably slow down the faster scanning rates, and in fact, is not an acceptable solution since many counter examples can be given. Simulation of pitch tracking subsequent to data collection by "massaging" the data in the audio buffer could probably be implemented, but would undoubtedly require a great deal of user intervention ("eyeballing" a large piece of data would probably work). In any case, the "cop-out" solution described above (adjust scan rate or sound source, listen to the pitch until they match) works well enough to be usefull.

Although there are obvious limitations on upper frequency response and duration of the sound to be analyzed, it should be noted that, in accordance with the stated objectives, the audio analysis program gives the microcomputer user a capability which formerly could be obtained only through the use of very expensive hardware oriented equipment. Since both the ADC and DAC used in this "mostly software" system are quite usefull even in non-audio applications, it should be clear that the only audio specific hardware item is the lowpass filter on the DAC output! The system described is thus extremely cost-effective.

The total memory requirement is fairly large for a microcomputer - 8k for the program, 16k for the audio buffer and ca 6k for the coefficient storage, ie a 32k RAM system is necessary.

It is outside the scope of the present article to detail the algorithms used in the analysis program. Implementation of most of the necessary routines should be possible from the description of the system. The FFT algorithm is by far the most complex used

in the current software, but this is treated in detail in the literature [4,5]. (it is still far from trivial, however, to translate such an algorithm into machine language, and far too slow to implement in Basic).

The authors are indebted to Hal Chamberlin who was kind enough to provide 6502 source listings of the FFT and several graphics algorithms.

BIBLIOGRAPHY

[1]. H.Chamberlin, "A Sampling of Techniques for Computer Performance of Music", BYTE Magazine, Sept.,1977
[2]. H.Chamberlin, "Advanced Real-Time Music Synthesis Techniques", BYTE Magazine, April, 1980
[3]. J.Moorer and J. Grey, "Lexicon of Analyzed Instruments", Computer Music Journal, Vol. II, no. 2, 1978 and prior publications in CMJ.
[4]. H.Chamberlin, "Musical Applications of Microprocessors", Hayden Book Co. inc., Rochelle Park, N.J., 1980.
[5]. T.H.Glisson, C.I.Black, and A.P.Sage, "The Digital Computation of Discrete Spectra using the Fast Fourier TRansform", IEEE Transactions on Audio And Electroacoustics, Sept., 1970.

Appendix

A satisfactory test of a particular implementation of the FFT algorithm is to check its action on some standard waveforms. The response to a pulse, which in theory yields equal amplitudes for all harmonics, is shown in Fig.8. The response to a squarewave, where the harmonic amplitudes are proportional to 1/n for only odd values of n, is shown in Fig.9.

The "leakage" effect of a rectangular window, Fig.10a, and a Hamming window in Fig.10b on non pitch-tracked sounds was tested by creating a 256 point sample of a pure sine-wave with 32.5 cycles, ie, halfway between the 32nd and 33rd harmonics.

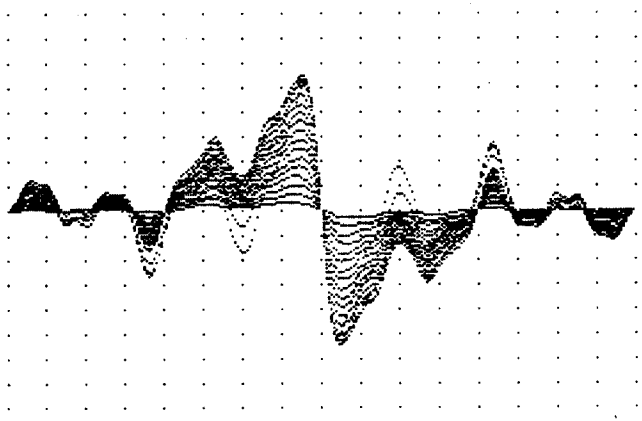


Figure 1a. Superimposed Set of 18 Waveforms for a Percussive, Piano-like Voice.

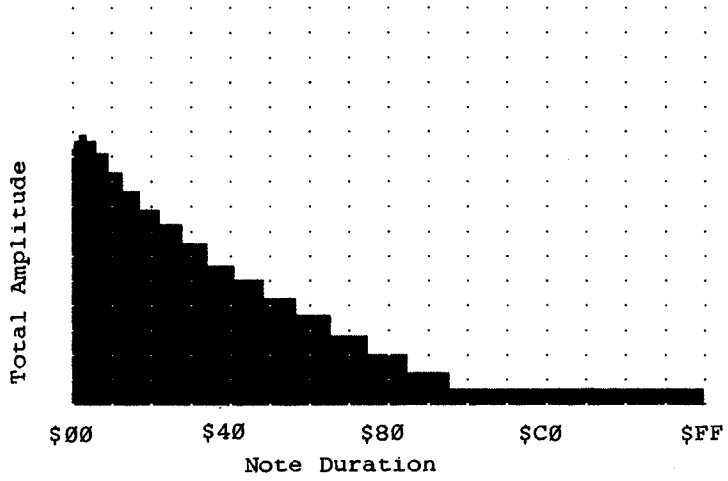


Figure 1b. An Amplitude Envelope Created Using the Waveforms of Fig. 1a.

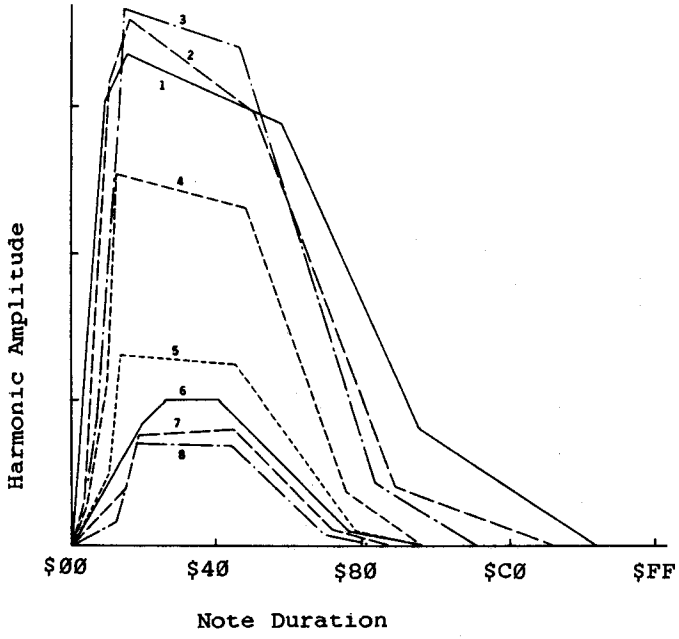


Figure 2. Straight Line Segment Approximation of Harmonic Amplitude Envelopes of an Analyzed Trumpet--See Ref.[3].

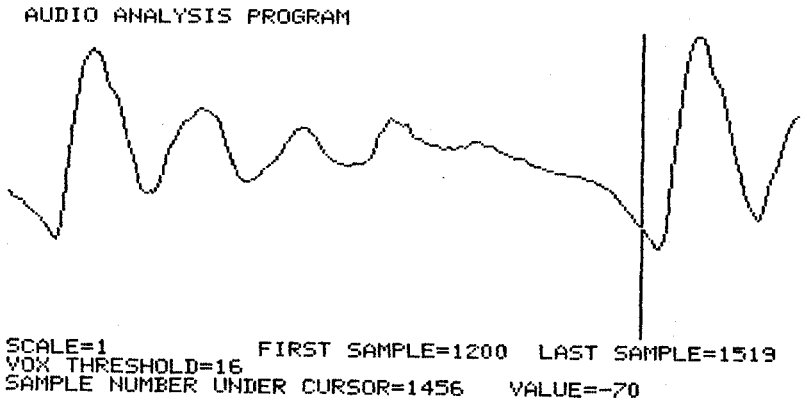


Figure 3. Waveform Display Feature. Example Shown is Part of the "E" Sound from the Word "HELLO".

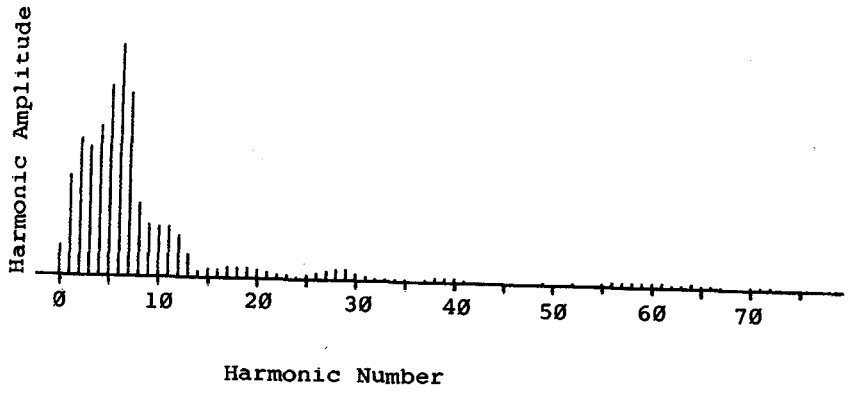


Figure 4. Set of Harmonic Amplitudes for the Waveform of Fig. 3.

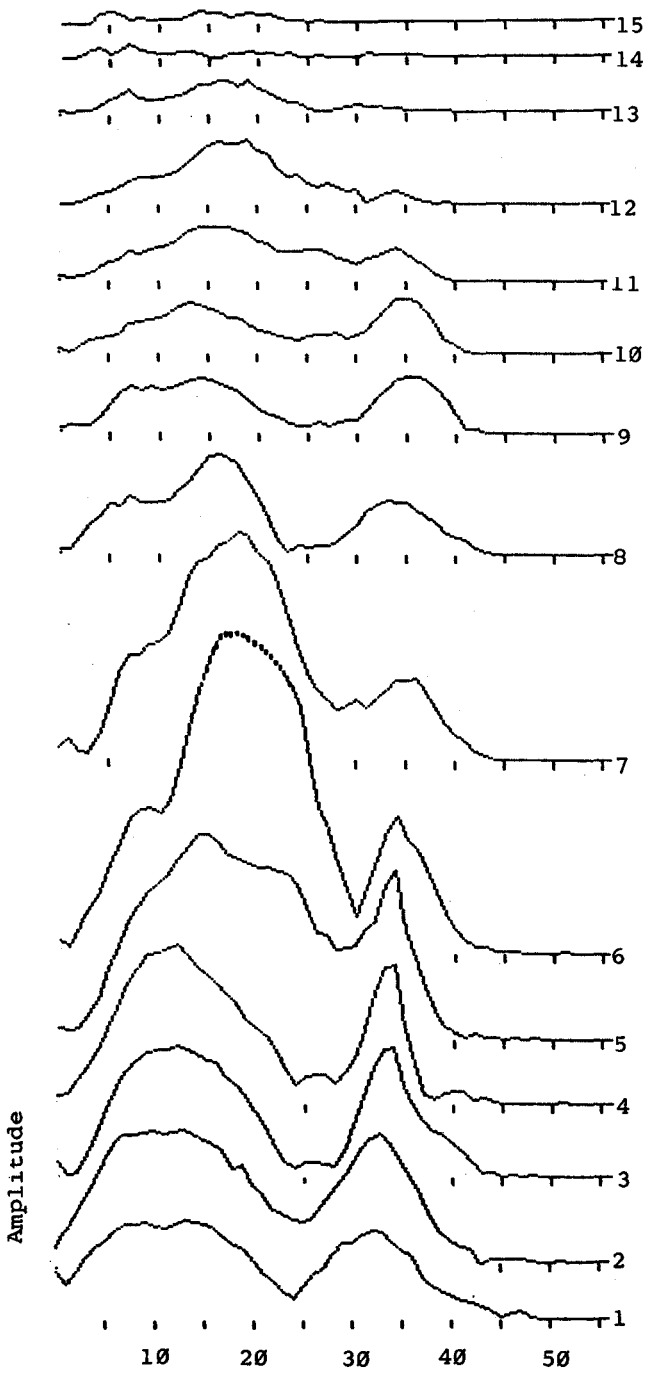


Figure 5. Amplitude Envelopes of the first 15 Harmonics of the Complete set of Waveforms from the Word "HELLO".

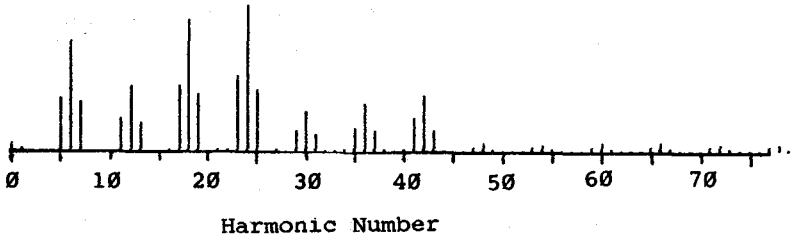


Figure 6. Set of Harmonic Amplitudes Using a Slow Scan Rate.

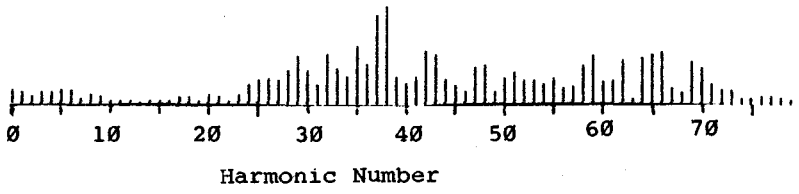


Figure 7. Set of Harmonic Amplitudes for Noise.
(The sound "shh")

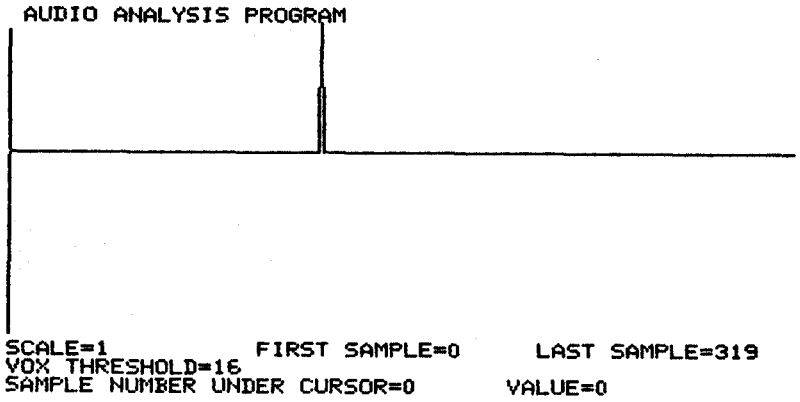


Figure 8a. Pulse Waveform.

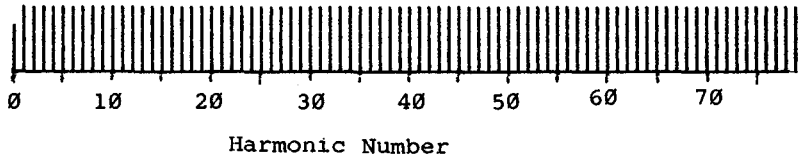


Figure 8b. FFT of 8a.

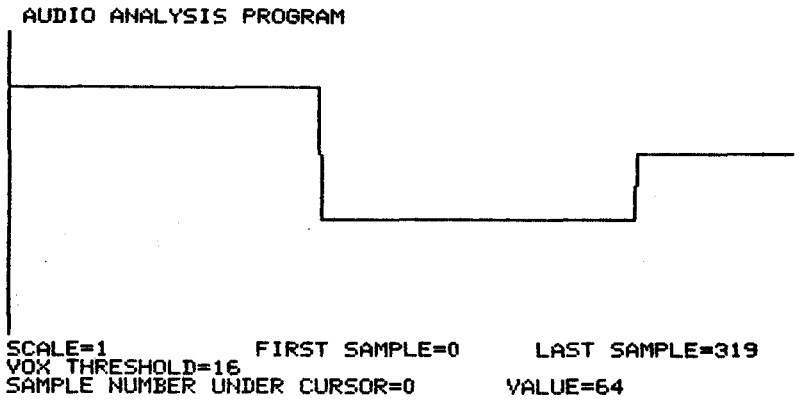


Figure 9a. Square Waveform

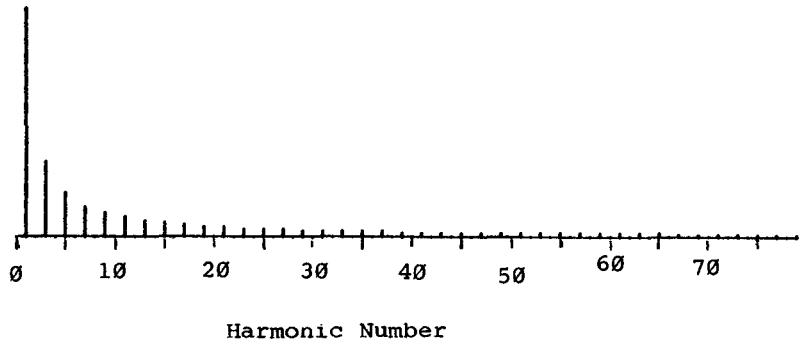


Figure 9b. FFT of 9a.

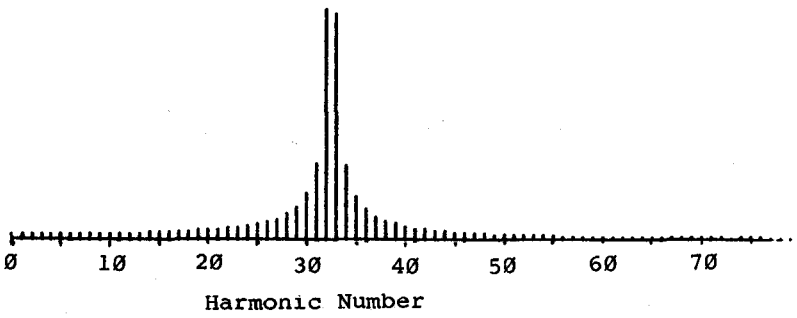


Figure 10a. FFT Analysis of 32.5 Cycles of a Sine-Wave Using a Rectangular Window.

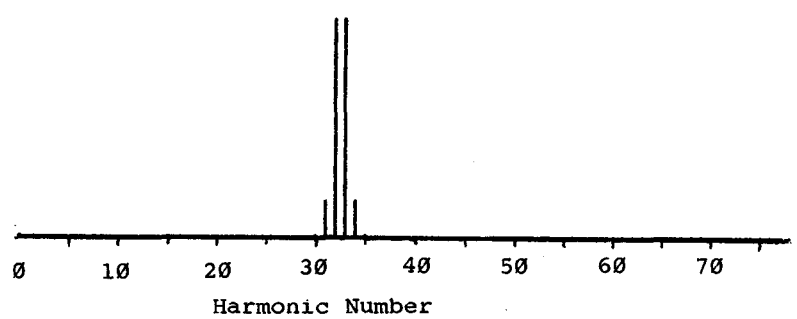


Figure 10b. FFT Analysis of 32.5 Cycles of a Sine-Wave Using a Hamming Window.

A COMPARISON OF DIGITAL SIGNAL PROCESSING CHIPS

By STEVEN LEVINE

AUDIO DATA CONSULTANTS
Box 224 Ambler, Pennsylvania 19002

Abstract:

A comparison and discussion of three digital signal processor integrated circuits is discussed. The features and applications are compared and a particular application is discussed for the NEC 7720 chip relating to digital sound synthesis.

The scope is intended to give a basic overview and to familiarize the designer of signal processing based audio applications with these new devices. References are given to allow further study of the subject.

Introduction

Digital technology is beginning to play a significant role in the economic and feature orientation of audio products. Where low manufacturing cost is the priority, the ability to fantastic new sounds is closely along side.

At present, analog techniques seem to dominate the marketplace, mainly due to their accessability and ease of design. Fully digital systems remain at the high end and are continuing to drop in price. (Although I won't mention these systems here, you will find a listing of them in the resource guide.)

A few factors are responsible for the limitations made on digital systems for audio, these are;

Sample Rate
Dynamic Range
Table Size

There are of course many more, but less critical factors which I will not mention here. (It is assumed that these factors are understood by the designer of digital audio systems and can be dealt with in the course of design.)

The three signal processors I will be discussing are:

Intel 2920 AMI S2811 NEC 7720

Although these devices are different in many ways, they are together in a new class of 'Analog Microprocessors'.

ANALOG MICROPROCESSOR PERFORMANCE TABLE

FEATURE	2920	7720	S2811
Instruction Cycle (Nsec)	400	250	300
Instruction ROM Capacity	192x24	512x23	256x17
RAM Capacity	40x25	128x16	128x16
Co-efficient ROM Capacity	none	512x13	128x16
Multiplier (parallel)	none	16x16	12x12
	(soft)	31/prod	16/prod
Multiply Time (Nsec)	----	250	300
Data Word Length(bits)	25	16	16
Assembler	yes	yes	yes
Simulator	yes	yes	no
Emulation	yes	yes	yes

It should be clear by looking at the table above that the NEC 7720 is our ideal for audio, and will be the focus of this paper. However, I will talk about the Intel 2920 and the AMI S2811, because they have some different but applicable features worth mentioning here.

The Intel 2920

There are a few features that make the 2920 interesting, mainly it's on-board data aqusition facilities. A D/A and A/D which are muxed to provide 4 in and 4 out analog lines with a resolution of 9 bits. This obviously makes a world of difference when trying to implement a signal process swiftly. There is also an EPROM which can be re-programmed many times to speed development. The 2920 has no hardware multiply, but does allow these operations to be accomplished in software. The time taken up by software arithmetic tasks reduces the overall bandwidth which is possible with this chip. The exclusion of branching functions in the instruction set achieves

extra streamlining when executing programs.

Intel has provided good software for developing applications with their signal processor. There is an Assembler, which takes mnemonic source code and converts it to binary object code for burning into EPROM. There is also a Simulator for taking input data and running it through transforms (which are to be E-prommed) to produce an output data file, which can then be plotted or converted to analog form for evaluation.

All in all, Intel has done a good job of documenting and supporting their signal processor. After evaluation, I consider it to be a good candidate for voice-band signal processing where high bandwidth (over 4 khz) is not needed. There have been several articles and a thoroughly written application handbook which contains tutorials on several signal processing fundamentals.

The S-2811

Billed by AMI as a S(ignal P(rocessing P(eripheral, the S-2811 easily interfaces to the 6800 family of microprocessors. It is addressed as 16 memory (8080 I/O locations) which control the 16 different modes it may assume.

Although not as detailed and tutorial as Intel, AMI provides good documentation in their MOS Products catalog. (Ref. 2) This includes hardware and software information. There is a good example of adigital filter using a second-order recursive IIR method, which when passed a sample, returns an output sample five instructions (1.5 microseconds) later.

The S-2811 posesses a 12x12 = 16 bit multiplier which is part of a pipelined instruction scheme that allows a compact (300 ns) Read/Modify/Write cycle. This cycle is characterized by a R(ead of a sample into the multiplier, M(odify by operating on the current or previous sample, and W(riting the results into RAM or out to a port. This results in a throughput of 3.3 multiply's per microsecond.

Mode selection by a microprocessor is simplified with the sixteen state address input. By putting the proper four-bit code on this bus, one of sixteen modes is selected. Two of these modes are worth stating here;

Mode 4 - XEQ
Mode A - XRM

XEQ allows the user to force execution of the internal rom program at a specified address. This feature simplifies multi-function control. You may also create a unique program just by pushing the program counter around a lot. This feature is nice for development purposes.

The other mode is the External PROM. This also is a nice development feature, while running slower than normal, permits de-bugging to take place prior to ROM masking.

One other thing that I think is vital to audio signal processing which is included on the S-2811 is an index register. This is 5 bits wide and is used for look-up table access. The index register becomes very handy when doing block transfers. The base register can be used with the index register to provide "double indexing" facilities used in such things as a non-linear transformation.
Due to the complexity of the S2811 diagram, I have excluded it in this paper. Please see reference 2 below.

NEC uPD 7720

Unlike the others, the NEC uPD7720 sports some fancy features that put it in a class of it's own.

Referring to the performance table, we can see that the 7720 devotes more integration to word length and storage. There is also an accent on development software.

This device does not allow any modification of it's program counter in any way other than with jumps and calls which may be conditioned upon external words which are bit-tested. It also doesn't allow external PROM Because of this, there are marked speed enhancements. NEC states that the addition of buffers onto the program ROM pointer would be difficult if not impossible. However they do have a limited number of special 'Evakits' which contain a "Romless" version of the 7720 which does run full-speed. This special chip has one hundred pins and can run programs out of high-speed RAM which is contained on the Evakit board. They rarely sell these chips (since they are very expensive to fabricate and do not yield high.) but do make the kits available to serious customers who are about to mask their program.

To make up for the limited program modification, there is extensive I/O facilities in the 7720. Refer to the architecture chart, Figure 1.

DMA - Using the DMA handshake signals DRQ, and DACK sixteen bit data can be transferred under spi (signal processing interface) control. This data is then passed to a bi-directional eight-bit port which is multiplexed on to the system bus to communicate with a DMA controller.

Parallel I/O - Through this same sixteen-bit register, normal parallel I/O with an eight or sixteen-bit microprocessor can take place.

Serial I/O - Through the use of on chip serial hardware, high speed serial communication can be accomplished between the 7720 and serial digital to analog converters, for instance. The speed of the transfer clock can be set by the user with an external clock signal. (See Figure-2)

In addition to serial and parallel I/O there are two single bit output lines which are named "P1,P2". These can be toggled in software to produce additional handshaking or control signals.

Instruction set

There are four basic instructions. They are:

Op and Rt- These two instruction types differ only in that the Rt can perform a return from interrupt or subroutine after completing it's normal operations.

They can;

Do an ALU operation on either accumulator-
Using operands from one of four different sources-
Modify the upper three and lower four bits of the coefficient ROM pointer-
Move data from one register to another
Perform simultaneous 16X16 multiply-

All in the same instruction!

Jump - Unconditional, conditional and subroutine call.

Ld - Load 16 bit data to any one of thirteen destination registers.

Audio Design Considerations

In my opnion the 7720 is more than adequate for real-time synthesis of audio signals. I have researched the requirements for an oscillator which has an arbitrary waveform, and has hardware envelope generation.

These are some of the specifications:

A sample rate of 32 khz
A sixteen bit D/A converter
DMA hardware under the supervision of a microprocessor
Frequency and Envelope specification will come from a host controller.

Using any waveform creation program, a 1024 word by 16-bit wavetable is moved into the synthesizer's system RAM. Envelope rate and slopes are also sent to this RAM, where they are picked up every five sample periods while the discrete samples are picked up every sample period and loaded into the spi. once inside they are buffered in spi internal ram until the program is ready for the new samples.
At each sample period the spi has 125 instruction cycles (of 250 ns. each) to calculate part of the envelopes and to process the current sample. This leaves enough time to compute about 5 voices worth of samples which are then summed and sent to the dac at a 32 khz rate.

A typical algorithm for such a voice might be:
Upon tick of sample clock fed to interrupt input of spi:

Get sample
Get env pair (slope, rate)
Integrate slope
Multiply wave sample by present envelope value
Output sample to DAC
Test for stop value of envelope
Process next sample

An FFT may be performed by the 7720 where factors and coefficients are judiciously allocated in data ROM 128 real points are possible in a minimum of 2.0 milliseconds.

The Future-

We can look forward to exciting advancements in the signal processing realm, made by giants like IBM, TRW, INTEL, NATIONAL etc. due to the Defense Department's "VHSIC" program. In particular, IBM has plans to fabricate first in MOS then C-MOS an acoustic signal processor for the Navy, which will inevitably become feasable for audio applications. They plan to use a 'Master Image Gate Array' technique which incorporates function blocks such as signal processors, and memories into gate arrays, ending up with a versatile semi-custom chip.

Look for more intelligent synthesis devices which will only require simple token-like commands to initiate complex audio functions. Voice analysis/synthesis combinations will undoubtedly appear on the scene as well.

This next generation of synthesis/analysis devices will bring the reality of human real-time control to a level of asthetic acceptance for digital synthesis which is long overdue.

References

Hal Chamberlain - Musical Applications of Microprocessors. Hayden Book Co.

EDN - February - May 1981. Five-part series entitled "Designers Guide to Digital Signal Processing", by Bucklen, Eldon, Schirm, and Williams, of TRW/LSI Products.

Electronics - March 1, 1979, Single chip N-Mos Microcomputer processes signals in real-time. Hoff and Townsend, Intel Corp.

Electronic Design, Feb. 15, 1980 - A signal processor fast enough to do real-time voice band. Dave Millet, Product marketing manager, NEC Microcomputers Inc.

Electronics, September 22, 1981 - VHSIC proposals take six fast tracks. John G. Posa, Solid State Editor.

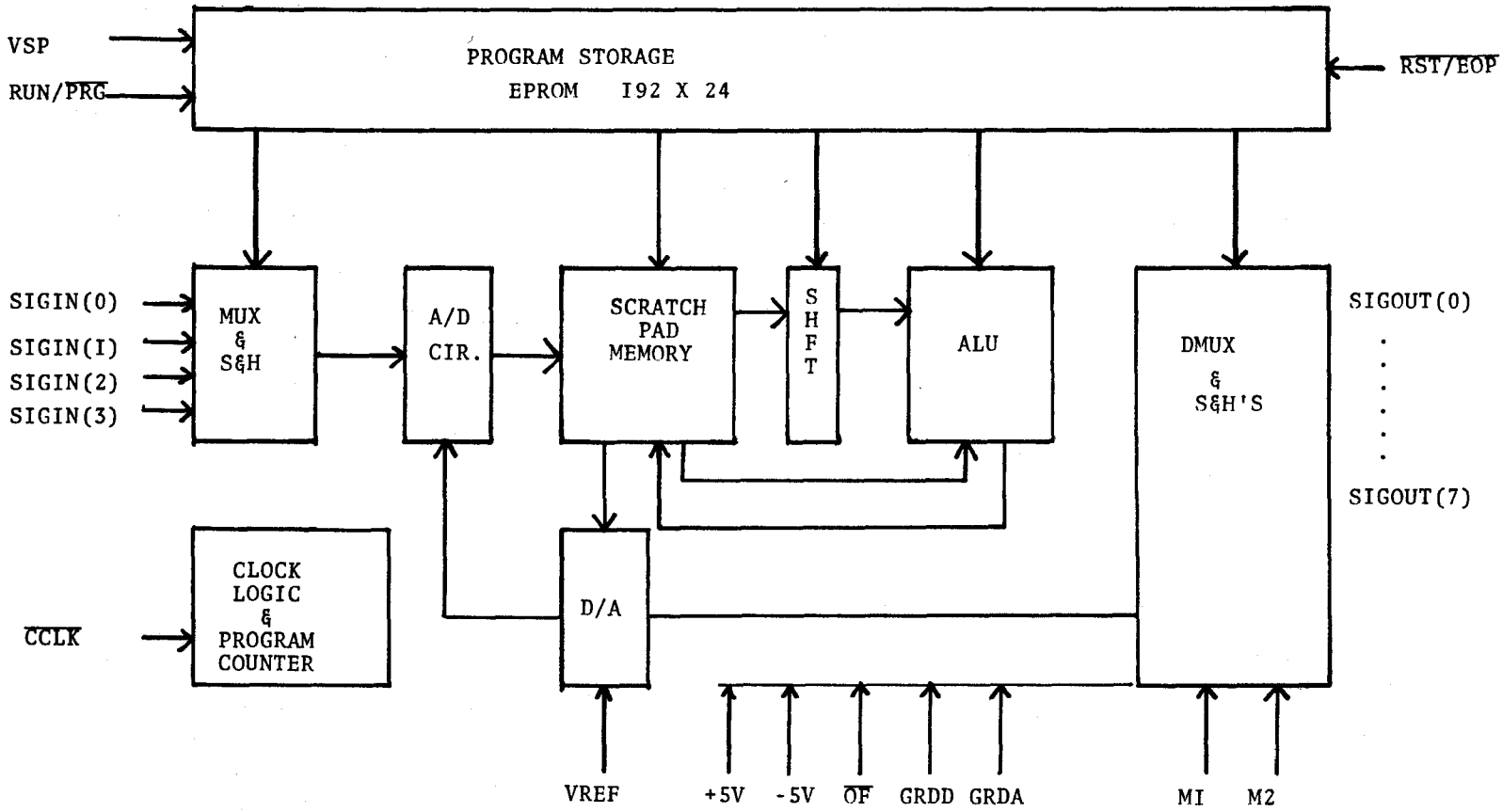


Figure 1. Intel 2920 Block Diagram

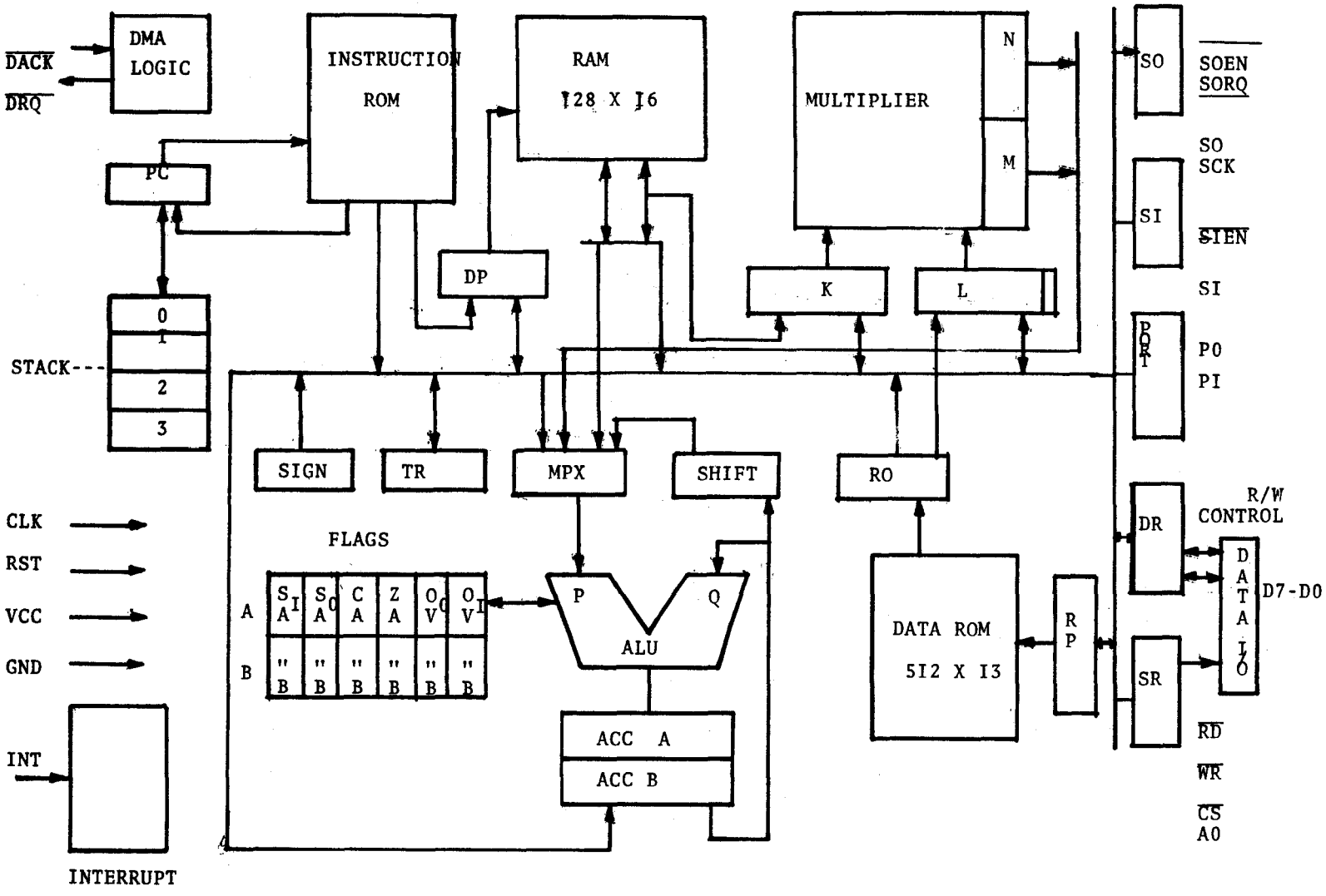


Figure 2. NEC uPD 7720 Block Diagram

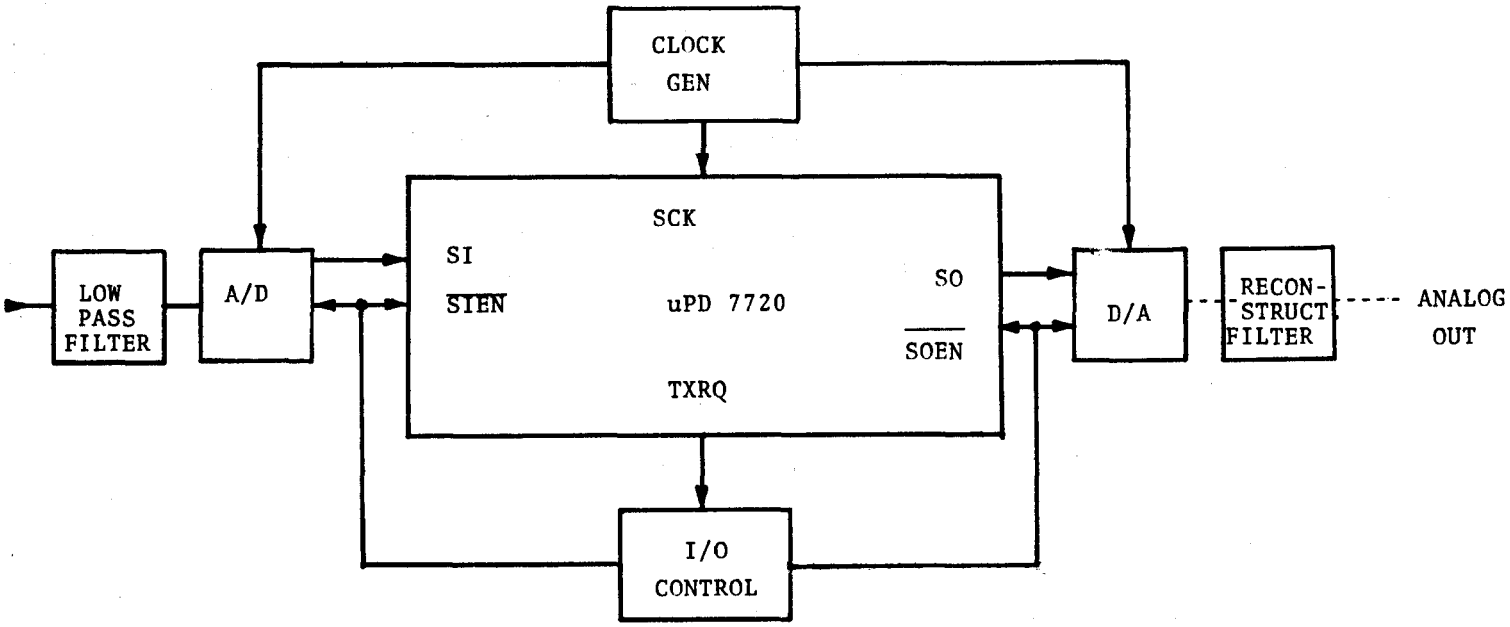


Figure 3. An analog to analog digital processing system using a single SPI

THE ALPHASYNTAURI INSTRUMENT
A MODULAR AND SOFTWARE PROGRAMMABLE DIGITAL SYNTHESIZER SYSTEM

by Robin J. Jigour, Charlie Kellner, and Ellen V.B. Lapham

Abstract

Based on an inexpensive, widely accepted microcomputer, the alphaSyntauri [tm] system provides musicians, sound engineers, educators, composers, and computer enthusiasts with a flexible digital synthesizer.

Introduction

The alphaSyntauri instrument is a modular, software programmable digital synthesizer system based on the Apple II microcomputer. The philosophy behind the alphaSyntauri design was to create a truly general purpose digital synthesizer by taking advantage of the flexibility and capabilites of a microcomputer system. By using the software programmability and expandable hardware interface, the alphaSyntauri system alleviates product obsolesence, allowing a continual upgrade path for new applications, performance features, and functions.

Background

Historically, sound synthesizing devices have been hardware based. The programming of early synthsizers was accomplished using either hardwired equipment or patch bays, such as in the original Moog synthesizer.

Dependence on circuitry and hardware has inherent limitations. First, the "programming" is not easily changed. Patch bays can become complex and on-the-road modifications require hardware expertise, time, and patience. Second, the cost of the instrument increases as a function of size and complexity - thus expansions and modifications can be expensive. For these reasons, early synthesizers were limited to large, fixed installations such as universities and studios.

Today, digital technologies are opening horizons for music exploration and control, and sound synthesis controllers range from dedicated microprocessors to large-scale general purpose computer systems. Microprogrammed synthesizers have been effectively " hard wired" at the software level: the user typically does not have access to the microprocessor to make programming and control changes.

To gain flexibility, and to take advantage of computer system technologies, designers of the Fairlight and Synclavier synthesizers both use sophisticated techniques such as disk storage and programmability to achieve a greater degree of user control than was available but a few years ago.

The alphaSyntauri, designed in 1979, uses a general purpose computer system, the Apple II, which is designed simply for information handling for a multitude of purposes. Here, by intent, the system software designer has put control in the user's hands through software-based instructions, letting the user modify (or even create from scratch) his own musical instrument.

For instance, users may additively create ten timbres bu using a built-in software module which is standard with the instrument. Then, saving these timbres to diskette is accomplished using standard Apple II disk utilites.

The computer's general purpose nature provides new and continuing espandable options to the music industry for working with synthesized sounds. From the instrument definition process to the editing of recorded pieces, the Apple II systems supports users with a mulitude of sofware tools. languages, printers, and utliities.

The alphaSyntauri system hardware

Primary system components are the keyboard (input device), software (operating system and utilites), interface hardware (between the music keyboard and the Apple II standard bus), and oscillators (sound synthesis devices available from numerous sources). Additionally, the system must include an Apple II computer with disk drive and monitor, and an audio system as illustrated in Figure 1.

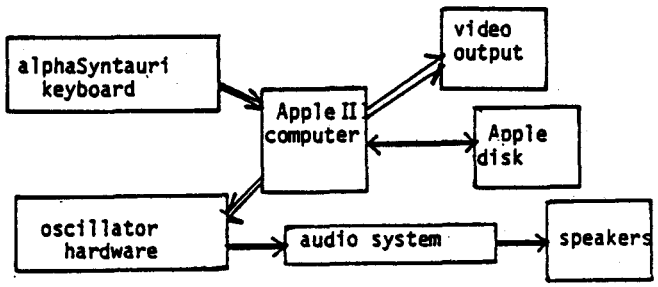


Figure 1

The alphaSyntauri instrument achieves variabliity through the user selecting the oscillator/software combination which will achieve the required muscial result. For instance, polyphonic voicing can range from as few as threee to a maximum of 15 separate voices with the square-wave-based ALF cards. Or, utilizing the Mountain Computer MusicSystem product results in an eight-voice fully waveform controllable instrument. Operating systems are tailored to get the most musical and flexible results from each oscillator set. And, because the Apple II is especially suited to adding new hardware components on the bus, new oscillators and outboard controllers are feasible and likely.

The alphaSyntauri system software

The key to the entire alphaSyntauri system is its operating system software which handles the user interface and controls the hardware. The operating system performs all the functions that a traditional hardware-based synthesizer does with the switches, knobs, patch cords, and circuitry to control envelope generators, mixers, and filters.

The bulk of the operating system software is written in 6502 assembly language to optimize speed through the instrument's hardware control processing loop of 16 milliseconds duration. The resot of the operating system, which is less speed dependent, was written in BASIC for the wase of implementation and access to standard Apple II functions.

Operating System Process

Within the alphaSyntauri system, there are multiple processes (or Tasks) performed essentially simultaneously. The computer system, and the 6502 processor in particular, are sufficientlt fast that all these tasks, and their attendant decisions, are executed without impairing the sound quality.

System tasks include reading the 61-note organ style keyboard input device, determining which key was pressed, reading tables (envelopes, waveforms, scales) and updating the output oscillators. (See the attached process flow diagram).

The oscillator update task is the most fundamental process in the alphaSyntauri system. The update tells the output hardware to produce a sound at a certain volume and frequency until the next update cycle, when the oscillator control(s) may be changed to reflect new keys pressed, new envelope stages, timbre modifications, or note terminations.

The keyboard event, the pressing of a key during live play, is the trigger for manu system control tasks. This event, interestingly enough, is treated as an exception since its detection and determination of what to do with the event information takes less than 1% of the total process loop time available.

Software controlled parameters and features

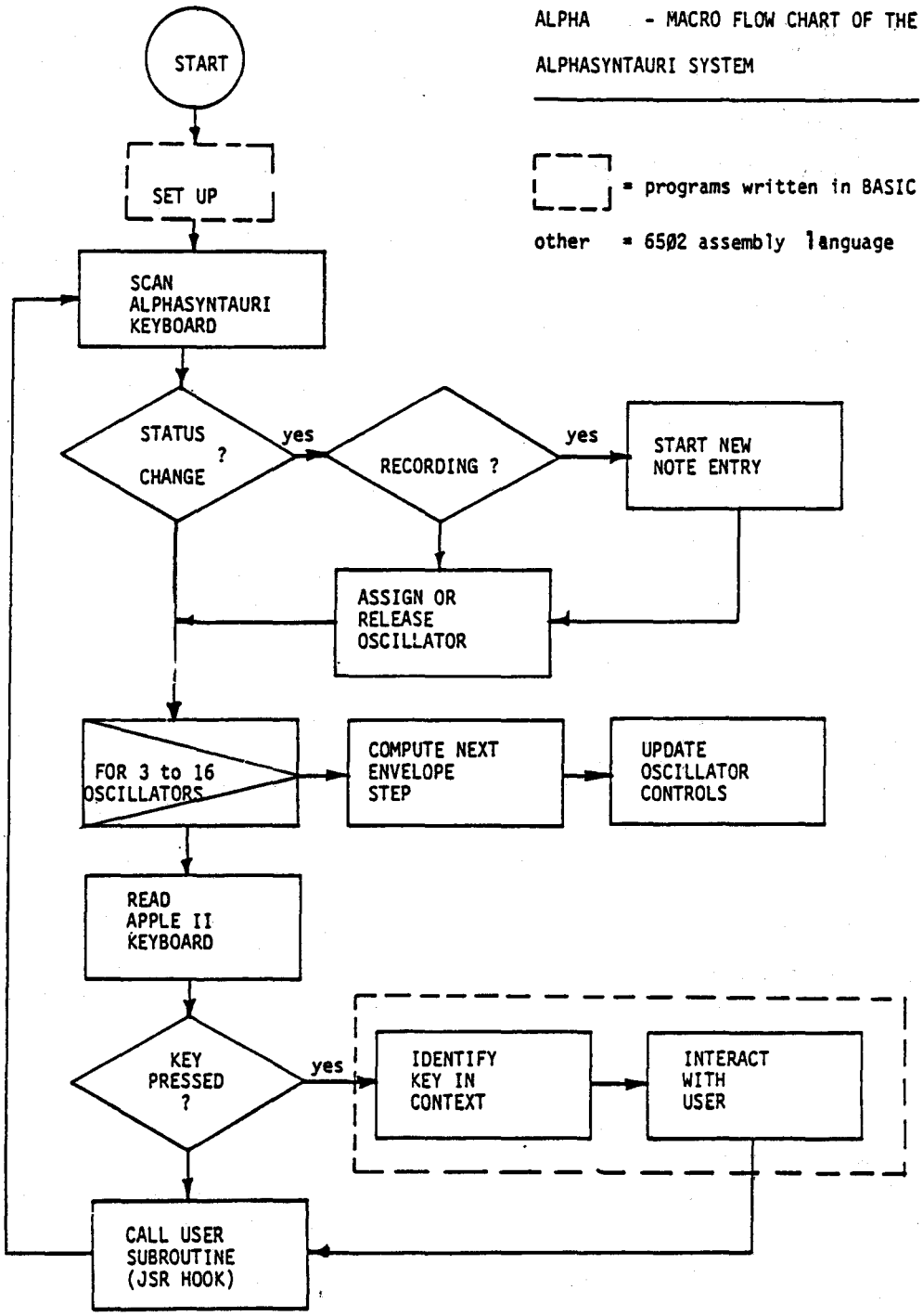
alphaSyntauri oerating system software defines the performance capabilities and features possible on the instrument, given the hardware oscillators (synthesis cards) being used. The current operating systems, alpha III and alphaPlus^{tm}, provice functions equal to and beyond those of many traditional synthesizers, including:

- polyphonic voicing
- recording and variable speed playback
- oscillator envelope control
- additive synthesis waveform definition and analysis
- disk storage and retrieval of instruments and recordings
- vibrato and tremulo
- key transposition
- definable scales
- sustain and portamento
- special effects loops
- keyboard velocity sensing
- user created program interfacing

The following examples were selected to demonstrate the range of features and control options achieveable using the general purpose microcomputer, the Apple II. Extensive use is made of the Apple's utilities, memeory handling, and disk operations to extend the alphaSyntauri instrument beyond what would have been possible using dedicated electronic components:

1. Record and variable speed playback.

Since all keyboard activity is monitored by the computer for oscillator hardware control, the computer's natural function os the logging of keyboard events as they occur. The alphaSyntauri record feature does exactly that: each event and its duration is logged into what is called a "notes file". A sequence of up to 3,000 events can be recorded into a single file (file size is limited by available RAM space for the recording buffer). Once an event eequence is completed, it can be played back digitally at speeds varying from 1 to 800% of the original recording input speed. A notes file


ALPHA - MACRO FLOW CHART OF THE ALPHASYNTAURI SYSTEM
= programs written in BASIC
other = 6502 assembly language
START
SET UP
SCAN ALPHASYNTAURI KEYBOARD
STATUS CHANGE ?
yes
RECORDING ?
yes
START NEW NOTE ENTRY
ASSIGN OR RELEASE OSCILLATOR
FOR 3 to 16 OSCILLATORS
COMPUTE NEXT ENVELOPE STEP
UPDATE OSCILLATOR CONTROLS
READ APPLE II KEYBOARD
KEY PRESSED ?
yes
IDENTIFY KEY IN CONTEXT
INTERACT WITH USER
CALL USER SUBROUTINE (JSR HOOK)

may be repeated indefinitely, beginning immediately after it completes for sequencing, for use in setting up bass lines and rhythms, Finally, notes files can be saved or loaded from diskettes using Apple II disk drives or any mass storage device plug compatible with the Apple II bus.

2. Velocity sensing

Each key on the alphaSyntauri keyboard has two electrical contacts, one of which makes about one-third of the key-down travel, the other of which makes at bottom. When a key is pressed, the following occurs:

a. The lower contact closes. A count register in the Apple II's memeory which is assigned and maintained for each key is set to zero. No other action takes place.

b. The upper contact closes, The accumulated count value in the key's timing register is used to index into the velocity assignment table. Attack rate and attack target volume (envelope) of the oscillator are determined. (Note, both attack rate and attack volume are varied. Psycho-acoustic research tells us that we perceive loudness to increase not only with absolute loudness increase, but also as a function of the rate of increase and the linearity of the curve).

c. A new oscillator is assigned for the key which has been pressed.

To achieve audible result, both attack rate and volume are used. Typically, the attack rate and volume for a given key are inversely proportional to the time between contact closures; velocity is the inverse of transistion time. Actual changes to attack rate and volume are handled by a look-up table which is automatically loaded by the operating sustem software into a pre-assigned memory location.

To achieve keyboard sensitivity, setting the loudness changes to emulate a stiff or a loose keyboard, a program value may be set by the user to make the look-up tables more or less linear. The loosest keyboard has the most linear curve, the stiffest has the most logarithmic. Thus, in the stiffest keyboard, a key has to be struck very fast and sharply to get a marked volume change.

Different velocity-sesnsing results are also possible by completely reworking the look-up table. Further, because the desing of this dynamic control parameter is general, the velocity-sensing software could be altered to affect the sustain level/ decay rate rather than the attack rate / target volume.
Additive synthesis waveform definition and analysis
Additive synthesis waveform definition and analysis

Unlike analog synthesizers which use voltage-controlled oscillators to produce a sine, triangle, sawtooth, or square waveform, the alphaSyntauri system (with Mountain Computer oscillators) constructs digital duplications of any simple or complex waveform in blocks (256 bytes by 8 bits) of memeory. A digital to analog converter (DAC) transforms the stored waveform into audio signals.

Waveforms are created in the standard alphaSyntauri operating system when users specify wave types,harmonics and amplitudes, For example, a complex wave might be created as follows: sine wave/fundamental/ 50% amplitude, square wave/ third harmonic/ 20% amplitude, sawtooth wave/ seventh harmonic/ 30% amplitude. Called " wave " in the operating system, this program takes the defined waveform parameters and creates the waveform block through additive synthesis, finally storing the waveform on diskette for later use.

Because the creation of waveform has been treated in the most general case within the system, a number of unusual approaches to sound creation are possible. The number of harmonics, for in instance, is limited only by the mathematical capabilites of the computer system itself (and, by the deign of the particular oscillator set being used). Additionalyy, the process for waveform table creation itself can be tailored to suit available knowledge and techniques within the music profession. Two utilities, " B-3 " and " Make Pulse " illustrate this point: B-3 allows the user to specify drawbar settings to exactly reproduce the human interface and the sounds of the popular Hammond B-3 organ, and Make Pulse lets users define a special class of waveforms known as pulse waves. Freehand drawing of an arbitrary waveform is also feasible: use of this technique is valid only if the user judges that the resulting sounds are useful.

Waveform analysis is also possible using the built-in " analyser " prgram of the alphaSyntauri synthesizer. The Analyzer program takes any predefined, stored waveform and results in a Fourier analysis listing of the harmonic sine components up to the twentieth and their relative amplitudes. Here, again, the limits on the analysis are imposed by the resolution of the computer system, and the programmer's decision to analyze through the first twenty harmonics.

User created program interfacing

The main process control loop (See illustration) contains what is called " an unconditional JSR " (jump to a user-written subroutine). This allows any user familiar with 6502 assembly code to devise his/her won special effects and controls.

Once written and inserted into the main process loop, this JSR routine is always accessed an and acted upon prior to any oscillator update. In keeping with the goal of producing realistic and pleasant sound effects, the time taken to perform the user's subroutine should be kept under one millisecond.

A typical JSR has been contributed by a user, composer Laurie Spiegel, who created the Pitch Sweep effects in the alphaPlus operating system. To To generate complex timbres and repeating patterns

of pitches from a single keypress, a form of digital frequency modulation is used. Envelope values already present (and accessed) within the system are AND-ed to the Apple analog controller setting to rapidly vary the pitch. The resulting sounds range from swoops to jangling, aliassing patterns; the extremes of effects are achieved precisedly because the computer-based digital system imposes few conceptual limits on musicians seeking individual musical expression.

PART II

SMALL COMPUTERS IN THE VISUAL ARTS

INTRODUCTION

IN COMPUTER CIRCLES, THE 1980'S HAVE BEEN CALLED THE DECADE OF SOFTWARE DEVELOPMENT. LOW COST MICROCOMPUTERS CONTAINING EVER INCREASING AMOUNTS OF MEMORY HAVE MADE IT POSSIBLE FOR THE GENERAL PUBLIC TO WORK WITH HIGH RESOLUTION DIGITAL IMAGES. THE BOOM IN VIDEO TECHNOLOGIES, OF VIDEODISC AND VIDEOTAPE, PLUS THE ADVENT OF TELECOMMUNICATIONS, THE ABILITY TO TRANSMIT THIS VIDEO, OPENS UP MANY DOORS FOR THE FUTURE CREATION AND EXCHANGE OF VISUAL INFORMATION.

WE FIND NOW THAT SOPHISTICATED TECHNIQUES ONCE ONLY POSSIBLE ON LARGE MAINFRAMES ARE MAKING THEIR WAY TO MICROS. IT IS NOT UNUSUAL TO SEE 3-DIMENSIONAL SIMULATIONS ON SMALL COMPUTERS.

WITH THE NUMBER OF PERSONAL COMPUTERS ON THE INCREASE, COMPUTER ART IS EMERGING FINALLY AS AN ART FORM. ITS UNIQUENESS IN HAVING AN INTELLIGENT TOOL TO WORK WITH SHAPED NOT SO MUCH BY HANDS BUT BY MIND IS A NEW EVENT.

WE REALIZE NOW THAT IT IS THE DEVELOPMENT OF SOPHISTICATED MAN-MACHINE INTERACTIVE TECHNIQUES FOR USE ON SMALL COMPUTERS THAT WILL FINALLY OPEN THE USE OF COMPUTERS FOR ALL ARTISTS.

THOUGHTS ON COMPUTER AESTHETICS AND THE FUTURE ROLE OF SMALL COMPUTERS

William J. Kolomyjec, Ph.D., M.F.A

Beginnings

Computer generated imagery possessing aesthetic quality was first accomplished by non-artists. Historically, there are three individuals who can be considered as the originators of computer aesthetics: F.Nake, G.Nees, and A.M.Noll. These individuals were promarily oriented toward engineering and science but because their imagery possessed a visual awareness, their works came to be exhibited as art. An excellent history of the medium and information on these and other significant individuals and events can be found in H.W. Franke's book, Computer Art-Computer Graphics.

In an attempt to put the process of generating computer aesthetics in perspective, several observations should be made. From its beginnings, around 1965, the actual production of any form of compute aided image was complicatied by at least three obstacles: 1) images had to be programmed in complex higher order languages on complex graphics hardware, 2) turn around time was excessive, and 3) access to the equipment was extremely limited. John Whitney Sr., made the analogy in an early film he produced entitled "Experiments in Motion Graphics" that the process of creating aesthetic works on the computer was similar to playing a piano and hvaing to wait several hours or more to hear the resultant sounds. Both the hardware and complicated programming of graphics peripherals severely inhibited the role of using the computer as an aesthetic medium.

The Aesthetic Process

The aesthetic process consists of three phases premeditation, production, and product. Figure 1 is a simplistic schematic representation of the aesthetic process. An idea is conceived or premeditated in the mind before any aesthetic image can be produced or stated. The second phase is to make, modify, record, or preserve the initial idea via some fom of aesthetic medium utilizing the craftmanship and expertise possessed by the artist. For example, a drawing is produced by the artist with pencil or ink on paper (the medium) by the hands of the artist (craftmanship). Lastly, the image is presented in final form as a visual statement. After the appropriate modification and editing, sometimes concluded with the signature of the artist, the drawing is exhibited in a presentable form. Thus, the image or artwork has become a visual statement, the result of the aesthetic process.

If Figure 1 is acceptable as a general model of the aesthetic process, then it is proposed that such a model can be used as a framework for a comparison of more traditional image making and the computer or machine enhanced image making process. Figure 2 illustrates two variations of the aesthetic process algorithm. In Figure 2, the craftmanship and expertise component of the production phase has been substituted by "hand" for the traditional variation by "machine" for the computer enhanced

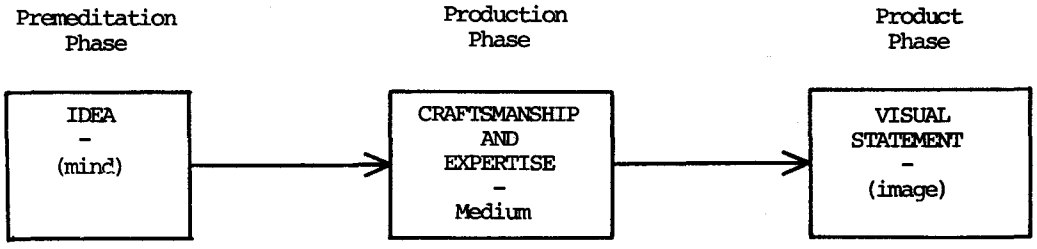


Figure 1. Diagram of the Aesthetic Process

CH1721-0/81/0000/0059$00.75

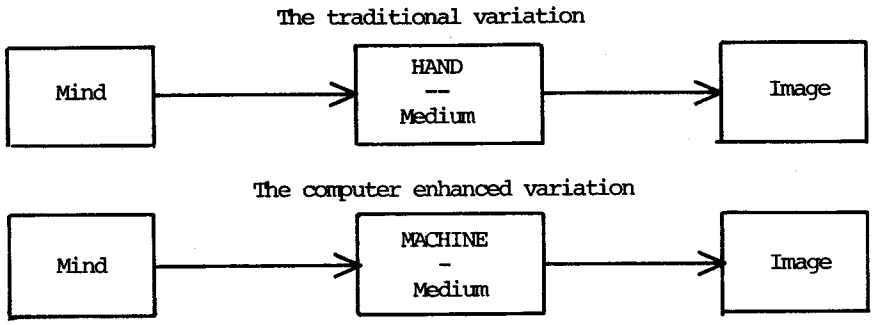


Figure 2. Variations of the Aesthetic Process

variation. These substitutions represent a major issue often used in discussions which seek to compare traditional versus computer enhanced media. Based on this presentation of the process, it it the author's position that both variations indeed represent the same process - that either the hand or the machine can be used to create aesthetic imagery.

The machine does not replace the hand, it merely modifies the aesthetic process with technology, The machine alone is not capable of producing aesthetic imagery (except by accident or chance). The machine must be directed by the mind and programmed via the hand. The machine, or more specifically the technology represented thereby, provides another dimension to the hands' capabilities. Thus, any production of a visual statement involves a process whereby the mind must be able to think in terms of the medium in conjunction with sufficient craftsmanship and expertise to produce the appropriate aesthetic image.

Attributes and Difficulties

Every aesthetic media used for individual expression has its own inherent capabilities and unique physical characteristics. An artist choosing the medium of computer graphics should develop a sensitivity to, and an awareness of, these qualities. It should go without saying that those who intend to produce the finest imagery in any media must possess a profound knowledge of the capabilites to the maximum. Computer image makers should be no different.

It is appropriate here to mention some of the attributes and difficulties of computer enhanced media. This media possesses its own capabilities and characteristics. To name a few: plotter-drawn lines exhibit a unique quality and are different than lines drawn by hand. CRT imagery has characteristics determined by the hardware, particularly through the variety of phosphors, which, in turn, can be modified via video or film to produce a multittude of effects. Computer images are extremely accurate, often to $\pm$ 0.001". Computer imagery can take advantage of repeatability (loops) to facilitate infinite variations or similarities within a single image or between images. By the same token, randomness can be employed to make trivial decisions or produce controlled variations. These and other characteristics give a unique feeling to computer aesthetic imagery. Figures Three and Four are offered as examples to illustrate the unique qulaity of computer aesthetic imagery. Both images were executed on an Apple II microcomputer. The hard copy was generated on a Houston Instrument X-Y plotter. Both images were done by this author.

Difficulties abound relative to using computers for aethetic applications and have kept many artists and other interested persons from exploring computer enhanced media. The problems which existed in the beginning, anmely the mathematics and logic requirements associated with computing, slow turn around and limited access, still prevail. It is the author's point of view that large computer facilities have stymied the artist. Since most facilities with graphics production capabilities are owned by industries, governments, or educational institutions, access to hardware is controlled and guarded. Once access is obtained, the question of cost must be reckoned with, ususally by direct or indirect billing procedures. Finally, because of a lack of graphics standards and device-dependent instructions, even though a person has programming expertise, the generation of graphics output is installation specific. Furthermore, even if graphic primitives are available, i.e. MOVE, X, Y or DRAW X,Y, rudimental graphics algorithms,i.e., draw circles or raotaiton of all or part of an iamge, are not obvious or available without proper instruction. Still, despite these overwhelming constraints a few highly motivated individuals have managed to produce computer aesthetic imagery.

The Role of Small Computers

The bright light on the horizon for computer aesthetics is the microcomputer. This writer believes that microcomputers with graphics capabilities will play a major role in promoting computer aesthetics in the future The notion of doing computer art prior to the advent of small or personal computers was indeed frustrating, depressing, and just plain scary to the artist. The capabilities of the latest generation of microcomputers have significantly reduced or eliminated the aforementioned difficulties. Microcomputers with graphics capabilities are

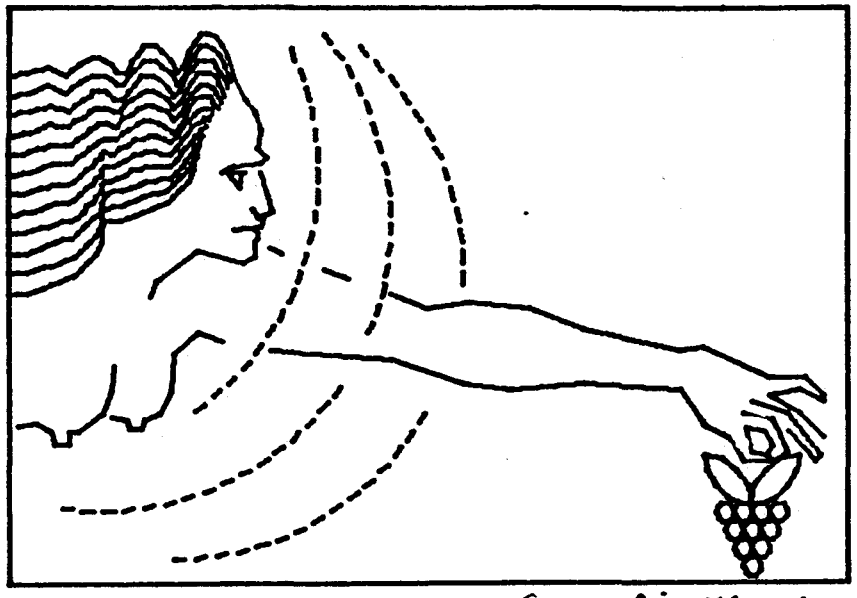


Figure 3. "Rape of the Grape" is a premeditated attempt at generating a representational image that shows the real aesthetic potential of the medium. (One of the major criticisms of computer art is that the majority of it looks like it was done by some engineer or scientist. In reality this argument has merit.) The majority of the image is digitized but several algorithmic processes are employed. The hair is formed by linear interpolation between the upper and lower curves. The grapes are formed by arcs and circles a la an arc subroutine. The aura by a dashed line variation of the arc sub.

practically turn-key image producing systems. They are affordable and easy to use. Small computer systems are not imposing and return more than a modicum of control to the aesthetic user. The tool can now fit in the artist's hand. As well as having built-in graphics primitives, they are for the most part interactive. Graphics software abounds in print in readable, personal computer periodicals. Small computers are recognized as versatile tools and have sufficient capabilites to do elaborate imagery. Artists using small computers can learn by doing at their own pace and at their own convenience.

Small computers have played, and will continue to play, a twofold role in the further establishment of computer aesthetics as a valid form of individual expression. One, small computers are a teaching and doing tool and represent a bridge between science , technology, and art. Two, small computers have finally made it possible for computers and artists to co-exist in the same realm. Small computers have enticed yound artists especially in the design areas. Graphic designers are using them for layout and typesetting. Industrial designers are using them for shape descriptions and display. Already a new breed of electronic/computer artists are exploring varigated motion graphics-color kinetic computer-driven video art. (Tom Defanti and Dan Sandin's students at the University of Illinois, Chicago Circle campus are an example). Small computers are making it possible to explore aesthetics in the artist's studio. No longer will it be necessary to make pilgrimages to a mainframe in some mecca of technology. The microcomputer with graphics capabilites has only begun to have a profound effect on computer aesthetics.

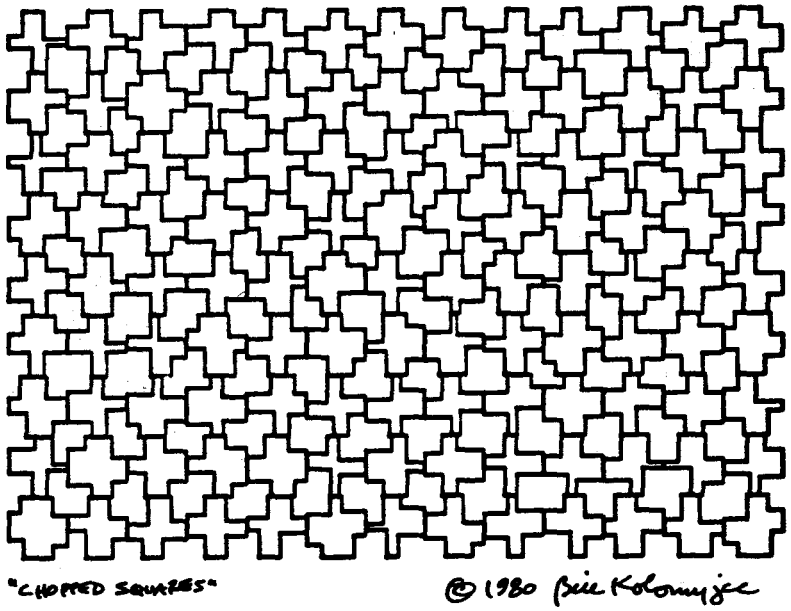


Figure 4. "Chopped Square Tessilation" emphasizes the repeatability and random aspects of the medium. The image is an array of squares which get their corners nibbled away by having the program randomly take some distance (less than half the length of each side) off each side.

References

Franke, H.W. Computer Graphics - Computer Art. Phaidon, London, 1971.

Whitney, John Sr. "Experiments in Motion Graphics" Pyramid Films.

Dietrich, F. and Molnar, Z. "Pix-Art" Computer Graphics and Art, 1980-81, Yearbook, Vol. 5.

BRAINWAVE DRAWING GAME

by Nina Sobell and Michael Trivich

The Brainwave Drawing Game utilizes Apple Computer Graphics, an electro encephalographic device, and an analog-to-digital converter to visualize the non-verbal communicative powers of one or two persons. An image based on brainwave output is created on the CRT in real time. Viewing the graphics in real time enables the participants to instant ly perceive whether they are simultaneously emitting the same brainwave or divergent waves. Each brainwave is color keyed as well as audibly keyed. Forms, shapes, movements, colors, and sounds are modulated according to the brainwave reception.

In 1970, I started using video to document the interaction of participants with sculptures that I had made and to create time lapse environments. I then began using it to document personal performances with the camera as a private audience. After discovering the potential in video to portray human interaction, I was inspired to explore more immediate means of communication. During this time I met Michael Trivich, an electronics engineer with whom I began to collaborate in cybernetics. When we began to expand the use of electronic mediums to express thought, the idea of drawing with one's brainwaves directly onto the CRT became very intriguing. In 1973, I received a grant from the Creative Artists' Public Service Program to pursue this concept. In search of access to high quality EEG equipment with professional instrumentation, I met Dr. Barry Sterman of the VA Hospital neurophysiology laboratory. He proposed that if we were able to provide quantitative proof of the existence of non-verbal influences between two people, we would be able to continue to work with their EEG equipment. Working together with Michael Trivich and the staff we discovered, after processing the EEG data through the DEC PDP-10, that these influences did indeed exist. We were then able to work at the lab, bringing in video equipment and subjects to prepare for the EEG: Video Telemetry Environment at the Contemporary Arts Museum in Houston, Texas in 1975.

> ...in an area of the Museum designed to remind one of a comfortable living room, two participants at a time sat down on a sofa, electrodes were attached to their heads, and their combined electroencephalographic patterns were transmitted into the home TV in front of them. At the same time a video camera recorded the image of their faces onto the screen. Emphasis was placed on the importance of having the home TV screen so conducive to a purely vicarious response, be the tool for allowing participants to experience the power they can have in controlling their brainwaves. By having their faces, which often mask genuine sensations and the actual state of mental agitation (as revealed by the EEG) displayed on the screen, a remarkably complete portrait of an individual at particular moments is observed...
> (quoted from an article by Laurel Siegel)

TECHNICAL DESCRIPTION. The following description by Michael Trivich is of our project to date. "The concept is straightforward: monitor brainwaves, analyze them with the computer, synthesize a video image, display it on a color TV screen. Earlier assemblages of borrowed equipment combined a Z channel electroencephalograph (hi-gain differential, bandpass limited amplifier) with an X-Y oscilloscope. Two participants were connected to X and Y inputs respectively. Any stability in the resulting Lissajou display would imply a synchronicity between the individuals participating. This pattern was superimposed on the image of the participants' and the composite was videotaped. Closing the eyes and relaxing reduces the higher

CH1721-0/81/0000/0063$00.75

frequency (Beta) activity and the lower frequencies,(Alpha, Theta, and Delta) then begin to dominate. This produces patterns of greater regularity: it is no longer a 'biofeedback' environment.The participants were provided with a remote control pause switch, which allowed them to control the videotape recorder.in front of them.

ENTER THE APPLE. The addition of a computer allows greater freedom in the transformation of electrical activity to visual images. Dick Heiser of the Computer Store (the first retail computer store in the world) where Nina had been working, provided the first computer. Herschel Toomim, founder of the Biofeedback Research Institute in Los Angeles provided the EEG. The first version connected the EEG earphone output to the audiotape input of the Apple. A commercial program entitled Kaleidoscope by Bob Bishop provided the transform. Thus far, everything had been done with borrowed equipment and limited time constraints. Nina met Chris Matthews at the Computer Store where he had come in search of the same analog-to-digital converter that we had needed. Chris had been working with the concept of an 'Alarm Clock' which would go off during a dream sequence so that the dream could be consciously interpreted; to be selective, it would have to monitor physio logical activity. The 'Brainwave Drawing' and the 'Dream Machine' began to collaborate.

Power spectrum analysis like Dr. Sterman was using with DEC PDP 10's was a bit beyond the Apple's capability but we wanted real time display. Since the emphasis was on art software development, we continued on the synthesis/display end of things. The Apple's ability to display, rotate, and translate predefined shapes is amazing on so small a machine. On the analysis side the cassette audio input is essentially a zero crossing detector that squares up the audio signal. With software loop sampling, it becomes a 1 bit re solution(sign) analog-to-digital converter. This does not provide amplitude information but it can provide some frequency inform= ation. Since the circuitry is AC coupled, net DC is zero, so on the average there will usually be as many ones as zeros. If the sample rate is greater than the highest frequency (by at least 2X, says Dr. Nyquist) we can count the number of ones in a row and this count will be inversely proportional to the frequency. This presumes a narrow power spectrum,i.e., close to a sine wave...such is life on a shoe string budget.

Development continues with emphasis on sound, image,symbol and shape manipulation influenced by the dominant frequency of the EEG input. In a parallel pursuit, the low end of brain/computer/video image technology is being explored on a Sinclair Zx-80(kit) using the EEG input to influence a cellular 'Life' model by altering the rules of 'Life' concerning birth, survival and death. Such are the life and death struggles of art and science. "

We hope to evolve the human species' innate ability to communicate non-verbally from this idea by making it visually evident in real time. Creating a closed circuit feedback loop to present the viewer their mental as well as physical image, we hope to reinforce control of one's environment. The accompanying videotape chronicles the experiences discussed.

REAL TIME INTERACTIVE COMPUTER GRAPHICS

Eric S. Podietz

Digital Mercury

Abstract

The modern computer is a powerful tool capable of processing many types of information at a rapid rate. Likewise, we humans process much information at a rapid rate, albeit of a different sort. As computers proliferate, we will be interacting with them directly more and more. What then is the common ground between the two? Can we use the computer to expand our culture? Interactive computer art provides a start toward examining these questions. This paper describes the evolution of a program designed to expand, both in time and depth, the information exchange between artist and computer.

Over several years of working with water color painting, I became aware of the process of myself executing a painting. The particular technique I was using involved letting the feeling of what I was painting manifest itself through all of the basic elements of water color painting: color selection and blending, brush selection and movement, drying and layering. There was a definite feeling of rhythm, fluidity, and wholeness that I felt while painting this way, particularly when moving the brush over the paper.

After acquiring my first microcomputer, a Cromemco System 3 (an S100 bus computer with a Z80 microprocessor), I became fascinated with the idea of exploring this process using the computer. I wished to use the computer as my brush, colors and paper.

With great faith in micro's abilities, I set out to create a paint system. The proposed system had two phases: In the description phase, the individual elements to be used in the painting, such as sun, plants, sky, lightness, would be described in terms of how they could be displayed and manipulated. In the painting phase, these elements would be manipulated, tweaked and otherwise adjusted interactively to the point where the painting harmonized. The link to water colors was that the elements would have to be able to bleed into each other, to overlay each other and still be transparent, to dissolve and reappear.

Due to some immediate limitations on video resolution and speed (I was using a 24 x 80 terminal and Basic), I simplified the design to the point where the objects or elements of the picture consisted simply of outlines of their own shapes. These shapes could be related in a hierarchal structure; the structure was employed such that when a higher level element was moved during the manipulation phase, all of its lower level members would move. However, lower level objects could be moved independently of their higher associates (see figure 1).

This program never got beyond the stage of the ability to specify elements and to have them move around the screen along structured random paths. What I certainly learned from the effort was the complexity of the problem of attempting to explain to a computer in a typical computing language what to do on the screen to make it look like it's raining. For this reason, I chose to write a program that would attempt to take this information from me in analog form and then expand upon it somehow using the computer's specialized type of processing power.

My question then became: "What process could be programmed that would retain the feeling of spontaneity and fluidity that I felt with my water color technique?" I knew that I did not wish to replicate the brush, colors and remaining tools of the trade; rather I wished to go deeper and get at the underlying elements of creating.

At that time I constructed a simple diagram, It is shown in figure 2. At first glance, the diagram seems simple: It is a person engaged somehow with a computer. It is the kind of diagram that people who work with computers, like myself, take for granted - or at least we tend to think about it in habitual ways. But what type of information flows between the blocks, and at what rate? What drives the flow of information around the loop? What animates the loop?

I thought about these questions and a few months later, armed with a new graphics board (Matrox ALT-512), a monitor, 2 joysticks and a 7 channel A/D board, I began programming in Z80 assembly language. Somewhat wiser from the experience of biting off more than I or my micro could chew, I proposed a simple experiment: Design a

CH1721-0/81/0000/0065$00.75

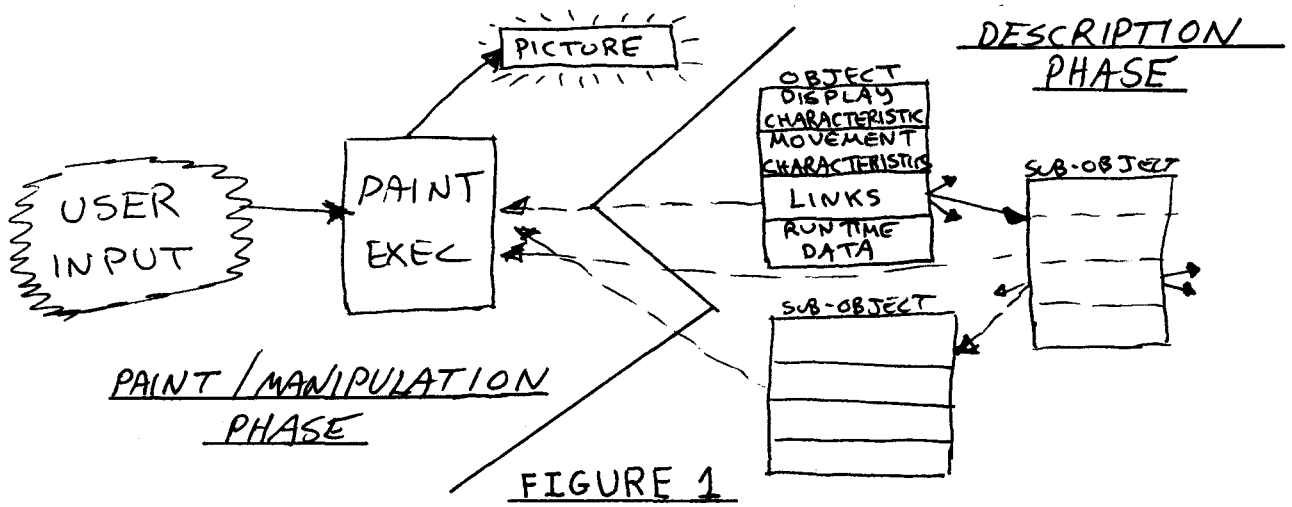


FIGURE 1

program that takes in fluid movement and feeds it back on the screen. The result was a program designed for two people and a computer; the two people each shared equally but differently in the same process while the computer read, combined, and processed the data, showing it on the video display at a rate determined by the people. One person controlled the speed and direction of a dot pattern while the second person controlled the pattern's shape. The dot pattern consisted of a series of up to 255 (x,y) vectors (see figure 3). The pattern was drawn onto the screen by plotting a dot at the current screen location maintained by the program, adding in the first vector to this location to arrive at the location for the second dot, plotting a dot there, and so on through the end of the pattern's vector list.

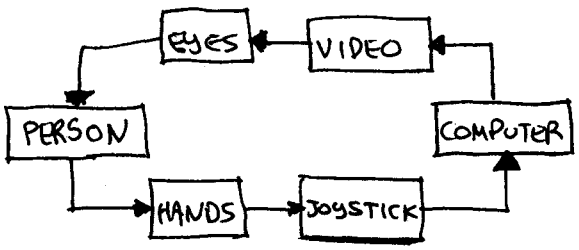


FIGURE 2

The flow of the program was straightforward: Executing in a continuous loop, it moved the pattern by the vector indicated by the speed & direction joystick, and then updated the shape of the pattern according to the input from the shape joystick. Before looping back, the program waited for a time determined by a speed control knob, and then erased the pattern in its current location. The program then resumed the loop (see figure 4).

The pattern was allowed to wraparound the screen in order that the feeling of freedom of movement not be restricted.

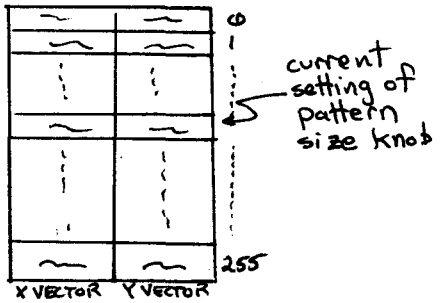


FIGURE 3

The most fascinating and admittedly unanticipated effect of this program was that of the dot pattern changing shape over time. Each time the program went through its main loop, it updated one of the vectors in the pattern's vector list with the vector just sampled from the shape joystick. As this was done sequentially, the effect was such that the dot pattern was constantly evolving, with the point of change proceeding from its head through its tail.

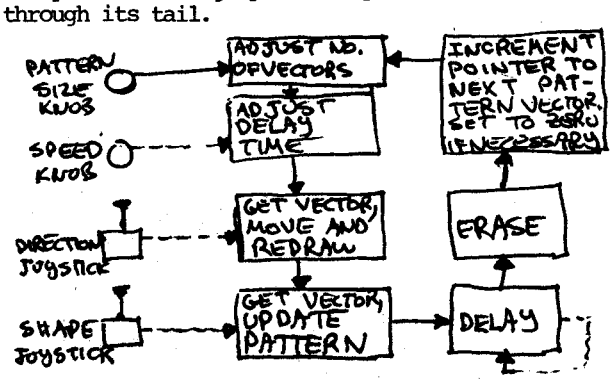


FIGURE 4

Even more intriguing to watch was a graphic demonstration of the dot pattern's memory (see figure 5). Since the change in the pattern's shape proceeded from beginning to end, the portion of the primitive that was between the head and the point of change would retain its shape until the program cycled through the rest of the pattern. However, the portion of the pattern following the point of change would be displaced by the difference between the vector being changed and the one replacing it. If the replacing vectors were all zero, the dot pattern would replay the motion that originally went into creating it. And since the movements were played back several seconds after they had been put in, it was quite surprising to see the computer play them back precisely. It was much like hearing a tape recording of oneself speaking, several seconds delayed.

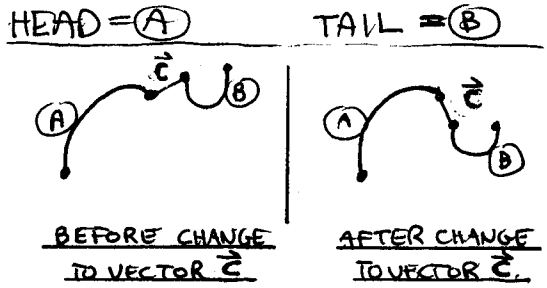


FIGURE 5

While rewriting this version of the program to use my computer's real time clock for smoother motion, a bug crept into the code and disabled my erase pattern routine. The resulting program, leaving splendid trails as the pattern changed and cavorted across the screen, kept me up well into the early hours, and earned the program the name of 'Channel Fifty Million.'

It was at this time that I began to confront a problem I would face a number of times during the evolution of the program: When does a feature not fit in? When is it too much or too little? What is the right balance of features, or the right way of implementing a certain feature?

A number of times I found that I had added a feature to the program that was either totally out of synch with the rest of the program or was so difficult to use or so remotely removed from the central activity of the program that it never got used. Looking back, I see now that the original purpose of the program, to be a fluid vehicle for movement, arose on its own as the central activity against which new features were rated. That is, a new feature which could not be employed to augment this central activity immediately stood out when it was tested. Typical of these errant features was something that would cause the user input to be transformed and fed back to the user in a less genuine way, such that boredom quickly ensued.

By this time Channel Fifty Million (CFM) had grown into a substantial program, and I began to use it to explore all of the features it seemed to offer. Sitting down with the program proved to be very relaxing as the continuously evolving screen almost begged the user to be more engaged with the process. Inevitably I would find myself putting on some kind of music to accompany the graphics. The ability to rapidly fill the screen, clear it, overwrite it within seconds allowed a kind of spontaneity that would be impossible without a computer. After a time, I would find that I was enjoying a particular effect. This could be something like a wavy pattern being moved upwards and to the left. It was here that it became desireable to have more exacting control over the program, so that something could be duplicated if necessary. For this reason I added a separate 'monitor screen' with its own dedicated video board, that would relay information as to the current joystick and knob settings, the currently selected level of grey, speed and pattern size.

Following are eight pictures that were created using this version of the program. Below are listed the features of the program.

1) Four level grey scale, 256 x 256 pixels
2) Variable speed control from 1 to 1,000 repeats of the pattern per second.
3) Variable pattern size control.
4) Ability to turn on/off either one or both planes on the ALT-512 graphics board, such that an image could be preserved in one board while another was written on top of it in the second board.
5) Ability to lock a given joystick setting so that one's hands would be free to manipulate other controls.
6) Ability to write in either or both ALT-512 graphics planes.
7) Ability to generate rectangular fill patterns based on joystick input.
8) Ablity to clear both or one graphics plane to white or black.
9) Ability to paint with a fixed or changing brush.
10) Freeze all/continue button.

With the intent of making my program available to the general computer public, I began creating a version for the Apple II computer. I increased the capabilities of the program significantly, taking advantage of the Apple's convenient passage ways from Applesoft to the disk and to machine language. Most important of these additions were:

1) The ability to use a dot pattern both as a brush and as a source of motion for another brush.

2) The ability to use the computer to compute pattern shapes based on various trigonometric functions.

3) The ability to specify painting modes, such as exclusive or with memory, fill dark areas only, change colors selectively, etc.

4) The ability to suspend painting in a particular direction while the brush is relocated to a different portion of the screen.

5) The ability to use the keyboard as well as the joystick to specify exact vectors. Also included is the ability to fix the horizontal or vertical components of vectors being input by the program.

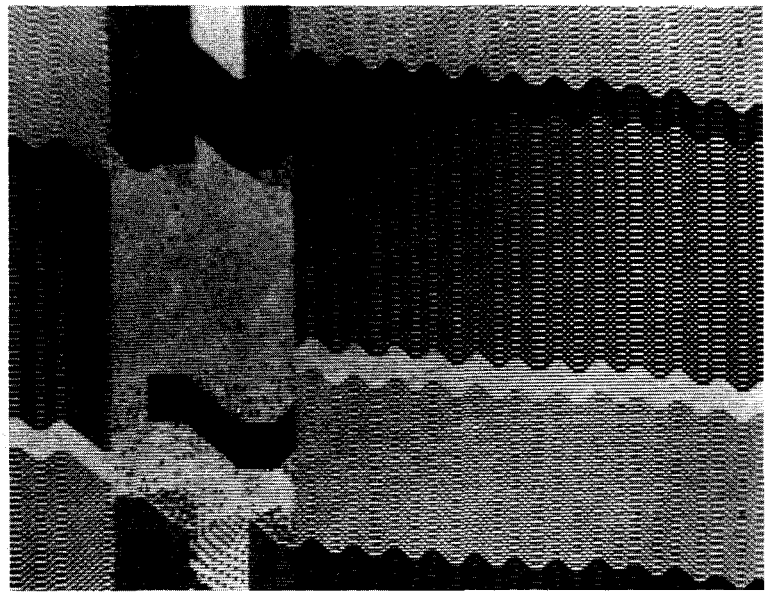
7) The ability to single step the program for exact movements.

Each of these improvements adds a new dimension of possibilities to the kinds of images that can be derived using the program. The more involved use of the program demands different kinds of attention at different times. For example, directing the program to create an elliptical brush as opposed to creating a brush with the joystick turns out to be quite a different activity, even though the two brushes may be the same.

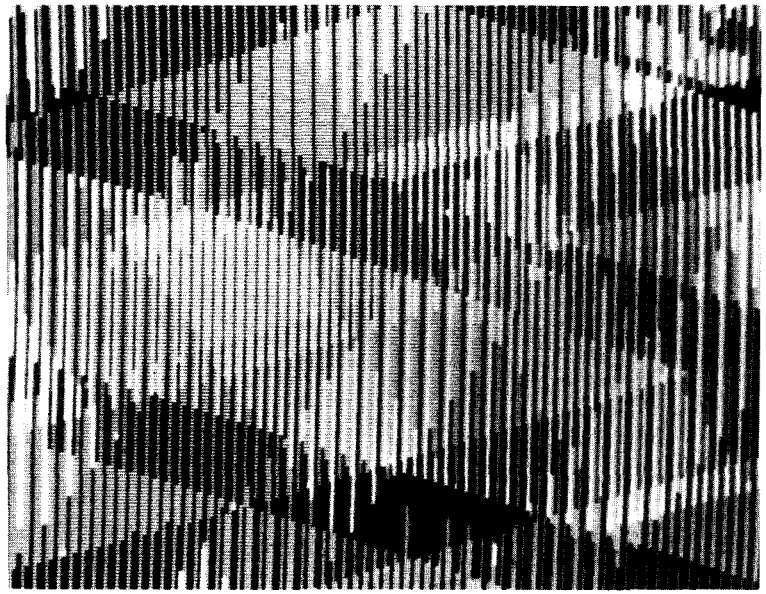
Conclusion

From this vantage point I see a program that allowed its user to jump in and feel the current of art in movement. I also see that my program has imposed a certain order on interactive computer art ; an order that ranges from what keys must be pressed when through the types of images that result. What I propose is that a language or system of some sort be organized and standardized such that people could impose their own order on the flow of art (or music). With such a system, art and music could be used as outlets for any process that could be understood by the system. Such interrelated activities would be something new for our culture.

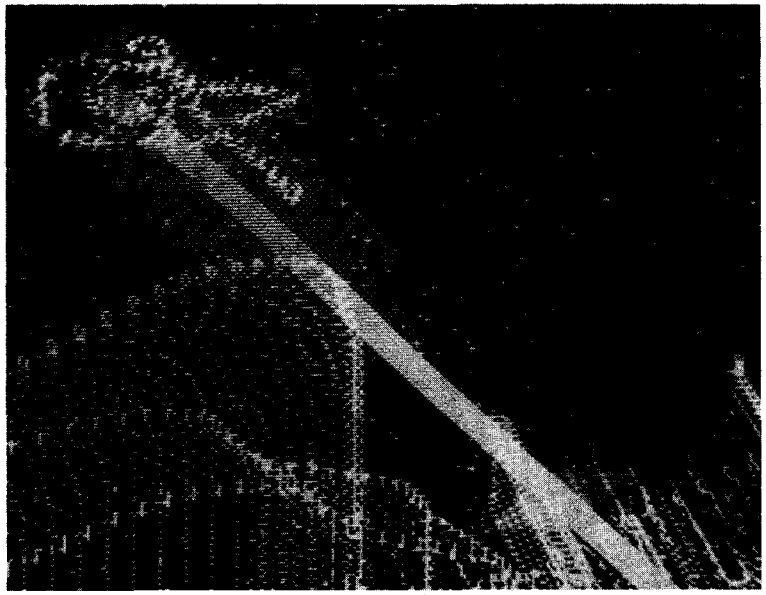
COLOR PLATES FOLLOW PAGE 92



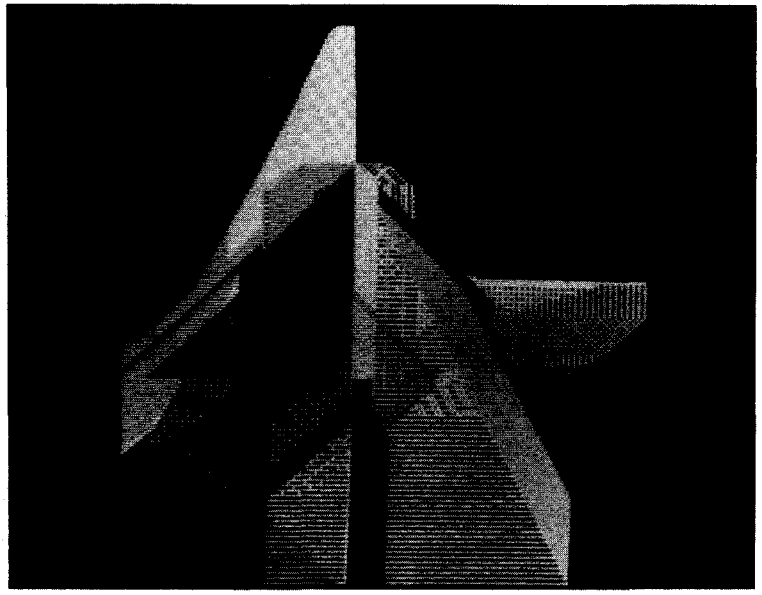
Kiosk



Slants

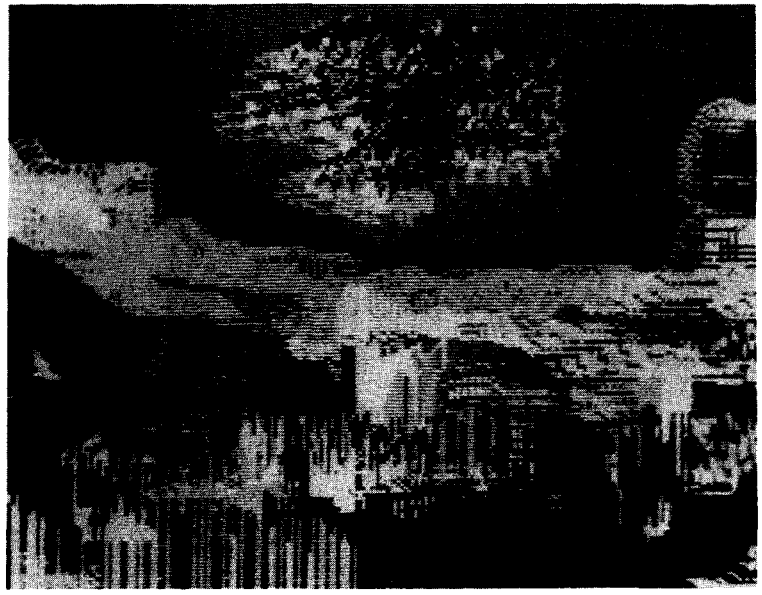


Tractor beam

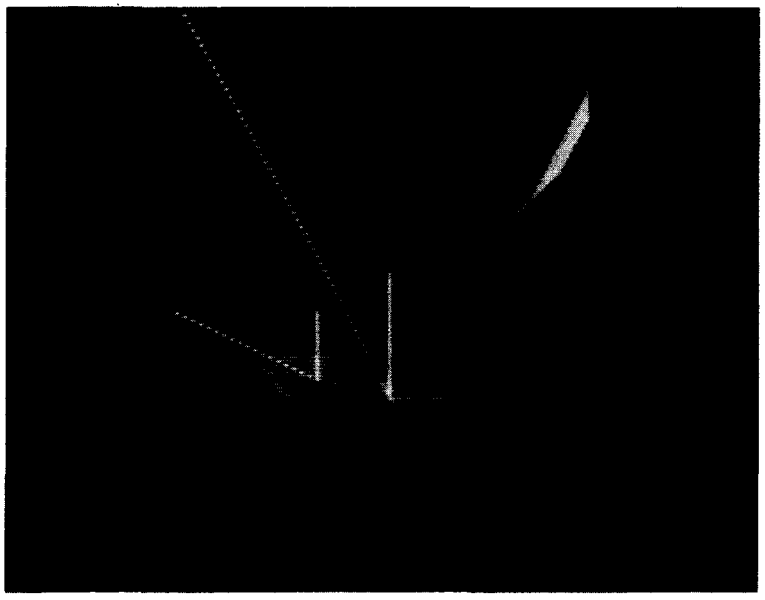


Hun

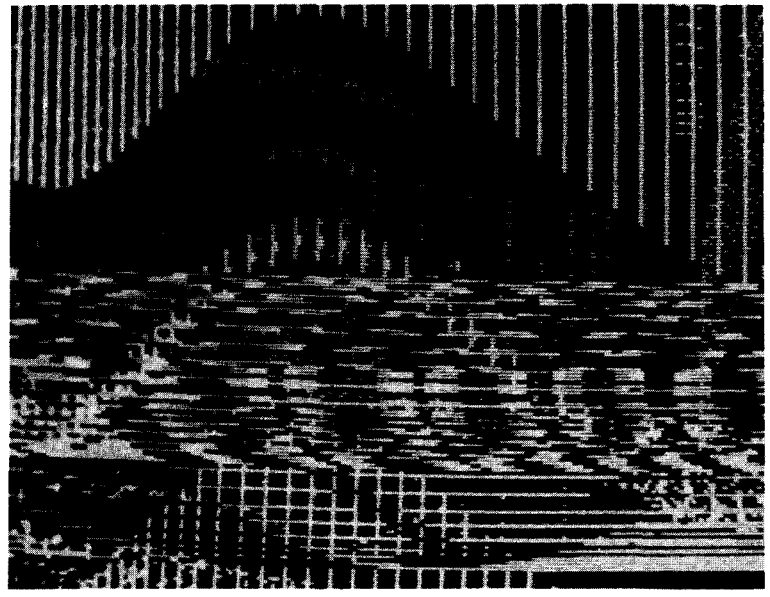
TV



Landscape



Mosquito



Subway

Structured Randomness in Real-Time Computer Graphics

by G. Sewell

TAF Information Systems
Spinnerstown, Pa.

Abstract

In this paper, randomness is presented as a tool that can increase a computer's appeal and performance. The inclusion of structured randomness in a program can be a powerful force in increasing user interest and attention span. The term "structured randomness" implies randomness within the superstructure of order. Programs which can benefit from randomly generated variables have two independent characteristics: (1) at some level in the program, multiple pathways can be taken, each producing a good result; (2) variables can be assigned any value within a defined range of values in specific routines. While the examples given use computer art programs, the subject of structured randomness could be useful in many fields.

Introduction

Randomness, when constrained within certain limits, can add a new dimension to computer programs. Two immediate results are apparent: (1) user interest and attention is increased, and (2) programs utilizing randomness appear more complex or dynamic than similar programs which generate values through some non-random means.

While not every routine can utilize these concepts, there is a class of programs which can be enhanced. These programs have the following characteristics: (1) within a program independent pathways can be taken to produce different, but pleasing results; (2) Within various modules of the program, certain variables can have any value between defined limits.

There is, of course, much software which does not fall into this class, such as missile guidance, profit and loss summaries, industrial automation, etc. The output of these programs must have a one to one relationship with the input. Calculations are performed by arithmetic formula, whether using differential equations or simple algebra, with no allowance for varying results.

Programs using structured randomness have two significant qualities. First, they tend to be of a creative nature, with unique and always varying results. Secondly, there is a greater opportunity for the program to produce these results in "real-time" (i.e. a separate phase of calculations is not required), because a random variable can be produced much faster than a variable calculated through an equivalent formula.

Consider, as an illustrative example, a graphic program displaying leaves falling from a tree. The program must define the general shape of a leaf, that each leaf must always be pulled by gravity in the general direction of the ground. The extreme limits of the flight pattern would be pre-determined. A program using the principles of randomness can then pick leaves of various shapes and colors and produce the appearance of a falling leaf by changing the glide ratio, the direction, and the movement of the air stream. This program would maintain high interest for the user as each leaf's fall would be different, unique. The display would be simpler and easier to produce than one generated by a program where every point is calculated to picture the fall of a leaf blowing in the wind.

Background

All things have a context, a history showing the forces which worked in the evolution to the present day form. Computer graphics represent a merger of two applications: the machine's power of calculations, and the image's power to evoke understanding of complex information, whether of data, or of symbols.

Graphics

Graphics have preceeded the written word. The first graphics were probably produced by cavemen recording the events of their lives. Early languages consisted less of an alphabet than of a collection of graphic images, each with implied meaning.

This early emergence of the graphic image in man's history should be a hint of the importance of graphics. The mind thinks, not merely in some linear fashion using letters and words, but using multi-layered images that at best are poorly

CH1721-0/81/0000/0071$00.75

translated into the language of the pen or the tongue.

Computer graphics has a power of representation that has been hardly tapped in the fullest expression of information exchange. As the computer has expanded the data processing ability far beyond the dreams even of Babbage and Turing, so will the computer expand graphic expression. The world does not revolve solely on hard facts and figures, and the role of computer graphics in the future can only be glimpsed through intuition and deduction.

Randomness

One of the aspects of life that everyone experiences daily, is the constant change that takes place in our environment. Each sunrise, each walk in the park, each moment is different and unique from all others. What is the source of this constant variation?

Since the early twentieth century, physicists have accepted Heisenberg's principle of uncertainty, holding that all physical phenomena are ultimately based on chance. This theory agrees with much experimental data and its expression has been traced through many physical and mathematical branches of science.

Contrary to this, people feel a sense of order and even meaning through the seeming random events in our lives. Religious people feel God is always in control of every event, shaping destinies of men and nations, while watching out for the sparrow and the lillies of the fields. Astrologers believe that the orientation of steller bodies at the apparently random time of birth bear a relationship to patterns of meaning. Mystics and occultists, scientists and farmers, all believe in laws which lie beneath seeming chance. Whether it is the prediction of rain or snow, the behavior of atoms, the arrangement of a tarot deck, good fortune or bad; there is the deep assumption that chaos does not reign, but that the randomness in our lives follows deeper laws of order.

It appears, then, that chance is more than it first appears. While experimental evidence insists that chance is at the root of physical reality, everyone grants that there is still order, whether imposed by physical laws or by part of God's greater plan.

Sample Programs

I am presenting two types of programs, both of which incorporate a high degree of randomness in their structure. These programs were developed using the Apple II computer, and were written in PASCAL, a programming language. The Apple II computer is known to all in the small computer world as an affordable "personal" computer. The cost of such a computer system is comparable to the cost of a good hi-fidelity system, or a large screen television.

The Dancing Wand

The first program is called "The Dancing Wand" whose main motif is a line or wand that moves across the screen display. (see figure 1 and 2). Although This program is very simple, it continuously displays fascinating results. It is a collection of artistic motif generators, basically independent of the others. Throughout the execution of the program, random numbers are used as much as possible.

The generation of random numbers in a computer is never totally random, but is created with software using a particular calculation routine. A sequence of numbers is initiated using a "seed" number, and after some long series of numbers, the sequence starts again.

In the programs being demonstrated, the random number generator is "seeded" with a "true" random number (which is distantly related to when the last key was pressed). In testing the generator for sequential repetition, it was found that sequence uniqueness continued into the millions. Therefore these numbers can be treated as random. When the random number routine is called in these programs, a random number between zero and a specified maximum will be supplied.

During execution of this program, the main routine requests a random number whose maximum is the number of subroutines (the motif generators) in the program. When the random number is obtained, there is a call made to one of these subroutines. This cycle repeats endlessly, randomly selecting which subroutine will be run next.

Each subroutine, once called, then chooses a variety of variables specifying colors, speed, size, position, etc. Each variable value is determined by the same technique of setting limits on the range of each variable, then calling the random number routine for each value. The patterns produced by the separate routines may or may not be cleared from the screen prior to the beginning of the next routine, again, depending on random chance.

The program by itself, is most interesting. There are spectrums of possibilities, each producing totally different results. There are differences in mood, differences in the apparent harmony between the different motifs, and differences in the combinations of color and form; in short,. always unique. The display on the screen changes rapidly and smoothly. The pictures illustrated (Figure 1 and 2), were taken only a few seconds apart. Any calculations required are executed as the program proceeds, hence "The Dancing Wand" can be called "real-time" computer art.

Very early in the writing of this program, the display became even more interesting to watch when music was playing in the background. The program appeared so entertaining that it became a regular pastime for my friends and I to play different types of music, and watch the program for

hours at a time.

It was in reference to the interaction between the music and the randomly driven nature of the program that I made my earlier remarks about the nature of randomness. When watching the program with music, it appears to almost everyone as if, at times, the output of the computer and the musical output are intentionally related. It was a favorite joke to introduce the program to newcomers as having been laboriously programmed for a particular piece of music. One senses a strong degree of interrelationship while watching the program. Even when the joke is explained, the interlock of the music and the graphic figure is, in the words of Spock, "fascinating". One senses a glimpse into the deeper nature of randomness.

Fascination with this program illustrates several points. First, the random display succeeds in maintaining the viewer's interest to an extreme degree. Few paintings hold the attention span to the extent that these video paintings repeatedly have done. Another program I wrote, based on the child's toy "spirograph", was very disappointing in terms of maintaining dynamic interest as no randomness was used. While producing beautiful mathematical shapes, the lack of spontaneous variation soon caused the mind to wander.

An additional point is that the total effect of "The Dancing Wand" is one of unity, and connection. Rather than producing a sense of dissonance, the incorporation of randomness has had an unusually harmonious effect on the appearance of the program.

Sunrise Over the Ocean

The second program I'm presenting (see figure 3), involves supplementing a preconceived artistic image with random parameters. The program generates a picture of a sunrise over the ocean in an impressionistic style.

While the rough details are fixed by the program, most of the fine details are randomly determined by the program. Thus, the ocean is below the sky, the sun rises, and the ratio of colors for each object on the display is predetermined. The specific shapes of the main figures (the beach and mountain), are randomly determined when the program is run. The timing of the sunrise is also randomly determined. Also, while the ratio and general area of placement of colors is specified, the exact color and placement of each brush stroke is randomly determined as the program progresses.

This program concisely demonstrates the flexibility of structured randomness, and is obviously more complex than the "The Dancing Wand", requiring additional support subroutines. The use of random placement and the color of the brush on each of the various shapes on the screen is still a much simpler approach then trying to create the same effect by explicit calculation of each point.

The setting of limits, and generating random numbers within these limits, allows the creation of unique works of "living" art, always changing and different. The beauty of the art, once established by the beauty of the programming, will be reflected in each new creation, rather than in one painting.

Conclusion

This paper has dealt with one particular theme: the role randomness can play in computer programs. The programs used for demonstration were of an artistic nature, but the theme can be applied to a much broader range of programs. It is my belief that computer graphics will play an important part in what many are calling the computer revolution.

Change the medium, however, and the attraction of this technique is still seen. Imagine language, rather than video art. Structured randomness will add new dimensions to the output of the computer. The programming must still be done, but the variety of the possibilities should provide a new dimension to such fields as artificial intelligence and computer aided instruction.

As mentioned before, pure randomness is chaotic in nature. Randomness without structure, or form is not the goal. In the specification of the form lies the creative act. The form may be specified by the programmer, or the end user of the program may participate in supplying the structure.

The study of nature has revealed many of the laws which bind her. Gravity compels a stone's flight back to earth. Electrons dance in measured tempo. With the same level of constraint, a computer program, using the techniques discussed dictates limits to random variables. Both physical laws and computer logic specify statistical limits of phenomena, yet randomness at the heart may contribute something all its own. Once made a part of the program, the random variable is a force by itself.

COLOR PLATES FOLLOW PAGE 92

COMPUTERS AND VIDEO

TOM MEEKS

COMMUNICATION RESOURCE MGT. SYSTEMS

A DISCUSSION CONCERNING SOME OF THE QUESTIONS RELATING TO USING COMPUTER ANIMATION WITH VIDEOTAPE.

SINCE THE LATE SIXTIES I HAVE BEEN INVOLVED IN THE PRODUCTION OF VIDEO TAPES FOR GOVERNMENT AND INDUSTRIAL CLIENTS. VIDEO HAS COME A LONG WAY FROM THOSE EARLY DAYS WHEN EDITS WERE DONE WITH A RAZOR BLADE AND A FINISHED PRODUCT WAS DEEMED A SUCCESS IF THE PICTURE STAYED UPRIGHT LONG ENOUGH TO GET A GLIMPSE OF WHAT THE NARRATOR WAS MUMBLING ABOUT. FORTUNATELY, MOST OF THE TECHNICAL PROBLEMS THAT PLAGUED US IN THOSE EARLY DAYS ARE GONE. (AT LEAST I DON'T CUT MYSELF WHILE EDITING ANYMORE.) UNFORTUNATELY, THE TECHNICAL IMPROVEMENTS INTRODUCED NEW PROBLEMS. NOW THAT THE NARRATOR CAN BE UNDERSTOOD (THANKS TO THE MINIATURE TIE-TACK ELECTRET CONDENSER MIKE) AND THE PICTURE STAYS UPRIGHT SO THAT IT CAN BE SEEN FOR OVER AN INCREDIBLE HOUR (THANKS TO ALL SORTS OF GOODIES)...THE VIEWER CAN TELL WHEN THE TAPE HAS PACING DEFICIENCIES. (READ: BORING!)

NOW THIS WOULDN'T BE SO BAD IF THE VIEWER WAS LOOKING INTO A HOLE IN THE END OF A SHOEBOX AND HAD NOTHING WITH WHICH TO COMPARE THE EXPERIENCE. I CAN THINK OF ALL KINDS OF GREAT EXCUSES WHY A PROGRAM VIEWED IN A SHOEBOX WOULD BE A LITTLE DULL AT TIMES. IT'S A LITTLE HARDER TO COME UP WITH CREATIVE EXCUSES WHEN THE VIEWING IS DONE VIA A TELEVISION SCREEN. GADS! WE'RE COMPETING WITH CBS, NBC, ABC AND THE MUPPETS. OBVIOUSLY, WE HAVE TO BE CONSTANTLY LOOKING FOR WAYS TO NARROW THE GAP BETWEEN WHAT THEY SEE AT WORK (BUDGET: LOW) AND WHAT THEY SEE AT HOME (BUDGET: OUTASIGHT.)

PORTABLE VIDEO SYSTEMS AND MODERATELY PRICED LOW-LIGHT COLOR CAMERAS HAVE HELPED MOVE US OUT OF THE STUDIO AND AWAY FROM THE TALKING HEAD FORMAT THAT DOMINATED THE INDUSTRIAL VIDEO PROGRAMS SEVERAL YEARS AGO. THIS HAS BEEN A REAL HELP IN PUTTING VITALITY INTO OUR VIDEO TAPES. WITH THIS IMPROVEMENT, HOWEVER, CAME THE REALIZATION THAT GRAPHIC SUPPORT WAS LAGGING BEHIND LIVE PRODUCTION. MOST OF OUR GRAPHICS CONSISTED OF STILLS SHOT FROM ARTCARDS...HARDLY EXPLOITIVE OF THE VIDEO MEDIUM. ATTEMPTS AT SIMULATING ANIMATION BY EDITING PROVED CLUMSY AND TIME CONSUMING. I THINK THAT EVERYONE IN VIDEO REALIZED THAT GRAPHICS WAS AN AREA RIPE FOR IMPROVEMENT. WE SAT BACK AND WAITED FOR THE VIDEO MOGULS TO SEND US SOME RELIEF. THEIR RESPONSE WAS CHARACTER GENERATORS WITH PRICE TAGS BORROWED FROM MERCEDES AND FRAME-STORE DEVICES PRICED LIKE A CONDOMINIUM AT THE BEACH. NICE FOR THE NETWORKS, BUT...

FORTUNATELY, HELP WAS ON THE WAY. IT WAS HARD FOR SOME OF US TO SEE BECAUSE IT CAME FROM OUR BLIND SIDE. IT HAS BEEN OBVIOUS FOR SOME TIME THAT VIDEO AND COMPUTERS WERE MOVING CLOSER AND CLOSER INTO EACH OTHER'S REALMS AND THAT THE LINES SEPARATING THEM WERE BECOMING MORE AND MORE BLURRED. THE FACT IS THAT THE LINES HAVE BEEN BREACHED FOREVER. FOR ME THE CROSSOVER INTO THE CAMP OF THE COMPUTERISTS OCCURRED WHEN I PURCHASED A BALLY ARCADE COMPUTER WITH TINY BASIC AND WAS FINALLY AND FIRMLY ESTABLISHED WITH THE PURCHASE OF A DATAMAX UV-1, A REMARKABLE VIDEO ANIMATION COMPUTER.

JUDGING FROM THE RESPONSE FROM VIDEO USERS IN WASHINGTON D.C., I CAN SAFELY REPORT A TREMENDOUS INTEREST IN THE KIND OF SUPPORT OFFERED BY COMPUTER GENERATED GRAPHICS, PARTICULARLY ANIMATED GRAPHICS. AS YOU MIGHT SUSPECT, THE KIND OF GRAPHICS SUITABLE FOR MOST INDUSTRIAL VIDEOTAPES LACK THE FLARE AND AESTHETICS NORMALLY ASSOCIATED WITH COMPUTER ART. HOWEVER, THERE IS A GREAT DEAL OF VALUE IN EXAMINING THE WHOLE ISSUE OF COMPUTERS AND VIDEO THROUGH THIS LOOKING GLASS.

FOR STARTERS, THERE ARE TECHNICAL QUESTIONS THAT MUST BE FACED WHEN DEALING WITH COMPUTERS AND TV ON A COMMERCIAL BASIS. IT IS NOT ENOUGH TO SIMPLY HAVE AN IMAGE REPRODUCED ON A TV RECEIVER. WE

CH1721-0/81/0000/0075$00.75

MUST BE ABLE TO VIDEOTAPE THE COMPUTER SIGNAL, EDIT THE VIDEO TAPE, COPY THE EDITED MASTER AND PERHAPS EVEN BROADCAST THE FINAL PRODUCT. BELIEVE IT OR NOT, THIS ELIMINATES THE POSSIBILITY OF USING MOST OF THE POPULAR MICROS. SECONDLY, THERE ARE THE DISCIPLINES IMPOSED BY THE FACT THAT THE COMPUTER IMAGE IS A SUPPORTING GRAPHIC...NOT AN END IN ITSELF. THUS, ANIMATION TIMING IS CRITICAL. OFTEN IT IS DICTATED BY A NARRATION TRACT RECORDED BEFORE THE COMPUTER ANIMATOR EVEN KNEW THE REQUIREMENT EXISTED. CONSEQUENTLY, THERE ARE SOME THINGS ABOUT THIS RELATIONSHIP BETWEEN VIDEO AND COMPUTERS THAT ARE MAJOR PROBLEMS IN AN INDUSTRIAL APPLICATION BUT ARE VIEWED AS MINOR ANNOYANCES IN OTHER AREAS OF COMPUTER ART.

FIRST, THERE IS THE VERY NATURE OF THE VIDEO SIGNAL. NTSC, THE STANDARD BROADCAST SIGNAL FOR THE UNITED STATES HAS COME UNDER INCREASING CRITICISM IN RECENT YEARS DUE IN LARGE PART TO THE FACT THAT IT JUST DOESN'T MEASURE UP TO THE RESOLUTION MADE POSSIBLE BY TODAYS DIGITAL TECHNOLOGY. RATHER THAN DEAL DIRECTLY WITH THESE LIMITATIONS SOME MANUFACTURERS HAVE BYPASSED THE NTSC STANDARD ALTOGETHER IN FAVOR OF THEIR OWN HIGH RESOLUTION DISPLAYS (INTELLIGENT SYSTEMS CORP.) WHILE OTHERS HAVE ATTEMPTED TO SOLVE THE PROBLEM BY ADOPTING A SIGNAL THAT IS "NTSC COMPATIBLE." THE PICTURE WILL PLAY ON A STANDARD NTSC TELEVISION BUT IS NOT A LEGAL BROADCAST NTSC SIGNAL AS DEFINED BY THE FCC. THESE SIGNALS CAN SOMETIMES BE RECORDED ON VIDEO EQUIPMENT BUT RARELY WORK WITH EDITNG SYSTEMS. THE APPLE COMPUTER FALLS INTO THIS GROUP.

ONE WAY TO DETERMINE IF A COMPUTER GENERATES A STANDARD NTSC "INTERLACED" PICTURE IS TO TURN THE VERTICAL CONTROL KNOB UNTIL THE BLACK BAR SEPARATING THE TOP AND BOTTOM OF THE PICTURE CAN BE SEEN. LOOK FOR A LINE THAT IS INTERRUPTED BY SOMETHING THAT LOOKS LIKE A HAMMER HEAD. IF THIS BAR EXTENDS ACROSS THE SCREEN IN ONE UNBROKEN LINE IT IS NOT A STANDARD NTSC BROADCAST SIGNAL. FORTUNATELY FOR YOU APPLE OWNERS THAT WANT TO BE ABLE TO RECORD YOUR COMPUTER OUTPUT THERE IS AN ENLIGHTENED VIDEO VENDOR THAT HAS COME TO YOUR AID. ADWAR VIDEO OF NEW YORK CITY MAKES TWO DEVICES THAT PERMIT THE APPLE TO BE RECORDED, ONE OF WHICH EVEN ALLOWS THE APPLE TO BE "GENLOCKED" TO OTHER VIDEO SIGNALS AND MIXED THROUGH A VIDEO SWITCHER.

THE PROBLEM WITH NTSC DOES NOT STOP WITH A NON-STANDARD SIGNAL FROM THE COMPUTER. IN FACT, THE GREATER PORTION OF PROBLEMS THAT A VIDEO PRODUCER WILL HAVE IMPLEMENTING COMPUTER GRAPHICS LIES AT THE VIDEO END OF THE EQUATION. THE DATAMAX UV-1 VIDEO GRAPHICS SYSTEM PUTS OUT A BROADCAST LEGAL NTSC VIDEO SIGNAL. THEY ALSO OFFER AN RGB MONITOR AS AN ACCESSORY. THE RGB MONITOR BYPASSES THE NTSC ENCODING PROCESS AND THE PICTURE IS GENERATED BY DRIVING THE RED, GREEN AND BLUE GUNS OF THE MONITOR DIRECTLY FROM THE COMPUTER'S COLOR CIRCUITRY. I WAS DELIGHTED WITH THE CLEAN CRISP ANIMATION THAT I WAS CREATING. THE RGB MONITOR GAVE ME A BEAUTIFUL PICTURE. BUT, WHEN IT CAME TIME TO TRANSFER THE ANIMATION TO VIDEOTAPE I WAS IN FOR A SURPRISE. THE VIDEOTAPE COULDN'T HANDLE THE INTRICATE DETAIL OF THE COMPUTER IMAGE.

A COMPUTER GENERATES A SIGNAL BY TURNING "BITS" ON AND OFF. THIS PART OF THE SIGNAL IS DIGITAL. A TELEVISION REPRODUCES THIS SIGNAL BY A SWEEPING BEAM THAT EXCITES PHOPHOROUS DOTS ON THE FACE OF THE SCREEN. THE BEAM CAN VARY IN INTENSITY FROM ALL OFF TO FULL POWER. THIS PART OF THE SIGNAL IS ANALOG. A DIGITAL SIGNAL IS VERY ACCURATE...IT IS EITHER ALL ON OR ALL OFF - INSTANTLY. AN ANALOG SIGNAL CAN BE LIKENED TO A VIOLIN. IF THE PLAYER IS REALLY GOOD THE CHANGE FROM ONE NOTE TO THE OTHER IS EXTREMELY ACCURATE AND FAST...BUT, IF THE PLAYER IS POOR THE CHANGES ARE SLOPPY AND SLOW. THE COMPUTER IS A GREAT COMPOSER - IT WRITES THE SCORE...AN RGB MONITOR IS A GOOD PLAYER...A HOME RECEIVER OR VIDEO RECORDER IS A POOR PLAYER.

NORMALLY, THE COMPOSER WOULD BE EXPECTED TO SET THE STANDARD. UNFORTUNATELY, IT IS THE POOR PLAYER THAT SETS THE STANDARDS WHEN IT COMES TO WHAT WE CAN AND CAN NOT DO WITH OUR COMPUTER GENERATED GRAPHICS. CERTAIN COLORS (MAGENTA) JUST DO NOT RECORD WELL. THIN LINES, LINE SPACED TOO CLOSE TOGETHER AND INDIVIDUAL PIXELS CAN CAUSE PROBLEMS FOR THE AVERAGE TELEVISION. THE ONLY CURE FOR ANY OF THESE PROBLEMS IS EXPERIENCE...LEARNING THE RIGHT COMBINATION FOR YOUR COMPUTER AND VIDEO APPLICATIONS. WHEN I CREATE COMPUTER VISUALS FOR MY OWN ENJOYMENT, I LIKE TO CREATE INTRICATE PATTERNS. HOWEVER, WHEN I CREATE VISUALS FOR VIDEO RECORDINGS INTRICATE PATTERNS ARE AVOIDED IN FAVOR OF LARGER AREAS AND BOLD LINES.

NOW, THIS MAY SOUND LIKE CREATING COMPUTER GRAPHICS FOR VIDEOTAPES IS TEDIOUS AND FRUSTRATING. ACTUALLY, THAT IS FAR FROM TRUE. THERE ARE SOME TECHNICAL PROBLEMS TO BE FACED. BUT, FOR ME, IT IS WELL WORTH THE EFFORT. THERE IS IS A GREAT DEAL OF PERSONAL SATISFACTION IN CREATING A WELL TIMED ANIMATION SEQUENCE.

MUCH OF THIS SATISFACTION COMES FROM THE FACT THAT I AM USING A LANGUAGE THAT IS PERFECTLY SUITED FOR CREATING VIDEO GRAPHICS. ZGRASS IS A PLEASURE TO USE. IT IS A "MACRO" LANGUAGE THAT LENDS ITSELF TO THE TYPE OF PRODUCTION METHODS COMMONLY EMPLOYED IN INDUSTRIAL VIDEO. WE ARE USED TO SHOOTING SCENES IN SMALL SEGMENTS AND LATER EDITING THEM TOGETHER. WITH ZGRASS,

I AM ABLE TO CARRY THAT SAME FAMILIAR WORKING STYLE OVER TO THE CREATION OF COMPUTER GRAPHIC INSERTS. EACH ELEMENT OF THE ANIMATION CAN BE CREATED INDEPENDENTLY AS A SEPARATE PROGRAM OR MACRO. FINE TUNING OF THE ANIMATION TIMING CAN THEN BE ACCOMPLISHED WHEN THE SEGMENTS ARE LINKED TOGETHER. THIS PROGRAMMING CONCEPT IS INDESPENSIBLE TO FAST EASY VIDEO ANIMATION. BASIC AND OTHER LINEAR LANGUAGES JUST AREN'T FLEXIBLE ENOUGH TO ALLOW THIS KIND OF FREEDOM.

RECENTLY, I CREATED AN ANIMATED SEQUENCE FOR A GOVERNMENT AGENCY THAT DEMONSTRATED THE PROCESS OF CONVERTING IRON ORE INTO STEEL. THERE WERE ABOUT 12 DIFFERENT ELEMENTS TO BE ILLUSTRATED. AFTER ACCEPTING THE ANIMATION, THE CLIENT DECIDED THAT THEY WANTED THE ORDER OF SEVERAL ELEMENTS CHANGED. IN BASIC THIS WOULD HAVE REQUIRED REPROGRAMMING THE ENTIRE SEQUENCE...OR, AT A MINIMUM, A HUGE RENUMBERING JOB. WITH ZGRASS, IT SIMPLY REQUIRED REORDERING THE NAMES OF THE MACROS FOR THE ELEMENTS IN A SMALL DRIVER PROGRAM. IT TOOK MINUTES - NOT HOURS. IT IS THIS KIND OF FLEXIBILITY THAT MAKES ZGRASS A PLEASURE TO USE.

MICROCOMPUTERS AND VIDEODISCS
Random Access Video Graphics

Dick Moberg

Abstract

Videodisc technology opens up a new medium for the computer graphics artist. This new technology when coupled to a small computer allows fast access to an enormous amount of high resolution video images. Although the initial cost of mastering a videodisc is still expensive, there are application areas such as picture data bases and interactive education where this technology outperforms all others. This paper will introduce the technology, describe the available equipment, and explore some of the applications.

Introduction

Working with high resolution graphics on a small computer can be a frustrating experience if more than a few pictures must be called from the disk and displayed in rapid succession. Disk read times are slow and the computer can hold only a few pictures at the most in their memories for rapid display. Videodisc technology solves these problems and allows fast access to large amounts of TV quality graphics both as single frames or as animated sequences. The limitations of this medium are the same as in the recording industry in that it is only cost-effective in large quantities.

This paper will explain how videodiscs work with a look at some of the equipment available. Then a few of the present and future applications of videodiscs in the arts will be explored.

Videodisc Technology

The idea of storing video pictures on a record-like disc has been around since 1927 when the Scottish inventor John Logie Baird demonstrated "phonovision" which played moving pictures from a phonograph record. His product was unsuccessful and the idea was largely ignored until the past decade where renewed interest on the part of several manufactures has produced a variety of commercially available videodisc players. These players, of course, are much more sophisticated than Bairds due to the complexity of today's TV signals compared to those in the 20's.

There are three methods for storing and retrieving the video information on discs but they all are based around the same idea. The video signal is converted to a frequency modulated (FM) wave such as seen in Figure 1.

Figure 1. An FM signal

Next, the FM wave is truncated at the top and bottom to produce a pattern as seen in Figure 2. Since the amplitude is constant in an FM wave, the original signal can be reconstructed from this pattern. The truncated wave can be likened to a digital signal in that it now could be sent as a waveform with two states, on and off, each of varying lengths.

Figure 2. Truncated FM Signal

In the disc mastering process the on-off patterns are etched into the master disc by a laser. The on part of the signal ends up as an etched "pit" in the disc and the off part as no pit as seen in Figure 3 An hour of video is usually etched per side of the disc in a spiraling pattern containing about 26 billion etch marks.

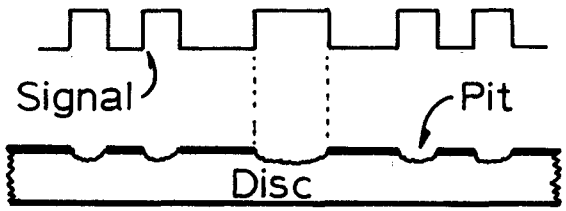


Figure 3. Etched Disc

The major difference in the three major technologies is in the reading of these etch marks.

The system developed by RCA treats the videodisc like a phonograph record and uses a stylus-in-a-groove approach to read the signal. The RCA discs are made of a plastic which is charged with electricity when spinning (450 rpm). The "needle" or stylus contains an electrode which senses the charge on the disc and since the pits are slightly farther away from the needle than the non-pits, less charge is picked up at these locations. Thus, an on-off signal is produced and the FM signal can be reconstructed as seen in Figure 4. This type of recording is called a capacitance electronic disc (CED).

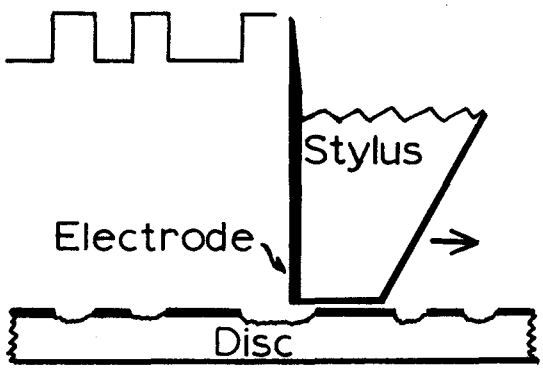


Figure 4. Capacitance-sensing Stylus

A second recording format developed by the Japanese company JVC is called the video high density (VHD) disc. It is similar to the RCA technology except there are no grooves on the disc. The capacitance pick-up stylus is guided along the track of pits by a second set of evenly spaced pits and a tracking mechanism in the player.

Instead of capacitance pick-up the third common recording format uses laser light and optical sensors. The pits are etched onto a reflective surface which is then coated with a protective plastic. The laser light is reflected back to the sensor in the area between pits and scattered when it hits a pit. From this on-off light pattern, the original FM signal can be reconstructed. These discs spin at 1800 rpm which is 30 times per second or once per TV frame. This allows a "freeze frame" by playing the same track each revolution.

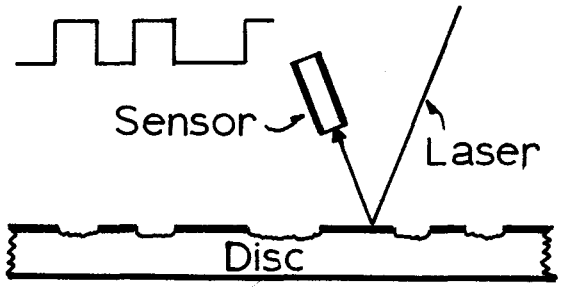


Figure 5. Optical Laser Disc

There are distinct advantages and disadvantages of each of these formats. Unfortunately they are all incompatible with one another.

Videodisc Equipment

Table I lists the various models of players. The large difference in prices indicates the difference between a consumer unit and an industrial model.

To a large extent, the playback technology dictates the availability of certain options on the player. The most important feature to look for from a computer interaction viewpoint is the ability of a player to find and play a particular frame (or frames) of video under computer control. Unfortunately this is not available with the RCA player which is the cheapest and most widely available player today.

Most players have slow motion, fast motion, and super fast motion (scan) capabilities in both forward and reverse. Most also have freeze-frame, two separate audio tracks, and a frame number indicator.

TABLE I

Videodisc Players

Make/Model	Format	Cost
DiscoVision PR-7820	Laser	$3000
Magnavox 8000	Laser	$775
Panasonic	VHD	$500
RCA SFT100	CED	$500
Sony LDP-1000	Laser	$3000
Thomson-CSF TTV 3230	Laser	$3500
US Pioneer VP-1000	Laser	$750

Interfacing a computer to a videodisc player can be done in two ways. The industrial models of the Sony and the DiscoVision player have a computer interface port right on the rear of the machine. An RS-232 interface board to the computer and machine language driver routines are all that is needed. Consumer models without an interface have been interfaced successfully directly through the remote control jack input. This method requires simulating the remote control outputs in software.

There are a few videodisc interface boards available which provide all the signal transfers and contain the necessary driver programs in ROM. Programming with these boards is quite straightforward and involves a set of BASIC commands which essentially duplicate the front controls of the videodisc unit and, in addition, allow switching between video and computer.

Videodisc Production

Videodiscs are mastered from videotape so, as far as the medium is concerned, the production techniques are the same as tape. However, the interactive nature of the videodisc must be understood to produce effective material.

At present, the cost of mastering a videodisc is around $2,000 and about $20 per disk in small quantities. These costs are expected to fall as the technology matures and higher yield mastering techniques are developed.

Applications

The applications discussed here will be based on the optical laser type of player due to its advantages in random access and computer compatibility.

Picture Data Base

Archival storage of picture data that can be readily indexed and accessed is possible with the microcomputer-videodisc combination. Text about each picture can be stored on a suitable mass storage device (floppy or hard disk drive) attached to the computer. 54,000 pictures can be stored on each side of a disc Certainly the world's greatest works of art could be stored this way for use in art education.

Distribution of Computer Graphics

Just as music is currently distributed on phonograph records, so it seems will be computer graphics pictures and animations. As soon as the costs come down, this will become a practicality

Computer Aided Instruction

Early attempts at having computers introduce new material to students were not largely successful due in part to the lack of good visual material Learning becomes boring if only text and questions are presented by the computer. On the other hand, video tape allows good visual material to be presented but it lacks the interaction necessary to provide feedback to the student and the instructor. A combination of the two technologies, computer and videotape, solves these problems; the only drawback being the long time period during tape searches.

Laser videodiscs go a step further and reduce the search wait time to a few seconds making them much more acceptable as the visual input in this type of training environment. In addition, there are other features of the laser videodisc players that make them even more attractive: single frame, slow motion backwards and forwards, single step, and two tracks of optional audio.

There are a few systems on the market today which combine videodiscs and small computers for training. The Bell & Howell "PASS" system uses their version of the Apple Computer and the DiscoVision player. The software is very expensive ($15,000 initial charge + $5,000 per year) but includes help from Bell & Howell.

An off-the-shelf authoring language, written by this author, is avaialble from Independent Video Consultants, Inc. and costs under $400.00. It combines an Apple Computer with any one of a number of laser based videodisc players including the Sony and MCA.

Summary

Videodiscs provide a means for the computer graphics artist to overcome the current storage and random access limitations when dealing with large amounts of picture data. The main drawbacks of videodiscs are the costs and that, unlike videotape, they cannot be re-used. But the technology will be developed rapidly due to its large potential market in the entertainment field and reuseable discs are not too far away. As a high quality, mass-producible video storage medium, the videodisc should not be overlooked by the computer graphics artist.

General References

1. Videodisc: How It Works, Technology Illustrated, pp 102-103, Oct/Nov 1981

2. Educational and Industrial Television, March 1981 issue on videodiscs.

3. Videodisc News, Published by Videodisc Services Inc., Washington, D.C. 20002

4. Independent Video Consultants, Inc., 125 W. Durham St., Phila., PA 19119.

The Electronic Gallery*

Martin Nisenholtz

Alternate Media Center

School of the Arts/New York University

Abstract

In this paper, the concept of a publicly accessible videotex art gallery is introduced. The mechanics and structure of the gallery are outlined and several economic scenarios are put forward. Four groups are discussed in the formation of such a gallery: the videotex system operator; the gallery owner or administrator; the artists; and the audience. The paper briefly touches on the distinction between entertainment and high art and attempts to place the videotex gallery in context to each. Finally, the first such attempt at an electronic gallery, sponsored by the National Endowment for the Arts, is briefly outlined.

The term 'videotex' refers to new telecommunications services linking the home or business to large data bases via the telephone network. Until recently, 'videotex' referred specifically to systems that used ordinary home television sets as low cost display terminals and dedicated hardware as decoder sets. The videotex concept has since been expanded to include the business market and the personal computer user because the public is not purchasing dedicated videotex terminals at the rate formerly predicted (e.g., millions of users by 1982) by planners.

As late as 1980, videotex was thought of primarily as an information - distribution medium which would allow individuals in their homes to pick and choose from a vast library of knowledge stored in a central computer. Since the sluggish growth of the British service, Prestel, in the domestic market, there is considerable doubt whether the general public will ever be motivated to disgest even the tiniest portion of this vast information pie. Rather, on the basis of more recent market research, videotex planners have shifted their thinking away from an information retrieval concept toward a transaction services concept. The latter implies home banking, electronic mail, teleshopping, and other services requiring two-way communication. Using a variety of terminal devices, including personal computers (e.g., Apple, TRS-80), videotex decoders (e.g., Telidon, Prestel), and dumb terminals, people in their homes may have access to a videotex service which "bundles" these interactive transaction, information, and entertainment services into an easily accessible package. Within this scenario, the communicating personal computer is the cornerstone of videotex market development.

Interactive services may lead to a reduction in personal travel and energy consumption. If we bank at home, there will be less need to travel to take care of our personal finances. Similarly, by shopping from home, we may reduce considerably our need to travel to shopping malls, stores, and other places where retail goods are sold. With access to information banks and telesoftware, there may be less motivation to go to the library, book store, newsstand, or software store. By sending mail and paying bills electronically, there will be less need to go to the post office to purchase stamps or mail letters. In general, interactive telecommunications shift one's practical orientation more into electronic space. Physical space becomes less important to carrying out a variety of mundane and not so mundane tasks. This does not mean that people will never travel, go out, go shopping, or leave their homes and offices. Nor does it mean that banks will disappear or that stores will vanish. Rather, the implication is that there will be less of a dependency on certain types of physical spaces like banks, to pursue specific activities. The experience and appreciation of new forms of visual art is one such activity; hence, the electronic art gallery.

The mechanics of an electronic art gallery are straightforward. The economics are not. Concerning the former, artists work on a computer graphics system, store their finished product on disk, and then make the work accessible to others via telecommunications lines. The arts data base can be structured in several ways. Perhaps the most logical structure is roughly equivalent to

*The Electronic Gallery Project is funded by the National Endowment for the Arts (NEA 12 3443 287). Any opinions, findings, conclusions or recommendations expressed in this publication are those of the author and do not necessarily reflect the view of the National Endowment for the Arts.

to the artist's cooperative. A group of artists collaborate to structure a data base, place their artwork into it, negotiate to enter the data base into a videotex system, and make it accessible to the public. Another organizational structure is parallel to the commercial gallery. Owners or administrators intercede between the artist(s) and the public. In this case, an entrepreneur or ad-administrator creates the data base structure and invites artists to participate in the gallery. The artist maintains no responsibility for funding the gallery. He merely creates work for it. Another less logical approach is the free-for-all method, in which individual artists simply enter work into an ever expanding, publicly accessible data base. By virtue of its size and lack of practical administration, the data base would likely be difficult to use and the quality of the work would be uneven at best.

The economics of a videotext gallery are roughly defined by the relationship among four groups. First, the system operators, the group who manages and markets the videotex service, are roughly equivilent to landlords who rent or sell electronic space on their system.* The electronic gallery owner or administrator will have to pay this fee for space on the system. In turn, this second group, the owners and administrators (in the collaborative scheme, these are the artists), determine how prices are structured and costs are recovered. Again, several options come to mind. The artwork can be sold on a single piece basis or by edition. In this scheme, the artwork must be copy protected, a process which is never entirely foolproof. A more logical course might be to sell access to the gallery on a subscription basis. Anyone who pays a monthly or yearly fee will have access to the material coming from the gallery for the time period covered by the subscription. A modification of this arrangement might be to charge on a per access basis. Each time a work is accessed, a charge is made. Finally, the gallery might be publicly available at no charge, supported by government or foundation funds. The gallery would be free to all users of the system.

The third group which will influence the economics of the system are the artists. They will have to negotiate on pricing and copyright policies. They might take a fixed percentage from the gallery, sell their works outright, or collaborate to eliminate the middlemen by forming their own data base and setting their own pricing policies.

*The British Post Office, which administers the British videotex service, Prestel, charges information providers 5000 pounds per year and 50 pounds per page per year. Thus, the yearly "rent" for supplying a hundred page magazine is ten thousand pounds.

The fourth and final group which will influence the economic setting are the people who may wish to experience these new forms of art. These will be individuals or organizations with access to terminals, primarily small computers, who can afford to rent or purchase these new artforms. One can imagine flat screen display devices prominently fixed against white walls, showing artworks which change periodically. In the short run, however, the personal computer user will be the major outlet for the electronic gallery.

In painting and sculpture, the artist usually works alone, creates a single copy of his or her work, and distributes that work through a commercial gallery. In the theatre, film, television, and the concert hall, artists work in collaboration with large groups and the costs of production demand very large audiences. This is an important distinction in the arts today and perhaps the major reason why the visual arts are less aligned with popular culture than are the performing arts. Like photography, electronic art can align itself either with popular or high art culture. In the pay per access and subscription scenarios, the economic rationale is to appeal to as broad an audience as possible, thereby plunging the artistic process into the arena of entertainment and popular culture. The technological basis of the artist's work, the digital information which makes up the program, can be distributed widely. Unless the program is copy protected, a process which is never entirely foolproof, copies can and will be made. The collector can never be entirely certain that his is a unique copy. Much like television, the medium can gear itself toward a mass audience.

On the other hand, it is possible to copy protect a program with a reasonable degree of security. If the subscription or edition is limited to a small number of collectors, the price could be fixed at a rate which might insure the creative independence of the artist. This is narrowcasting at its extreme, where the act of receiving the transmission brings with it a sense of ownership and status.

Perhaps the best way to insure creative independence is to make public funds available to artists who work in this new medium. Without the impositions of the marketplace, the artist and gallery administrator will feel less motivation to pander to popular tastes. In fact, the first videotex gallery is publicly funded. In February, 1981, the National Endowment for the Arts awarded a grant for the first such gallery in the United States. The grant was made to the Alternate Media Center, School of the Arts, New York University, under the directorship of the author. Fifteen artists are being accepted to create works using Telidon, the Canadian videotex system. The works are stored on floppy disks. Although the project is still in its initial phase, it is envisioned that the gallery will be transmitted through Telidon field trials in the U.S. and Canada. The works will also be exhibited at a public gallery

in New York.

As peripherals and software are created to interface Telidon with personal computers, the range of potential artists and audience will greatly expand. Artists will no longer need to come to a center which has a Telidon terminal to create works. Artists who wish to participate will be able to create works from their own studios. People who wish to experience the gallery may do so using their personal computers, without having to purchase a dedicated Telidon decoder. The team of the National Endowment for the Arts' Electronic Gallery look forward to, and support, the marriage of videotex and the small computer.

COMPUTER ART EXPERIMENTS OF THE LAST TEN YEARS

H. Huitric - M. Nahas

Departement d'Informatique
Universite de Paris VIII

Abstract

In this article we will present a survey of the stages of our work from 1970 to the present. First, we will take into consideration works completed between 1970 and 1975: period preceding specialized peripherial devices for direct color processing. Then, we will show how this research was expanded on a color video monitor by summarizing a few methods for generating images as well as their transformation achieved from the use of data files of recorded images. In a few examples we will introduce problems posed by the encounter with other fields of research: chiefly digital music and picture processing,

To include, we will explain how the continuation of this work is heading towards the production of realist images on a new machine of improved precision.

I. 1970 - 1975

In the decade from 1965 to 1975, computer art was almost exclusively centered on black and white graphics. Very different products from different authors and the birth of varied graphic styles, proved the richness of this tool as evident and it's permutational potentials s quite interesting. But, very little was produced in color image processing except for the examples of J. Whitney, J. Citron, and K. Knowlton.

I.I In 1970, specialized peripheral devices allowing automatic color processing did not yet exist in France. How, then, go about calculating and representing colors? With the ' Groupe Art et Informatique de Vincennes', we adopted a pointillistic method which permits approaching a given color of the chromatic circle by optic fusion of colored points chosen from a range of six colors. The desired color is represented by a matrix of colored points distributed aleatorically according to certain percentages. In this way a palette of 48 colors is produced.

Being limited to a palette imposed considerations of the harmony in the calculation of results. As early as this time we tried to conceive color as a continuous entity and not discreet; for this reason we abandoned the idea of a palette and we considered the percentage of basic colors in the matrixes as our parameters. Therefore, we were able to approach any element of color or luminosity. In this way we developed our first model of a " continuous series of constant luminosity" where the percentages of basic colors are interrelated by a linear relationship designed to assure an identical luminosity in each surface element. (1, 2)

This relationship does not determine all of the percentages. Certain, arbitrarily chosen, are fixed by continuous functions of the variables X and Y of the working plane. Thus, each element on the plane of the image is of a shade close to that of all of its enighbors; the fact which evoked the name " continuous series". The results obtained brought forth a movement of color noticeable from one zone to an adjacent zone. These zones, where the probabilities of one basic color remain constant, are zones of different levels. " Levels" intrinsic to any models of continuous color variation; a continuous series, by its construction, implies the existence of constituent " lines of level".* The images produced through this pointillistic method present a granulated structure formed from neighboring colored points melting into each other and contrasting to the rest of the image. This is what we call the texture of an image. In our case, the texture is related to the pseudoaleatorical distribution of colored points inside the matrixes. In order to control this texture we simply varied the periodiicity of the aleatoric function.

To summarize our first works, two concepts come forth in the construction of a continuous series:

Lines of Level
Texture.

I. 2.Production of results from 1970 to 1975

The restitution of calculated colors was first achieved manually by painting colored points on the letters of computer listing where

* " Lines of level being the direct translation of "lignes de niveau" corresponding to isometric lines such as isobars.

each letter represented a specific shade.

In order to accomplish this fastidious procedure of painting each point of an image by hand, it was necessary to reorient the use of equipment that was at our disposition: card punch and plotter.

The card punch was used to create stencils from cards;

Through programming, the position of a basic color was perforated on a card that constituted an element of the resulting surface.

This technique was most satisfying and permitted numerous experiments through programming.

Subsequently, we used the plotter to outline stencils for silkscreens corresponding to the decomposition of primary colors (yellow-cyan-magenta) of calculated images.

This process permitted an enrichment of the color and opened a new area of investigation: that of color permutations from the permutation of the basic inks.

II. Video System Controlled by Computer: Colorix I, 1975 - 77

II. Generational Methods

In 1975, Louis Audoire and Roger Tanguy (Computer Department, Universite de Paris VIII) constructed a system that permits visualizing color images on a cathode screen controlled by computer.

Thus, it is possible to generate color images automatically from additive synthesis, On this system, Colorix I, color is defined for each point of the screeen by three numbers between 0 and 15, which characterize the degree of luminosity of its three components: red, green, and blue.

The number of shades of color is 16^3 or 4096.

Therefore, each color is described by a point in a three dimensional space formed by its three components: R, V^+, B. The group of possible colors on the cathode screen is formed by the points of whole coordinates of a cube of colors.

The basic functions of this system, accessible in LISP (3), permit coloring any and all of the points of the color screen with any of the 4047 shades.

It is therefore possible to program the generation of images point by point.

In the beginning, our work was oriented towards the contruction of continuous series on this new system. The value of a color component was calculated by a continous function, f (x,y), for each point of the screen with coordinates x,y.

The image was then defined geometrically by the shape of the curves of " levels" attached to each color component; curves defined by the equation f(x,y)= constant.

The first continuous series were thus conceived as having both straight and circular lines of level: each component varied continuously from 0 to 15 according to a family of concentric circles or parallel lines filling the screen. In programming we reduced the total surface in order to obtain continuous series in rectangles of any dimension. In repeating this process we built homothetic variations of continuous series.

To obtain complex images bounded by any contours, we constructed a continuous series demarcated by a triangle. Then, we used this triangular function inside a repetitive loop. This triangular procedure can be apparent as in figure 3, or not as in figure 4, depending on if the color control variables are dependent on each triangle in consideration.

We obtain, in this manner, effects of folding or depth where the process of construction(triangulation) ceases to be noticeable in viewing the results.

II. 2. Transformational operators of Images

Up to this point we have outlined methods for generating images. Techniques of recording the colored points of the screen in data files will now permit us to work from a library of images previously established in LISP (3).

It is then possible to develop methods for the transformation of given images:

The first example of such a method is the " mixture of images";

A mixture is an image obtained from two or more basic images, in combining, at each point of the screen, the corresponding color points of each image to be mixed;

This combination is a function which characterizes a " mixture";

For example, using a barycentric function in the form: f (x, y)$=\frac{ax+by}{a+b}$, it is possible to scramble an image by another, or to make one appear as transparent on another.

We have experimented with diversified methods to create sequences of images corresponding to different formulas for mixtures, Depending on the formula chosen, the characteristics of the sequence vary. They either converge towards an image of a certain limit or they diverge, or they may also oscillate between the two images.

For example, if we chose a sequence where each element is obtained by mixing the preceding element

+ V for Vert, french for green

with a fixed image, by a center of gravity, we obtain a sequence converging towards an image that is possible to calculate.

As a result, we can create a chain of images between an initial image and an image serving as a boundary.

We might ask then, if any image can be placed in a sequence? This problem does not always have a solution. It does in sequences having a center of gravity (barycentric), where each element of the sequence is obtained by mixing its two antecedents, In this case the two antecedents are filters of the original image, tending towards either light of dark shades.

Note that deforming images are also facilitated by recording the color points of the screen in data files. In this way sequences of deformed images can be created for animated works.

III. Encounter with other fields: Techniques Derived from Digital Image Processing

The computer being an instrument used in many fields, it allows intersections between them. We are particularly interested in techniques of image processing (4) which leave artists all but indifferent.

Most of these techniques work with the decomposition of an image by basic orthogonal functions. These could be, for example, decompositions of images by Fourier or Walsh.

These techniques are well know in computer music, but to our knowledge they have not yet been introduced in computer art.

We began to explore the concept of spatial frequency by constructing low frequency images; either from a previous image for whcih we calculated the first coefficients of Fourier, or in a purely generative way of arbitrarily assigning the first Fourier coefficients.

To create images having a large spectrum of frequencies we used a method well known in computer music ; that of frequency modulation (5). In this way we are able to enrich the shades of a low frewuency image and obtain effects of vibration.

Lastly, we have used Fourier's expansion of an image in order to produce different filtered forms of the same iamge. We can thus deform an image progressively in blurring it towards one uniform image by successsive high frequency filters. In the same way, we obtain, by low frequency filters, progressive deterioration of this same image towards an image where only the contours are visible. The use of Walsh functions seems equally interesting, and more economical to calculate, but leads to less continuous results.

We have also experimented with techniques of histogram transformation, used for problems in the restoration of paintings. This method permits a reduction of the number of color levels present in an iamge in a way that accentuates forms or sometimes produces the appearance of a form that was behind the contrcution process.

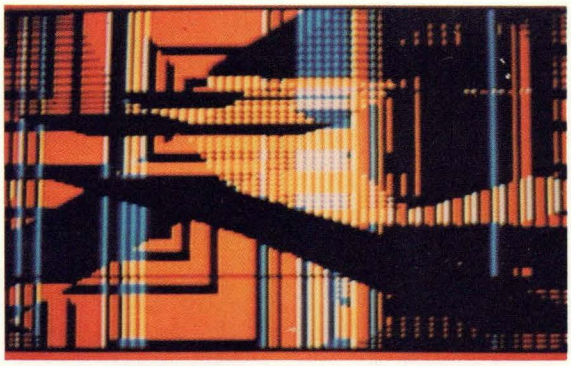
IV. Conclusion, Present Work on Colorix II

Since 1980, we have a new system at our disposition that provides improved graphic precision, 100,000 points, and color definition identical to that of Colorix I, 16^3 different shades. On this system we have resumed studies of continuous series with a luminosity parameter related to the video monitor in the form $L = \alpha R + \beta V + \gamma B$, where α, β, γ and are corrective factors and R, V, and B are the values of the component colors of a point. We have thus generated a sphere illuminated by a single light source, as well as a triangle in space illuminated in the same way. The introduction of a relation of lighting by the variable L permits us to undertake the study of realistic images. The precision of the screen also permits us to employ algorithms for the generation of arbitrary curves, for example, that of SPLINE (6, 7) which allows the constrcution of curves from control points.

Lastly, the mixture of images described in paragraph 2 now permits us to simulate transparencies. With these tools we can generate continuous series in any contour : a hand,a back, where the continuous variation of color represents a variation of light decided ahead of time. In this way we approach the generation of realistic images. In this respect we are most interested in the numerous works of research being developed at this time in the field, and hoping to be able to profit from their results when our means will permit us. For the artistic impact of this research can not be denied.

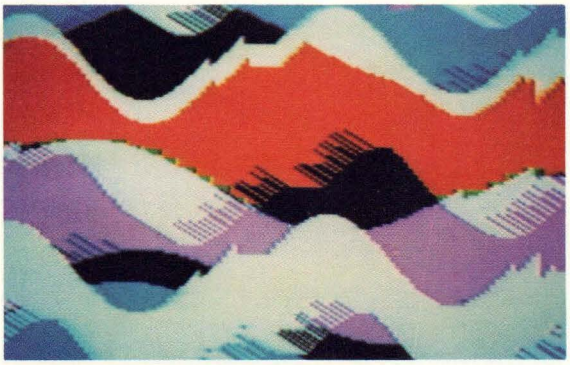
COLOR PLATES FOLLOW PAGE 92

Author Index

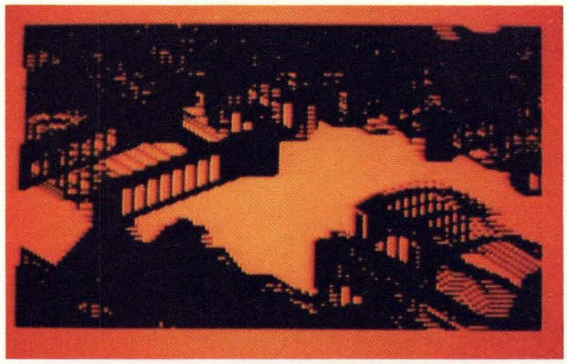


Day 'n' night

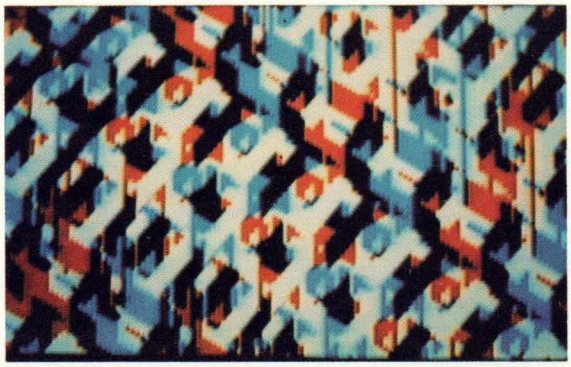
Oriental



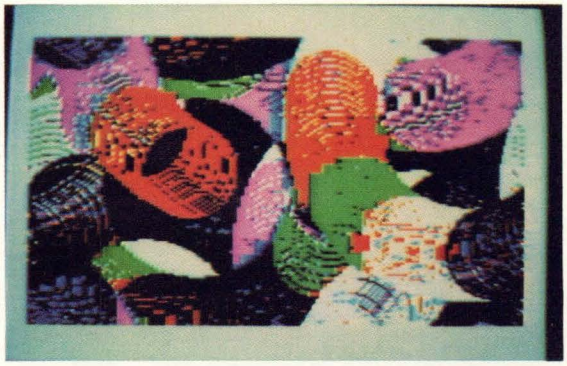
Double circles



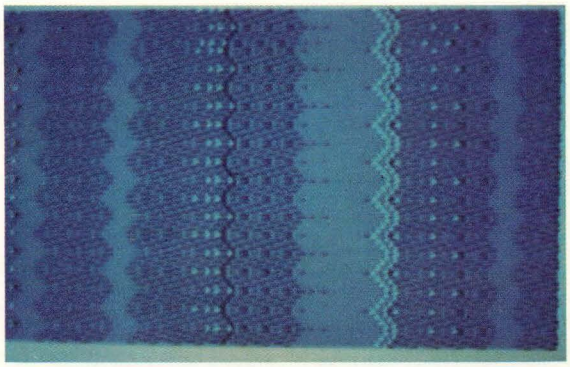
City at nite



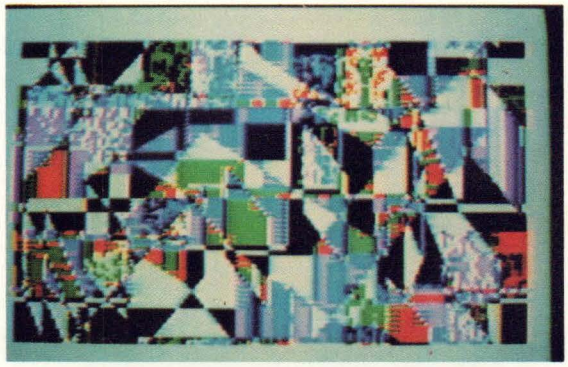
Clickwork



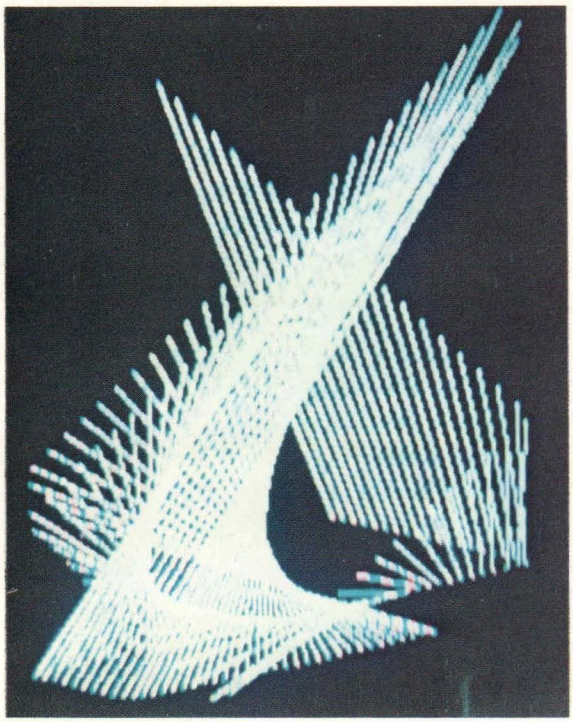
Into circle

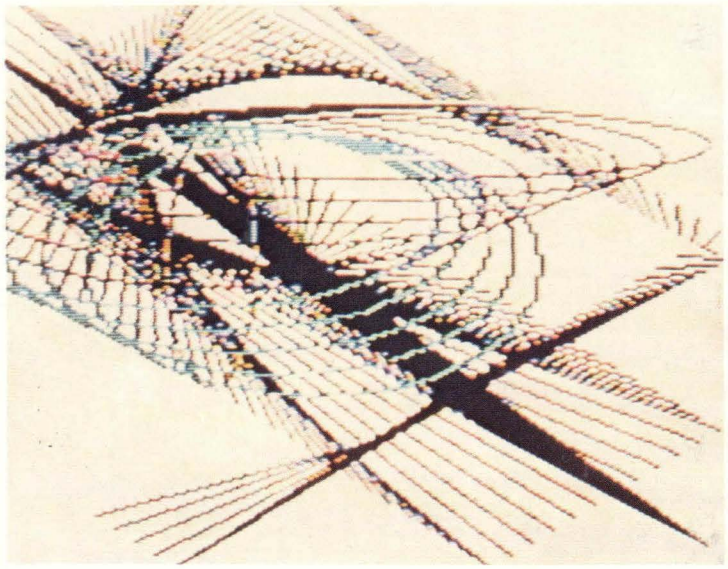


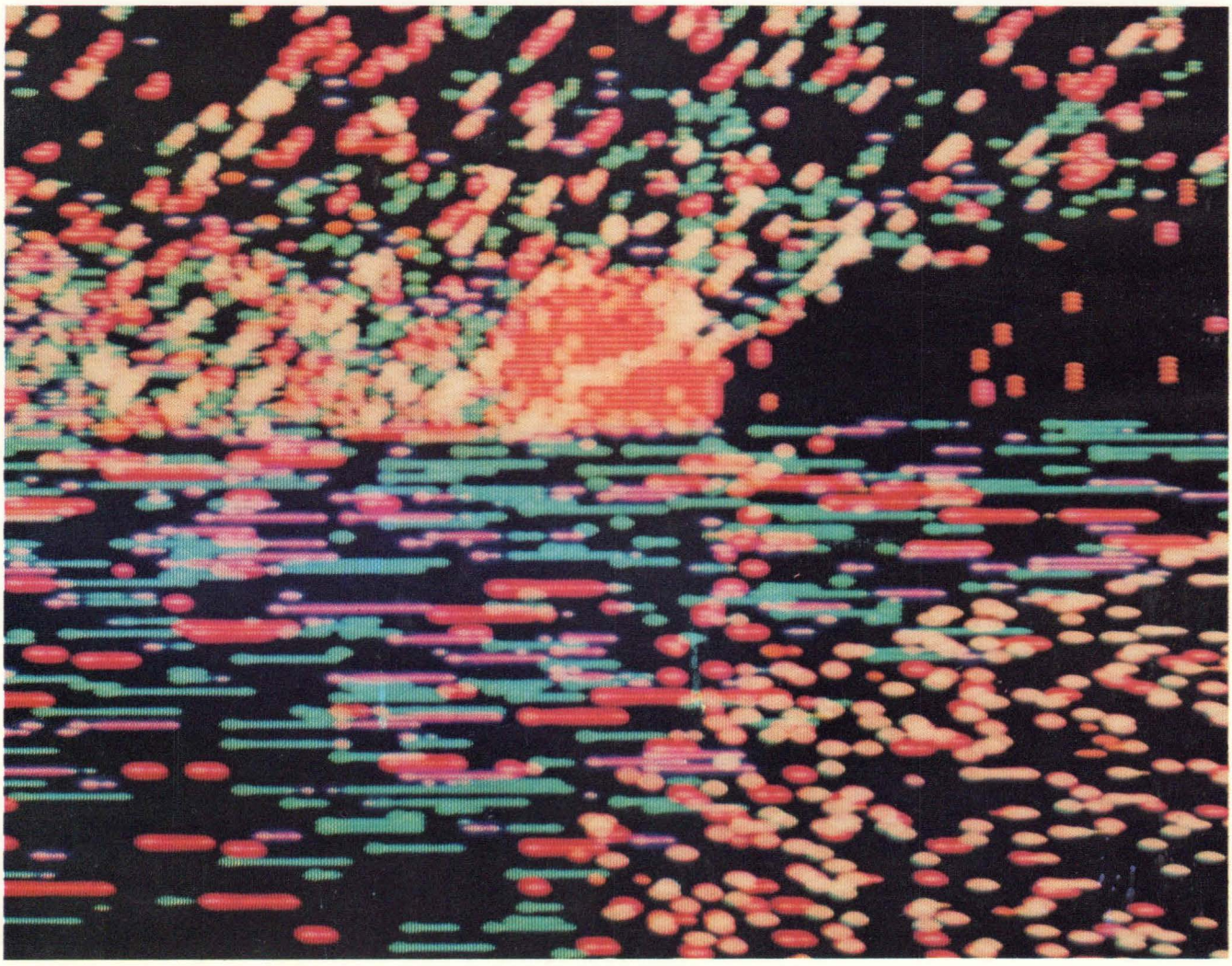
Speed color

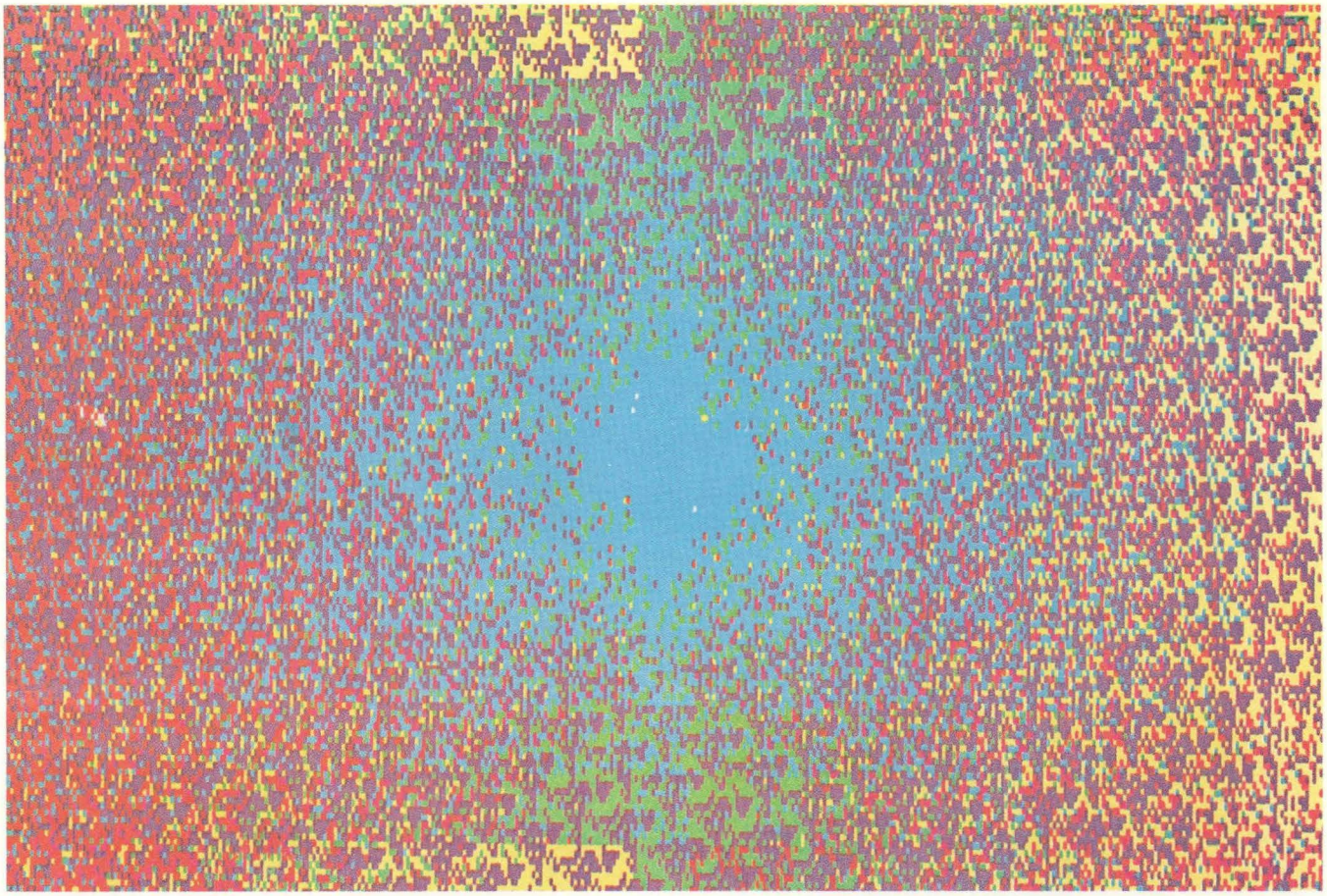


Cubist

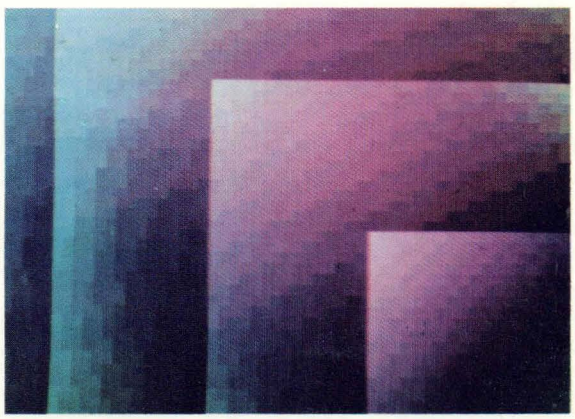




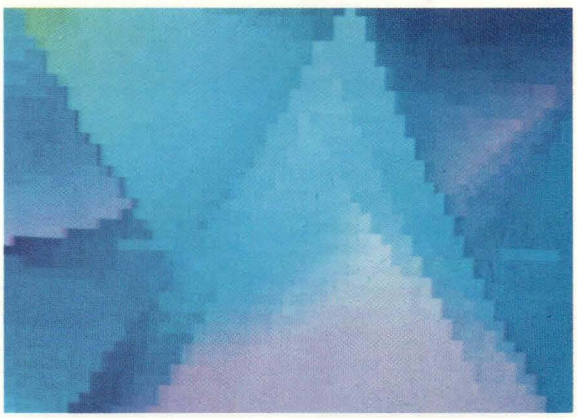




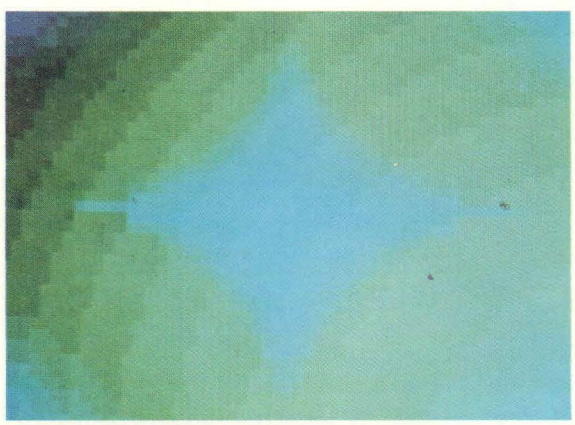
Example of a silkscreen where the stencils were drawn on a plotter—from a model of a continuous series of percentages of basic colors.



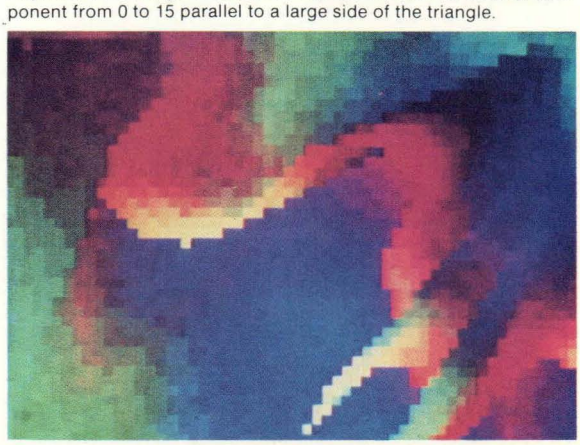
Homothetic variation of continuous series.



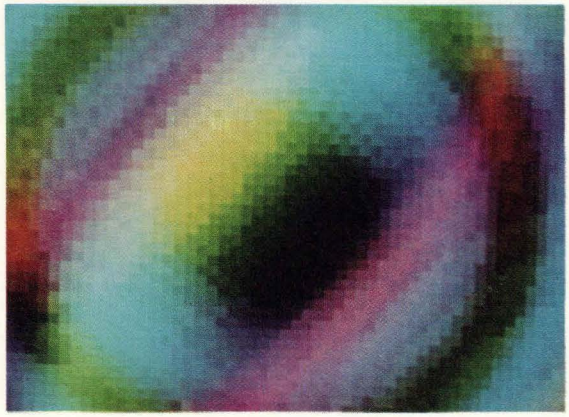
Repetition of triangle accentuated by the variation of a color component from 0 to 15 parallel to a large side of the triangle.



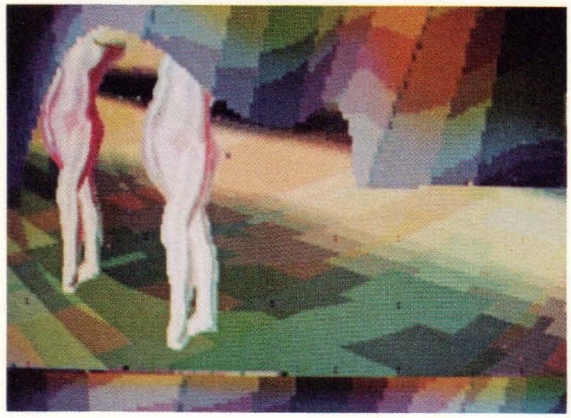
Example of surfaces filled by successive triangulations: one angle of the triangle is fixed, the other two recline on a central curve.



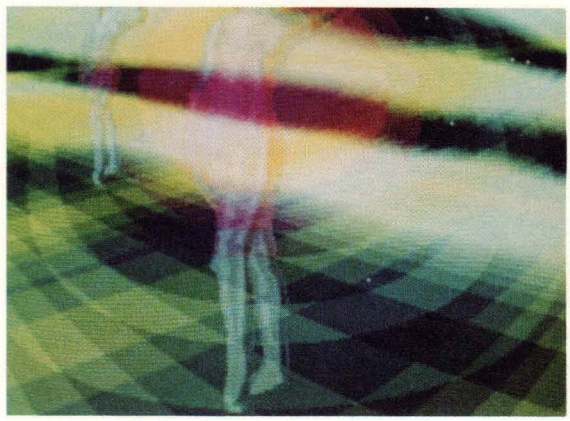
Effects of folding.



Color peaks



Images constructed using B-splines



Example of a transparency

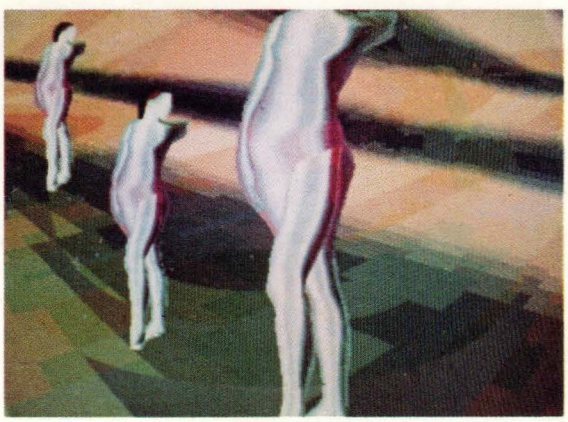


Image constructed using B-splines

Hand constructed using B-splines

PROCEEDINGS

symposium on small computers in the arts

IEEE Catalog No. 81CH1721-0
Library of Congress No. 81-84728
IEEE Computer Society Catalog No. 393

IEEE COMPUTER SOCIETY PRESS