JN4.GESTURE: AN INTERACTIVE COMPOSITION FOR DANCE

DOUGLAS B. HOLMES, B.M., M.M.

Dissertation Prepared for the Degree of

DOCTOR OF MUSICAL ARTS

UNIVERSITY OF NORTH TEXAS

May 2003

APPROVED:

Jon Christopher Nelson, Major Professor
Susan Cheal, Minor Professor
Thomas Clark, Committee Member
James C. Scott, Dean of the College of Music
C. Neal Tate, Dean of the Robert B. Toulouse
School of Graduate Studies

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure Page

CHAPTER 1

INTRODUCTION

jn4.gesture is a composition designed to extend the possibilities of artistic expression through the use of contemporary technology. The work is an interactive multimedia composition for dancer and computer. The performance utilizes audiovisual materials that are controlled by the movement of the dancer on a custom rug sensor through the use of a score run by a computer program. The dancer's movements influence the audiovisual and musical elements and determine the form of the work. It is this communication of the dancer's expression to the computer that will control the multi-faceted elements. The performance of the work represents collaboration between the dancer, composer and the computer technician. At the time of performance, the interaction between the technician and the dancer will define certain parameters of the compositional form as well as the audiovisual elements. The composition is designed to be an intermedia artwork combining all of the audiovisual elements into an indivisible artistic creation that comes to life at the time of performance.

The work is designed for performance in a specialized black box performance space with surround sound, thus creating an isolated sonic and visual environment. The computer program running the composition has been created to control all aspects of the audiovisual presentation through time. The sonic elements, including the mixing of the audio signals, are synthesized, filtered and re-mixed in reaction to the dancer's choreographed movements. Simultaneously, the visual material is projected onto the dancer. Movement also governs the placement, speed and depth of the graphic material. The graphic elements follow and are focused on the dancer, reducing the amount of unwanted light in the projection field. The dancer's movements cause the excitation of these graphic elements and at the same time the dancer is the canvas on which they are displayed.

The composition uses a Power Macintosh®[1] G4/ dual 500MHz computer. The computer is supplied with two video cards and a digital audio interface. The audio material is digitally produced in real time at 16 bit, 44100 samples per second in eight discreet channels. Two computer projectors present the visual material. Each projector is mounted on a 10-foot custom projector stand. The projectors are connected to one of the video cards in parallel and therefore shine the same material onto the dancer. A distance, however, separates the projectors from each other so they project material directly onto the dancer from two different perspectives. The remaining video card is used to present a computer interface to an audio technician who controls the overall flow of the composition using a customized graphic score. The composition has many indeterminant features and therefore each performance will differ. The approximate duration of the work is 10 to 15 minutes.

The theatre in which the work is presented must be designed with the proper light and sound absorption material. The work was conceived for a space like the Merrill Ellis Intermedia Theatre (MEIT) at the University of North Texas. The dance is performed in front of the audience so more traditional performance spaces can also support the work. The theatre, however, should be as dark as possible during the work's performance. Thick black curtains covering the structure from ceiling to floor can be used for light and sound absorption. The wall height, floor covering and ceiling color will also effect the absorption qualities associated with the performance space. When the visuals are projected in the total darkness of a structure like the MEIT, the images are vivid and surreal as they reflect off the dancer's outfit. However, if

[1] Power Macintosh is a trademark of Apple Computer, Inc.

there is any non-projected light in the auditorium or too much reflection from the projection on other surfaces, the richness of color and subtlety of graphic design are diminished.

A visual environment is formed with the generation of real-time graphic patterns and three-dimensional movies that are projected directly onto the dancer. The visual elements are designed to reflect solely on the dancer and not on any of the physical surroundings. In order to cut down on unwanted light, the program actually adjusts the area in which the projection is focused based on the sensor data. The two projectors shine the visual material directly onto the dancer, whose costumes are designed to take advantage of this light source. The costumes were designed to control the size of the projection termination. In other words, to adjust the size of and volume of the dancer. Visual contrast is supplemented by the two costumes that each react to the projections uniquely. Initially a costume with fabric that clings tightly to the body of the dancer is worn, and as the work progresses, a kimono-like costume is added over the original costume. The kimono is made of translucent fabric draped over the arms of the dancer. The graphic material is visible on both sides of the kimono fabric, adding dimension to the moving projection as can be seen in figure 1.

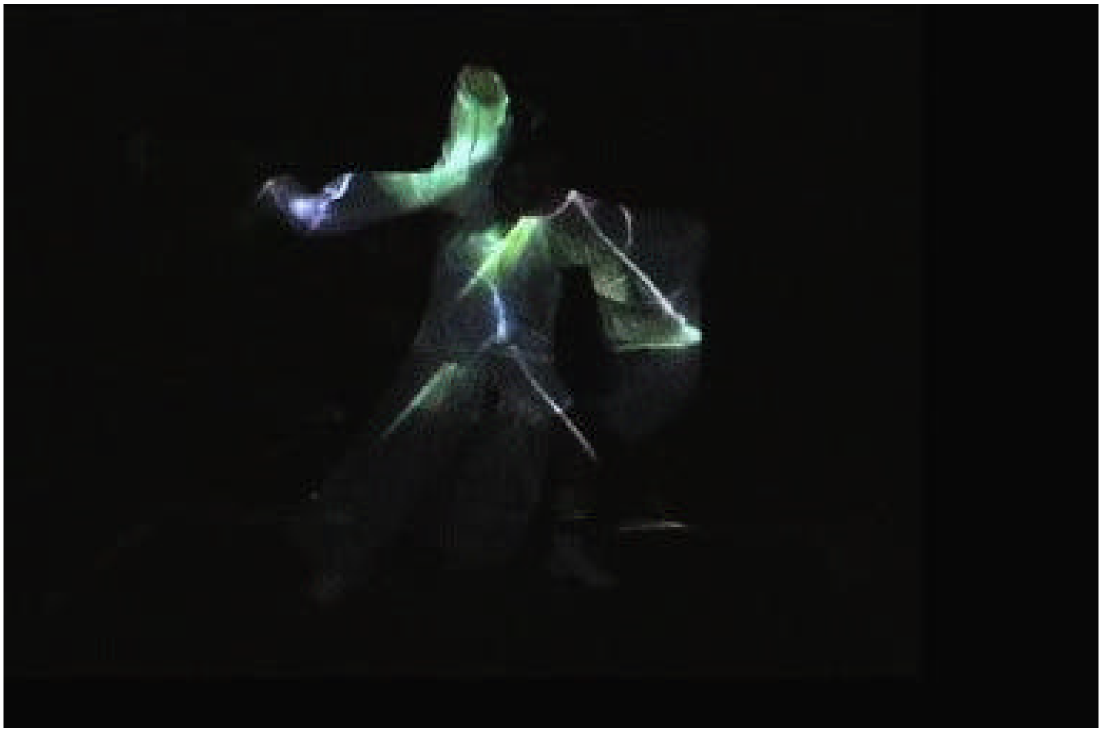


Fig. 1. Dancer as the screen, photo of Arleen Sugano during the rehearsal of jn4.gesture April 2002.

At the same time as the visual presentation, a sonic environment is created that is controlled and influenced by the motion of dance. The sonic elements are presented in a surround sound design controlled by the computer program in relation to the sensor-input data in real-time. The arrangement of the loudspeakers is important in the creation of the aural space. They are to be placed in an octagon shape surrounding the audience. The eight outputs from the computer hardware are bussed to the correct speaker by using a mixing console. The outputs from the computer are arranged in stereo pairs (1 = L, 2 = R, 3 = L, etc.). The loudspeaker arrangement has been designed to produce maximum separation of the multiple stereo fields. Each output signal is also bussed to a bass bin. The sound reinforcement design is utilized by internal computer routines that control the routing of the audio signals in the digital domain. The output from the computer creates stereo, quadraphonic and octaphonic fields based on this loudspeaker arrangement.

The audience must be seated in relationship to the performer and the offset stereo fields. Figure 2 shows the sound reinforcement design utilized for the performance of the work in the MEIT. Multiple stereo fields are created between each of the consecutive odd and even speaker numbers. The computer controls the placement of the monophonic source materials in these stereo fields. The panning of sound between each of the speakers is subject to the movement of the performer. In this way, the panning routines place monophonic sound sources within an octaphonic field. A quadraphonic field is also made between the speakers 5, 8, 7 and 6. The pre-composed quadraphonic sound files that are used to give structural cues to the performer are designed to utilize this quadraphonic field.

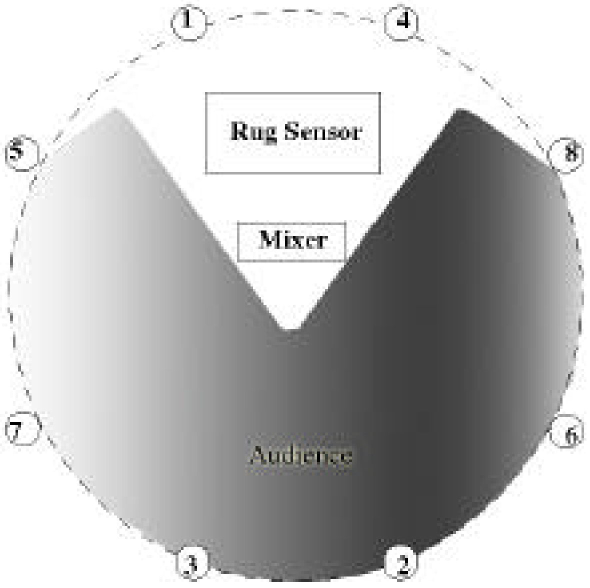


Fig. 2. Auditorium setup of the MEIT shows the loudspeaker arrangement.

As the imagery follows the dancer in the projection field and the sonic elements react to these same movements, an interactive audiovisual environment is created. The composition, in the form of a computer program, is in charge of controlling this environment through time. This program manages the mapping of sensor data and the choice of computer function by means of a graphical user interface (GUI), the score. By stepping through the GUI score, commands are sent to the computer modules in an orderly fashion. The score visually represents the introduction and three major section of the work on a computer monitor. And, through interactive communication at the time of performance, the dancer and the technician progress through sections of the score as “scenes” of the composition. Therefore, the duration of the work, sectional tempo and pacing, and the sonic and visual presentations are governed by the performance.

CHAPTER 2

GESTURE CAPTURE

Background and Significance

The integration of electronic music with live performance in the past has traditionally been one of limited interactivity. One need only examine a composition created for tape and performer to reveal this lack of interaction. The performer must follow the tape and a human technician must be used to achieve any electronic reactionary elements in the composition. The works titled *Synchronism* (nos. 1 to 7) by Mario Davidovsky are exemplary of these concerns for integration of the media, expanding the interaction between the performer and the recorded electronic material. Paradigms for such performances, indeed, are not as limited when using a computer. Program design can specify that the pitch or amplitude of the performer be tracked and used to initiate composed material. This type of programming combines the musical material being presented in an integrated fashion unbinding the flow of time through the compositional material. The performer, thus, has more control over the recorded or synthesized sonic material during the realization of the work. The computer can be programmed to adjust tempo, source material and timbre in reaction to the material being performed by the human counterpart.

The electronic music industry, early in the 1980s,[2] standardized a communication protocol for the exchange of data that is used to control electronic musical hardware and interfaces. The musical instrument digital interface (MIDI) soon became a common device used to connect electronic musical devices to the serial port of the computer. MIDI protocol is a useful communication standard for mapping small and fine movements associated with many

[2] Curtis Roads. The Computer Music Tutorial (Massachusetts: Massachusetts Institute of Technology, 1996), 974.

traditional instrument designs to musical parameters of computer function. Retail keyboards, wind controllers, percussion sensors and string-based instruments are designed to send MIDI note and controller data in a specific format. The accuracy and depth at which these devices capture the movement of the performer is proportionally related to their cost.

Alternative capture designs have been developed as practical tools in other branches concerned with computer interaction. The motion picture and gaming industries as well as industrial and medical fields utilize movement tracking for specific tools. Scenarios governing the capture of movement are as varied as their application. The interfaces used to translate movement to the computer have limitations related to their capture, sensor and data communication strategies. For instance, keyboard controllers are stationary interfaces and are further evaluated by the number of keys and controller devices, such as pedals and buttons, that are utilized in their design. The device must also be attached to the computer in some way. Radio frequency (RF) transponder and receiver units can be incorporated severing the umbilical connection between interface and computer but they also add other variables to the accuracy of communication.

The data, once captured, must then be assigned a functional role in the programming of the computer application. The relation of movement and gesture mapping for the creation of sonic material has received much attention in the past by academic institutions.[3] Using an interface for the reproduction of traditional instrumental techniques and using gesture associated with traditional performance is one area of research. However, creating a new sensor-based instrument for non-traditional application demands alternative approaches to capture a

[3] Academic institutions supply the resources allowing individual scholars the means to explore the possibilities of developing computer hardware and software from the theories and models that are created.

performer’s movement. In the expression of dance, for instance, one must combine fine motor movement with more grand gestures in order to create the art. This movement is rhythmical, phrased and potentially lyrical and presents an opportunity to utilize these assets to control musical elements compositionally.

State of Research

The use of the computer in relation to our everyday activities is ever increasing. The ability for humans to interact seamlessly and accurately with programs represents an integrated tool that greatly increases the usefulness of the computer. General inquiry into this area of study is called human-computer interface (HCI). HCI is concerned with the application and development of hardware and software that are common to the concerns of the computer music composer. Research including scientific studies showing the relation between device and software designs and physical and mental functions provides analysis of various interactive paradigms. Studies in the field of HCI that examine GUI and software development, which are directly influenced by user input, can be useful in the design and functional programming for the creation of computer-produced artwork. Although these studies rarely focus on artistic concepts, the information is contemporary and influential in the design of artistic computer-generated environments. Many journals and other resources are available on-line that post and supply contemporary research on HCI at a high scholarly level.

Since the turn of the century, panels of scholars have been assigned by leading international societies and groups in the field of computer music to address the possibilities of HCI research that directly relate to the creation of musical art. The International Computer

Music Association has assigned a workgroup on computer interaction and performance[4] including a subcategory on interactive systems for dance. The Institut de Recherch et Coordination Acoustique/Musique (IRCAM), Centre Pompidou, France has published an electronic source titled Trends in Gestural Control of Music.[5] This source is in the form of an interactive compact disk, representing a compilation of research and essays relating to gesture capture for a musical end.

A large variety of research tools and results are available on the Internet providing information on computer interaction and the arts. For instance, the Dance and Technology Zone[6] (DTZ) is an annotated bibliographic source that has information directly relating to the subject of dance and technology in an electronic format. DTZ is maintained by art.net[7] and supplies links that point to a wide range of electronic books, papers, journals, monographs and essays that relate to the issues surrounding the use of media for production of dance and related live performance. CVonline©[8] is another valuable electronic source. The editor, Robert B. Fisher describes it as; "The Evolving, Distributed, Non-Proprietary, On-Line Compendium of Computer Vision." The Division of Informatics and Artificial Intelligence,[9] University of Edinburgh, hosts this site.

Professional dance groups and companies have been drawn to the possibilities of controlling computer-generated material through movement. These groups and companies utilize a variety of gesture capture paradigms to create interactive compositions. The Palindrome

[4] The International Computer Music Association Workgroup on Computer Interaction and Performance. URL: http://www.computermusic.org/interactivesystems/wg.html

[5] M. Wanderley, and M. Battier, eds. Trends in Gestural Control of Music. Ircam - Centre Pompidou, 2000.

[6] Dance and Technology Zone. URL: http://art.net/resources/dtz/theory.html

[7] Art on the Net. URL: http://art.net

[8] CVonline© 2002 Robert Fisher. URL: http://www.dai.ed.ac.uk/CVonline/

[9] Division of Informatics and Artificial Intelligence, University of Edinburgh. URL: http:// www.dai.ed.ac.uk/

Intermedia Performance Group[10] actually markets the hardware and software that they develop and use. Palindrome incorporates video tracking systems as well as a suit of electrodes and sensors providing muscle capture. The EyesWeb Project[11] in Italy provides gesture capture devices that have been utilized for many different artistic and educational applications. The EyesWeb Project includes both research and development for application-oriented activities.

Tracking systems

The capture of human gesture for the control of computer application is one of the main concerns in the field of human-computer interface (HCI). Gesture capture focuses on the accuracy of data acquisition, the relation between movement and function, the statistical possibilities of repetition of gesture, and how the gesture itself is mapped to the programmed interface. The form of the device must be well matched to the functional role of the system. Devices and scenarios used for the capture of human gesture are diverse.

Alex Multer, in a technical report "Human movement tracking technology," describes a system of specifying human movement tracking systems. He suggests that the generalization of tracking systems can be grouped into three categories: inside-in,[12] inside-out[13] and outside-in[14]. Each of these categories is optimal for the capture of certain kinds of movement.

Inside-in systems employ sensors and sources that are attached to the body and are well suited to make the measurement of fine movement. An exemplary device used for movement

[10] Palindrome Dance Company, Inc. /e.V. Johannisstr. 42, 90419 Nürnberg, Germany). URL: http://www.palindrome.de/.

[11] InfoMus - Laboratorio di Informatica Musicale. URL: http://musart.dist.unige.it/sito_inglese/research/r_current/eyesweb.html

[12] Axel Mulder. Human movement tracking technology. (Technical report 94-1 Hand centered studies of human movement Project., Copyright 1994 Simon Fraser University.), 4.

[13] Ibid., 5.

[14] Ibid., 7.

tracking in this category is an interface glove. The input device in this scenario is worn on the hand and the sensors are arranged to take measurements of specific joints. The data is then relative to the source being used, in this case the hand. Tracking the movement of the finger joints in relationship to the palm is therefore a good example of an inside-in movement tracking system. Inside-in tracking systems generally do not provide three-dimensional movement tracking.

Inside-out systems are designed with sensors on the body that measure the body's movement in relation to an external source. The measurement of movement in this category, uses artificial external sources (electromagnetic fields) and natural external sources (reference a stable object or a stable force like gravity). The sensors in these scenarios relate body movement to a fixed reference. Examples of this category of device are accelerometers (gravitational field) and hall-effect (electromagnetic fields) sensors. Inside-out systems can produce three-dimensional references and are well suited for the measurement of medium and large body parts.

Outside-in systems on the other hand are designed using external sensors that measure an artificial source. As the artificial source moves past the sensor the value is forced to change. A simple example of an outside-in system would be a computer keyboard. The sensors in this category have a restricted field in which measurement of movement can be made. The subject's movement is relative to the stable sensor. Outside-in tracking is considered to be the least obtrusive form of input because the sensor is not physically attached to the subject. Electro-optical tracking systems are included in this category of movement tracking. Electro-optical systems can be as simple as a phototransistor translating light values into electric current or as sophisticated as video tracking systems.

The Rug

The rug sensor is a hand-made contact sensor device created by the composer. The dimensions of the rug sensor are five feet seven inches by eight feet six inches. The rug sensor is an outside-in tracking system according to Alex Multer's categorization of tracking paradigms. The individual sensors are normally in an open state and are closed when pressure is applied to the surface of the carpet immediately above the sensor. The sensors themselves run along the total horizontal and vertical axis of the carpet setting up an x y grid. There are 7 (y) by 9 (x) sensors beneath the rug (see fig. 3). Pressure on the surface of the carpet causes the triggered sensors to send a "down state" and trigger identity to the computer.

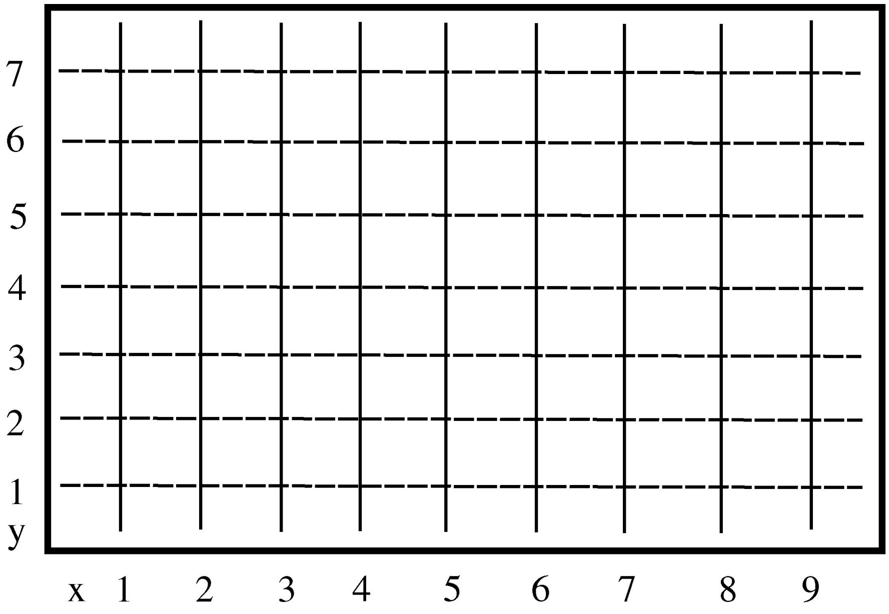


Fig. 3. The sensor construction of the rug showing the triggers running the length and width of the carpet creating a grid of x and y coordinates.

The data, in the form of serial ASCII code, is sent to the computer via a USB port. Depending on the programming, each of the trigger states can be tracked and timed. The 16 sensors create an x-y grid at a resolution of 63 crossings. The incoming data from the carpet is mapped to computer related functions that control the audiovisual data and their presentation. Scenarios relating trigger states to the gestural movement are manipulated to give multi-dimensional control over mapped data to the performer. One progressive movement by the dancer is interpreted differently by any of the active audiovisual modules. For instance, the outer and center regions of the carpet can be differentiated and the triggers will control various aspects of sound reproduction or diffusion while at the same time proximity to the audience will control some sort of visual aspect such as size or speed of animation.

Scenarios are created to relate timbre and pitch or refresh rate and graphic elements to the dancer's speed and direction of movement. The parameters associated with an amplitude envelope, for example, can be directly controlled by the individual trigger states. The attack and release of an envelope is mapped in a direct relation to the down or up state of an individual trigger, while the speed of file playback is controlled by the duration of the down state from a group of sensors. Furthermore, the control over audiovisual elements can be taken away or disassociated from movement at any given time. The dancer then, through instruction, improvisation and practice, can intuitively expect certain reactions to gesture that are related to audiovisual material and the progression of the compositional form. Audiovisual cues associated with formal and sectional structure will aid in the association of movement and gesture to compositional elements.

CHAPTER 3

PROGRAMMING

General Information

The computer program controlling the audiovisual elements was created using the MAX©[15] object-oriented programming language, developed by Opcode Systems, inc. and IRCAM. MAX/MSP©[16] is an integrated program by Miller Puckette and David Zicarelli that enables the user to create patches from programming objects that function at the control or audio rate of the computer system. The software is available from cycling'74[17] for the Macintosh platform. MAX/MSP can utilize MIDI, Ethernet, video, and serial data ports to control the input and output of information enabling communication with exterior multimedia-hardware. MSP is based on the programs Max/FTS, and the Advanced Digital Signal Processing Platform from IRCAM, which allows the input, output and processing of digital information at an audio rate. Exterior hardware converts the audio-rate data from digital to analogue form and vice versa. Using external hardware like the 888/20 digital audio interface by Digidesign,[18] an eight-channel design can be created. The programming in *jn4.gesture* calls for an eight-channel audio output design. The media output is controlled by a group of max patches that are designed to serve specific audiovisual functions.

[15] MAX copyright © 1900-1997 Opcode systems/IRCAM.

[16] MSP © 1997 David Zicarelli—All rights reserved based on Pd by Miller Puckette © 1997 The Regents of the University of California MSP and Pd are based on ideas in Max/FTS, an advanced DSP platform © IRCAM.

[17] Cycling '74, 1186 Folsom Street San Francisco, CA 94103 USA. (415) 621-5743. fax (415) 621-6563. info@cycling74.com. URL http://www.cycling74.com

[18] Digidesign, a Division of Avid Technology, Inc.

Compositional Construction

The composition was constructed using a modular programming approach. Audiovisual stimulus is achieved with the control of the various media, in reaction to the sensor data, with a specific output role. These modules are intended to produce discreet audiovisual elements whose qualities are varied and triggered by the sensor data mapping. All of the modules are designed to interpret the generic mapping parameters provided as output from the mapping routines. The modules are dynamically called into use, and their variable parameters are adjusted by a system of scripting language (lingo) developed for the composition. Using lingo script that is sent in a timely manner, the role of the incoming control data from the rug sensor is changed within the various audiovisual modules.

Each module is re-initialized and its parameters are adjusted over the duration of the composition by sending global messages in composed intervals. The sensor data can therefore be used to control a variety of audiovisual modules, and their control mapping can be adjusted through time. This application of lingo scripting allows for variation in mapping of control functions to the movements of the performer over the duration of the composition.

The sensor data is organized by routines that scale and group the information making it available as generic mapping parameters. The mapping routines sort, group and time the sensor data to provide specific output parameters that are always available to the modules. The audiovisual modules look for a specific data type from these generic-mapping parameters in the global message stream. The sensor data in this generic form is useful in creating a variation in a causal relationship with the dancer’s actions. The sensor data is used directly and indirectly to control aspects of the composition. The state of the activity for each of the sensors is measured. When the sensor is closed by pressure, the program measures the duration of time that pressure remains. As soon as the pressure is removed and the sensor is open the program concludes the timing measurement. The data is then subject to filtering and sorting before it is turned into a global message. This generic data is individually interpreted and adjusted further by each of the audiovisual modules as described by their programming.

The direct data from sensors can be erratic because of the construction of the conductive elements in the rug. For this reason, a more advanced filtering routine is occasionally needed to create stable data for the control of sensitive audiovisual elements. If the pressure on top of the sensor is not direct, the data can show repetitive open and closed states at a sporadic rate. For these reasons repetition filters and timing restrictions are built into the sensor data routines at various levels.

Unique filters and routines are created within an audiovisual module when a sensitive parameter requires that the mapping variables be reinterpreted for its own individualized goal. For instance, subgroups of the generic data are used to differentiate between movement in confined and unconfined areas on the carpet. This type of filtering routine is used to control the re-draw rate at which graphic elements are removed from the projection window. If the dancer triggers sensors in close proximity to one another the graphic elements are allowed to build up and create a composite design. But when the dancer moves a great distance, the design elements in the previous projection quadrants are removed and the projection focus is moved to the current position of the dancer.

Mapping Categories

Two groups of sensors are formed that are distinguishable by grid axis (x, y). Within each of the groups the trigger-id numbers are re-assigned to an increment which sets its position within the group. Figure 4 shows the x and y sensor groups on the rug. The sensor data for each of these groups form a unit in which the sensor activity is a relative action within that group.

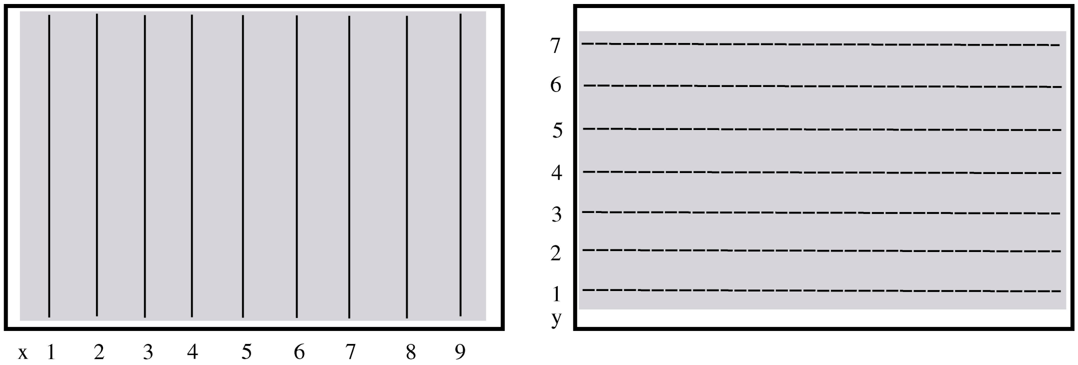


Fig. 4. The x and y sensor groupings shown on the rug.

The input data is filtered in a sieve or linear event-paradigm (see fig. 5). The direct sensor data is sent through these filters in order to determine which sensor within the group is active. By passing the data down a branch until a logical selection has passed, only the selected trigger states are measured. Each consecutive action within the group is viewed as a linear succession.

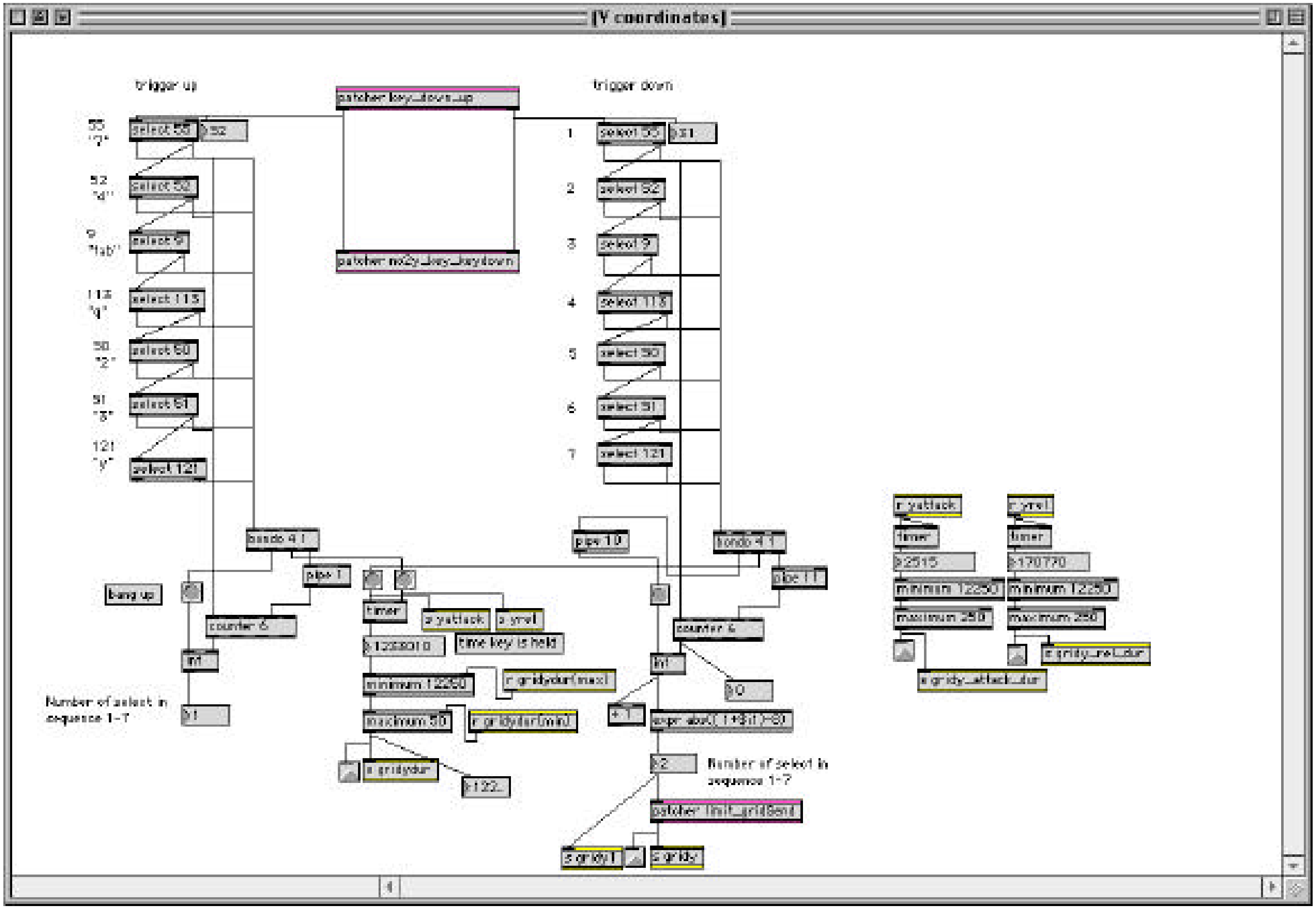


Fig. 5. Linear sieve, code used to group the y-axis.

For instance, when one jumps on the rug and both feet simultaneously trigger separate sensors in the same group, the program using this routine will interpret the event as two separate occurrences, one closely following the other. The entire group is used to determine the duration between activity within the group. The duration between initial and successive down and up states for the group is calculated. Each of these products can be mapped as input parameters used by the audiovisual modules.

Figure 6 shows the sensor grid area re-scaled to floating point numbers. Each x or y group forms a unit in which the sensor's position in the respective group is scaled to a number between 0.0 and 1.0. A line segment is used to make a smooth transition between points in the scaled sensor groups. In this way, the output data from this mapping routine provides a stream of incremental numbers connecting the points provided by the sensors. This data can be readily re-scaled to various input parameters and used for signal modification.

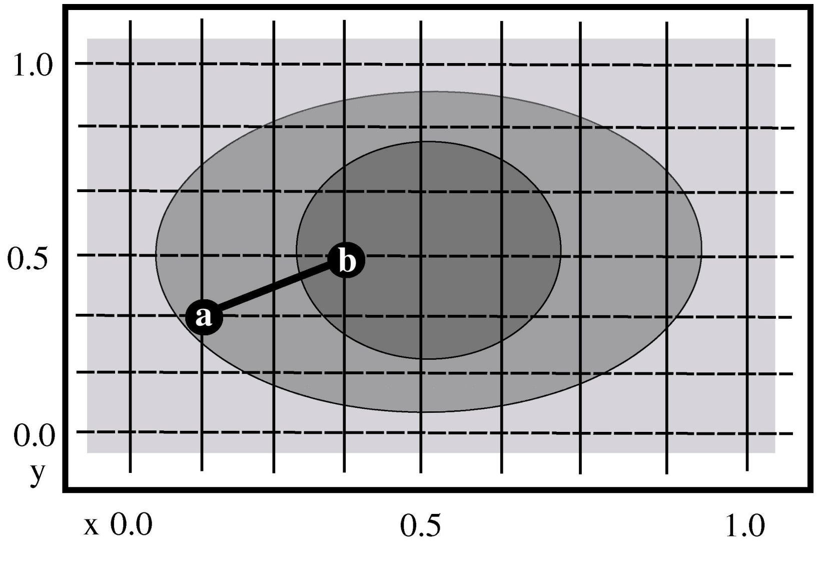
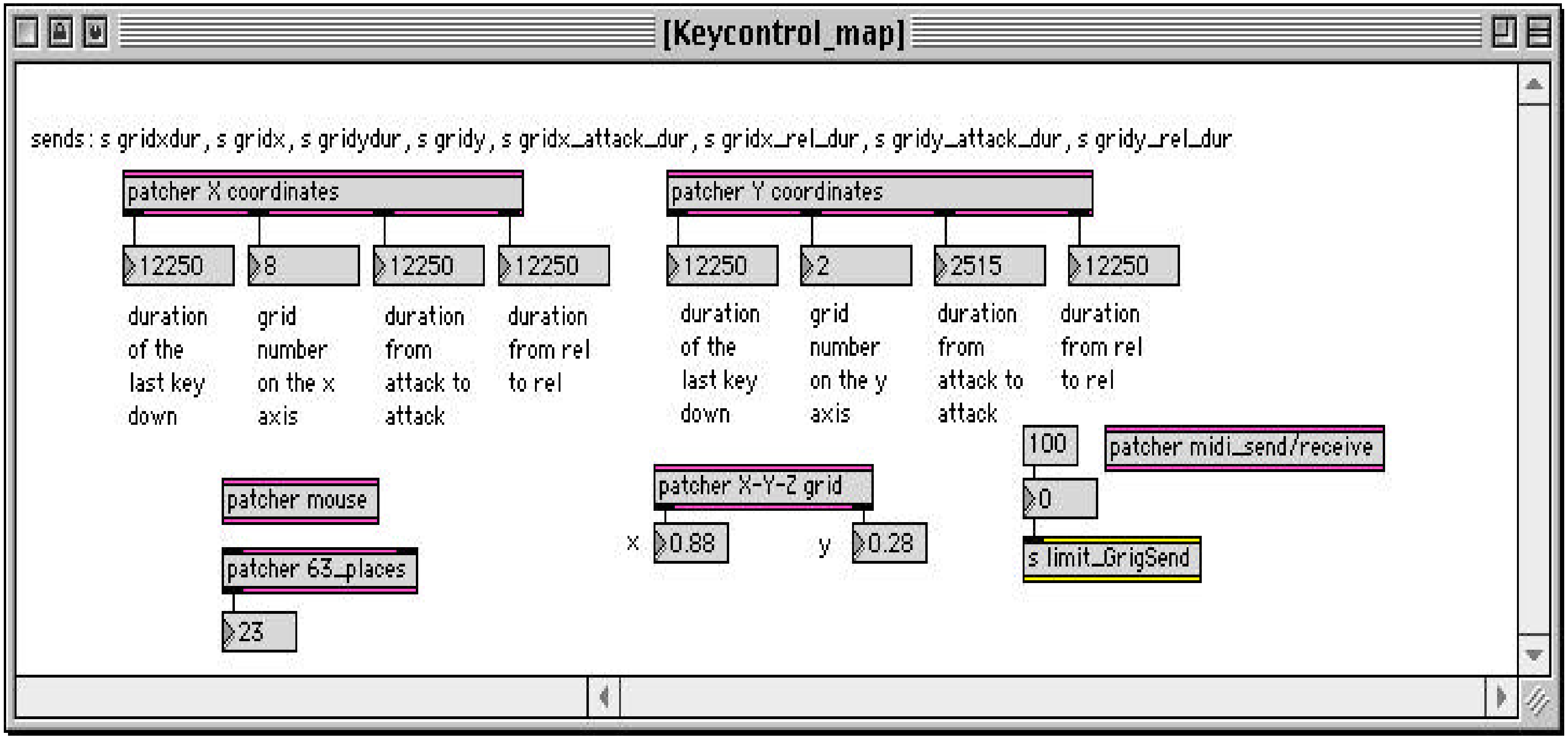


Fig. 6. The scaled sensor groups are used to track movement, the change from point a to point b is connected with a line segment provided by the mapping module. Angled connections between points are therefore described with a linear succession of floating point numbers gapping the distance between the sensors from each group.

The sensor grid information is also interpreted on an individual fashion and used for the direct control over the audiovisual elements. Each of the triggers is mapped directly to an identification number. In this way, if two feet were to land in unison at separate coordinates on the carpet, the data would be interpreted as separate incidences happening simultaneously. The integers 1-9 and 11-17 are used to identify each of the separate triggers. These identification integers are then combined to calculate crossings or points on the x-y grid. Sixty-three different crossing points on the rug are identified in this way. The sensor data in each of these forms is continuously available to the media modules for the mapping of control functions. Figure 7 shows the window that displays the global mapping data to the technician.



sends: s gridxdur, s gridx, s gridydur, s gridy, s gridx_attack_dur, s gridx_rel_dur, s gridy_attack_dur, s gridy_rel_dur

Fig. 7. Window used to display the sensor data.

CHAPTER 4

FORM AND TIME

Time Management

The overall flow of the composition is controlled by means of a hybrid language script created to control the initialization, data mapping, file management and parameter adjustment of the media modules through time. A network of global receivers is fed this lingo script and the individual modules look for their particular data in the global stream. The progress of the composition is directly related to the progress of the lingo script. This script is read by a MAX object called Qlist (version 0.02) which was created by Miller Puckette. The object has been programmed to read a specific number of lines from the script at a controllable rate of progress. The script-lines are organized into groups and sub groups that ultimately govern the form of the composition.

The modules themselves have a certain predetermined duration of execution and must terminate processes before they can be successfully adjusted without injecting noise elements into the audiovisuals. In some cases, a module that reads audio or video files must first properly dispose of old data before new images or sounds can be loaded. Many of these files are loaded directly into random access memory so care must be taken in their management to avoid a crash or lockup of the computer functions. The management of CPU utilization is also controlled by using the script in this timely manner. The allocation of CPU time is made more efficient by rendering any audio rate processes dormant when the module is not being used. Because of the size, depth and number of audiovisual processes this CPU time management is essential.

To successively turn off the audio processes in a module without injecting unwanted sounds into the audio stream, the output signal from the module is supplied a line segment starting at the functioning line level and terminating at zero. After the audio signal has been properly attenuated, the object can be successively turned off and the CPU allocation is freed. This control over the output signal is specified by the lingo script and is predetermined to function at a specified rate when the command to deactivate a module is given. This command to deactivate a module is often given when the dancer is in progression. Each step from the dancer will continually trigger new events in any of the modules that are active. So, when a module receives the command to deactivate, chances are that the dancer's movement has just initiated or started a sonic element. If the signals were cut off at this time a noticeable click or static tone would be created. For this reason, the command to deactivate a module first supplies a line segment that attenuates any audio signals that are being produced. Only after the module's signal fades to zero, is the command to turn off the audio functions given, freeing up the allocation of CPU time associated with the module.

The audio rate objects are turned on in a similar fashion; a line segment starting at zero will terminate at the module's normal operational level. This normal operation value is a stored variable that can be adjusted by using the mixing consoles. These windows give the audio technician control over the ultimate mixing of every audio signal in the digital domain, and supply the settings as the normal operational levels used for automated attenuation during deactivation or activation of the audio modules. Figure 8 shows a window used to set and store the values for the effect sends signals. The ultimate mix of the audio signals can therefor be changed to fine tune the sonic environment in various performance spaces without having to reprogram the lingo script.

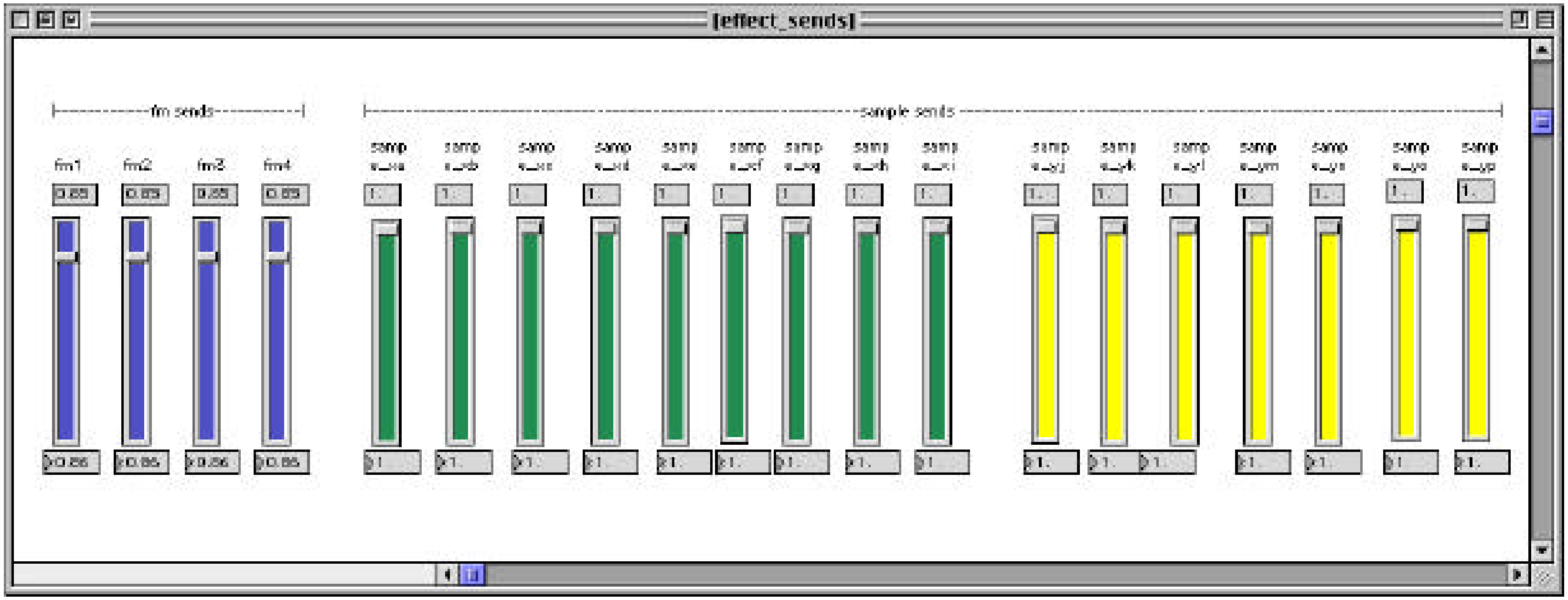


Fig. 8. Sends console, one of the adjustable modules for the audio rate signals.

Score Interface

The lingo script under the supervision of a graphical user interface (GUI) provides the control over each of the audiovisual modules during performance of the work. Because ASCII code is being used to communicate the sensor data to the computer, typed keyboard input is not desirable. A mouse-based interface was developed to ensure accurate and segregate user control. Feedback, in the form of colored buttons and numeric output, guides the engineer through the composition (see fig. 9). Generally, the introduction, which lasts approximately 30 seconds, is followed by three major sections that last about 3 minutes each. The GUI score shows the major formal divisions stacked in descending order on the screen. The three major sections, represented in the GUI, are further graphically subdivided. These buttons control the distribution of the lingo script, and an engaged button causes a number of specific script-lines to be read at a set pace. The graphic symbols are not placed proportionally and do not represent a relationship between the duration of sectional material and their placement in the score.

A color scheme describes the progression within and through the formal divisions in time. The buttons are symbols and their placement relates their position within the compositional form. Initially the buttons are red, indicating that they are not prepared or ready to be activated. When green, the buttons show that the media modules have received the proper settings associated with the formal segment and are ready to be activated. While in operation the buttons are blue. The GUI’s colored buttons direct the user through the score in a set order at prescribed time frames. The technician should only initiate the “scenes” of the composition in this fashion during performance.

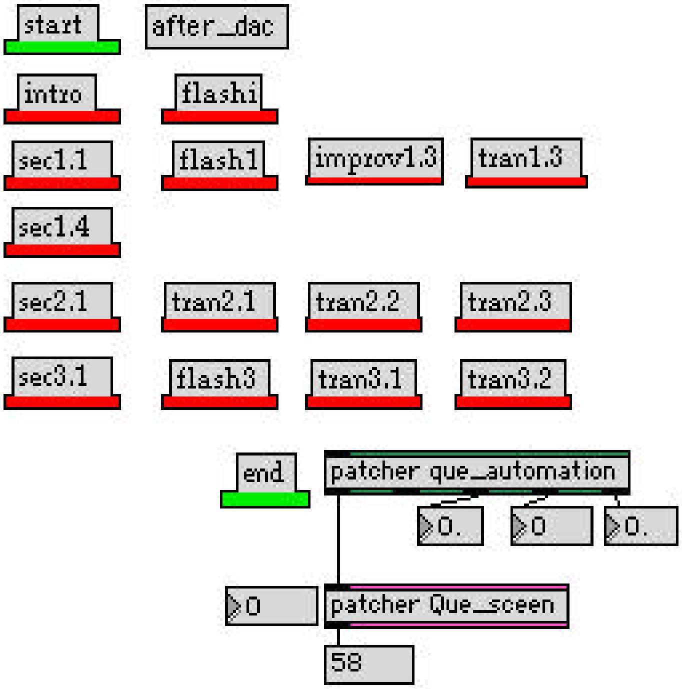


Fig. 9. The GUI score reset and ready for initialization.

The score is initialized when the patch is opened. Once initialized, only the start and end buttons are green. The end button always remains green and resets all of the program's default settings. Upon selecting the green start button the composition is set into motion. This turns on the digital audio converter (dac) object and loads the files that are used for the beginning of the composition. The projection field is darkened and all audio functions are also disabled at this point. When the start routine has finished these tasks, the intro button will turn green and the performance is ready to start. At this time the rug sensor is disassociated from the dancer's movements and therefore will not react during the entrance and positioning of the dancer. The audiovisual material is loaded and ready but will not start until the intro button is clicked.

Through time, each button or formal segment has multiple functions. Along with controlling active audiovisual module parameter adjustments the lingo associated with the button is also responsible for pre-initialization of the next formal segment. Lines of the lingo script are initiated at determinant times within the duration associated with each button segment. The color of the next button in the sequence is changed only after the active segment reaches a pre-calculated point in its progression. Specifically, button colors are green only after the sentence structures within the previous group has been properly managed and the pre-initialization of any new routines has been successively completed. The progress of the lingo script can also be tracked or followed on a line by line basis. The number below the "patcher Que_sceen" in figure 9 shows the current line of lingo being sent to the audiovisual modules. Once the tasks of the current group have been completed the next segment button will turn from red to green signifying that the next group is ready for use.

At this point in the composition some audiovisual routines are allowed to continue. However, as soon as the next button is depressed the new script terminates certain routines and initializes new ones. The termination of elements associated with the GUI score does not necessarily line up with the composition's formal subdivisions. The overlapping areas represent transitions, improvisational sections, extensions and general pauses within the overall formal outline. Many of these formal areas have been designed into the score to provide depth and indeterminant portions within the structural form. As the next button is clicked, the new formal segment is started with its own time format and the color scheme continues its forward progress. Only after the objective of the phrase group has been completed will the GUI score encourage the technician to continue.

Rehearsal Techniques

This form of compositional control, using time segments to manage the computer functions, presents a sectional rehearsal scenario, which represents a potential problem. To rehearse the work from beginning to end each time is not a desirable option. The GUI score addresses this problem by providing access to each section as a starting point after re-initialization by selecting the end button. This allows one to utilize section buttons in a manner that is analogous to orchestral rehearsal symbols in a score. At these points, the initialization and termination of module functions is logical and sufficient to be used as a starting point. The technician is also able to access other portions of the score between the major sections by manually running through the lines of lingo script out of time. By manually loading the script lines, the files and settings before the desired position will have been properly managed. This is done by using the variable number box, to the left of the “patcher Que_sceen,” on the GUI score (see fig. 9).

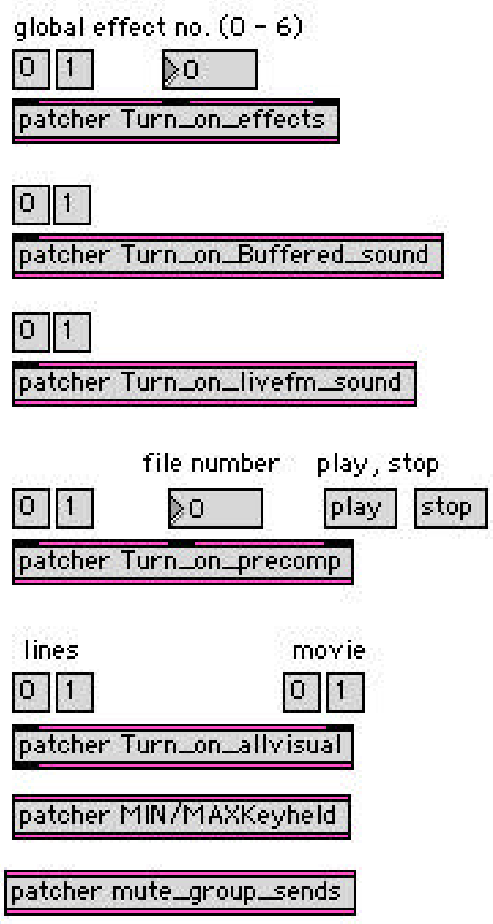


Fig. 10. Audiovisual module controllers provide the technician the ability to control the individual modules for rehearsal purposes.

Other controls have been created to aid the technician and the dancer during rehearsal. These controls directly configure the modules that are used in various sections. In this way the dancer and technician can explore combinations of modules through improvisation. Each of the control patchers shown in figure 10 can be opened revealing more parameter adjustments. This allows the dancer and technician to isolate the audiovisual modules, which function out of formal time, in order to get a good feel for the computer's reaction to gesture.

CHAPTER 5

MEDIA MODULES

Audiovisual Modules

The modules that are responsible for the creation of the audiovisual elements are actually smaller computer programs that can function as standalone software. Each of the modules is designed to create the media elements in relation to the score, movement on the carpet and any file management responsibilities it may have. In this way, the audiovisual modules are synchronized and work together; yet, they still have their own rate of progress, which give them something similar to a personality. The modules are designed to take information through input parameters and output new processed material. Other programmed routines then send the information to its ultimate destination, projection and/or sound production.

The modules process either visual or audio materials. The audio modules control the production, filtering and placement of sound in the 8-channel environment. These materials are created using digital synthesis techniques including frequency, amplitude, and ring modulation as well as filtering and variable-speed playback of audio files. The outputs are discrete audio signals that are sent to banks of effects and panning routines for their placement in the sonic environment. The visual material is created by programs that generate real time graphics and by other routines that read three-dimensional geometric renderings that have been saved as movie files. These modules create the visual elements that are then displayed through a unique window designed for output to the computer projectors. All of the audiovisual modules are under the direct supervision of the lingo script and are designed to be self-contained units that function independently and simultaneously with other modules.

Modulation Units

Four audio rate modulation units are utilized to create the synthesized sonic elements. The modulation units supply the panning and effects routines with four discrete channels of audio. The x-sensor group is used to control two of these modulation units and the y-sensor group is used to control the other two. The data from the sensors is used to control the amplitude envelopes as well as the modulation indices for non-linear synthesis. This data also controls the re-initialization parameters in relation to the other modulation units. This is achieved by synchronizing of initialization and parameter adjustment to a meter whose value is determined by the up and down states from each group of sensors. The dancer perceives this meter as the number of steps that it takes to conclude a sonic element.

The sensor data is reinterpreted to produce an adjustable succession of sensor states. The number of up states in relation to down states determines the variation in meter. After an initial down trigger the units ignore consecutive down states until a succession of up data has been completed. The sensor group can be expected to have a meter or a pace that is associated with an initial down state. The re-initialization of parameters effecting timbre and the choice of line segments used for modulation techniques is synchronized to this meter. In this way, the re-initialization can be scheduled to coincide with the end of an amplitude envelope and therefore be carried out during the attenuation of the audio signal.

The meter is reset through time by the lingo script. The meters' available range spans from one down state, which initiates an envelope with a determined duration, to a down state followed by as many as five up states. Figure 11 shows a graphic representation of sensor states in relationship to the number of steps needed to complete the meter. Just as a one feels the meter of a waltz, the meter from the dancer's steps will cause the creation of sound and a dynamic change of timbre.

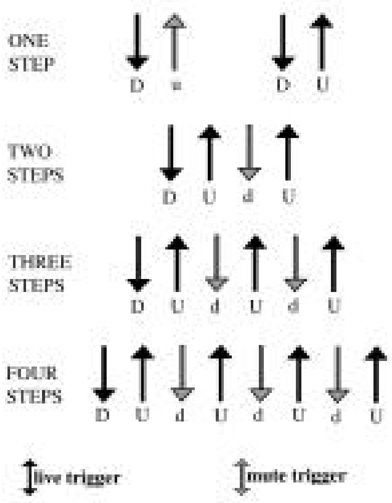


Fig. 11. A diagram showing the sensor states grouped into meters. Black arrows indicate that the direction is used to trigger events within the modulation units. A gray arrow indicates that the sensor event is disregarded.

The initial down state is associated with the attack of the sound and each up state initiates progression through the pre-designed line segments that are mapped to parameters controlling timbre. The last up trigger in the meter represents the release of the amplitude envelope. As the dancer moves on the rug, it is possible to displace the synchronization between the x and y modulation group meters. This adds to the texture and complexity of the sound. The units are, however, synchronized again when the meter is changed by the command of the lingo script. Using these techniques the envelope and timbre of the sonic material is controlled by the dancer's progression and tempo. The dancer is able to step and pause, causing the tones to be sustained. As the foot is lifted the sustained tone is adjusted further. The sustained tone is not static since its parameters are changed dynamically according to the modulation unit's settings.

Figure 12 shows one of the four modulation units. The audio rate line segments that drive the parameters controlling timbre and amplitude of the unit can be seen in this window. The breakpoint function editors, that drive the line segments, are adjusted using the preset table seen in the lower right hand corner of the window. The breakpoint settings have been pre-calculated and are stored in a text file that is read by the program at the time of start up. The progression through these settings is automated and synchronized to the sensor meter of the active modulation unit.

The breakpoint line segments provide a line-driven unit of numbers with a range from 0.0 to 1.0. The initialized parameters are then multiplied to the line segment, which provides a linear adjustment to the set-state of the unit. The number of breakpoints in the line segment are arranged and addressed according to the chosen meter. The line segment is read with the initial attack or down state associated with the specified meter. The line segment will reach its target and hold this setting until the segment continues when initiated with the next up sensor from the group. This process is continued until the meter has been successfully completed.

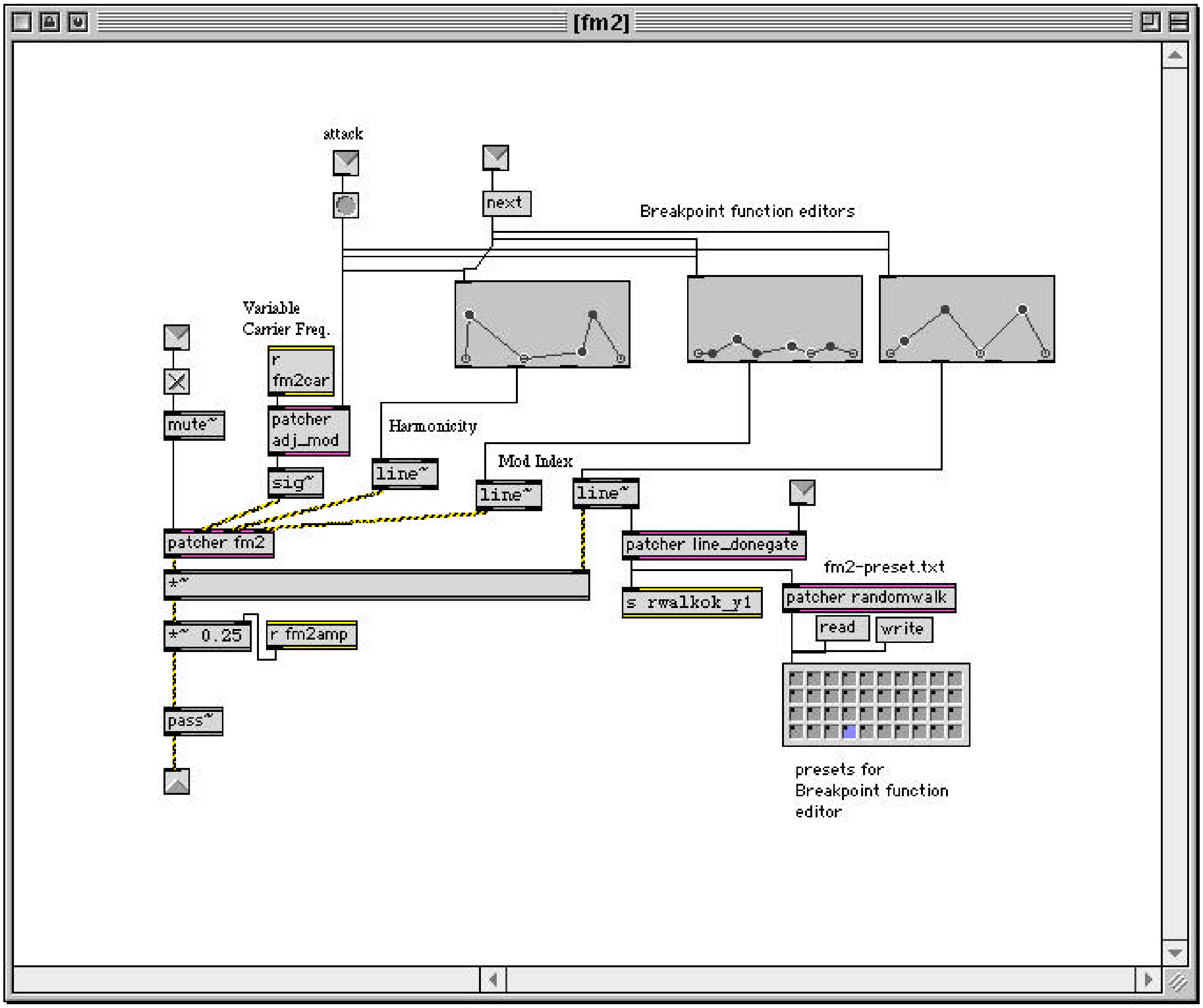
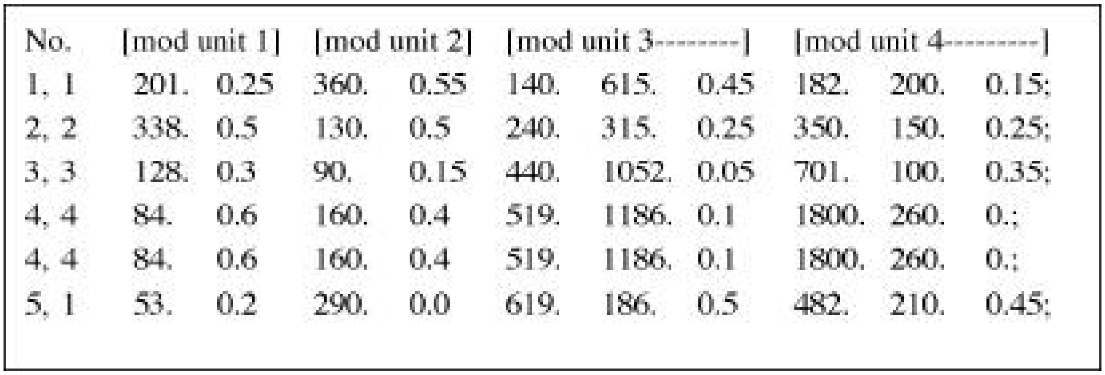


Fig. 12. The “fm2” modulation unit window housing the breakpoint function tables that are used to dynamically adjust of the modulation parameters between sensor activity. In the breakpoint editor windows, the points in the line segments surrounded by a white circle represent consecutive pausing points in the line segments progress.

When the meter has been completed the unit can be successively re-initialized without adding noise elements into the audio stream. The re-initialization parameters are pre-designed and placed in a text file that is read by the program at the time of startup. This file supplies each modulation unit with frequency, amplitude and other parameters that are used to create its initial pitch and timbre. Figure 13 shows a portion of the text file that is used to store these initialization settings. Using these pre-designed parameters a tuned unit of modulation generators is created that is changed as a consort over time.



```
No.    [mod unit 1]  [mod unit 2]  [mod unit 3--------]    [mod unit 4---------]
1, 1    201.  0.25   360.   0.55   140.   615.   0.45   182.   200.   0.15;
2, 2    338.  0.5    130.   0.5    240.   315.   0.25   350.   150.   0.25;
3, 3    128.  0.3    90.    0.15   440.   1052.  0.05   701.   100.   0.35;
4, 4    84.   0.6    160.   0.4    519.   1186.  0.1    1800.  260.   0.:;
4, 4    84.   0.6    160.   0.4    519.   1186.  0.1    1800.  260.   0.:;
5, 1    53.   0.2    290.   0.0    619.   186.   0.5    482.   210.   0.45;
```

Fig. 13. A portion of the text file containing the modulation unit's parameter settings. The text file is read line by line, and the modulation units are supplied the parameter settings that are related to one another.

These parameters are then chosen by a random walk routine that is scaled and timed to change according to the sensor meter. The output from this random operation is used to drive a routine that can read any line from the text file and store the information until the data can be utilized by the separate modules (see figure 14). In this way, the initialization of parameter settings for each of the modules is changed at the same instance, yet, the modules themselves are not adjusted until they have concluded their previous meter configuration when the meter of the modulation units are not synchronized. The modulation units are designed to operate in this way without disrupting any other audio functions that are happening simultaneously.

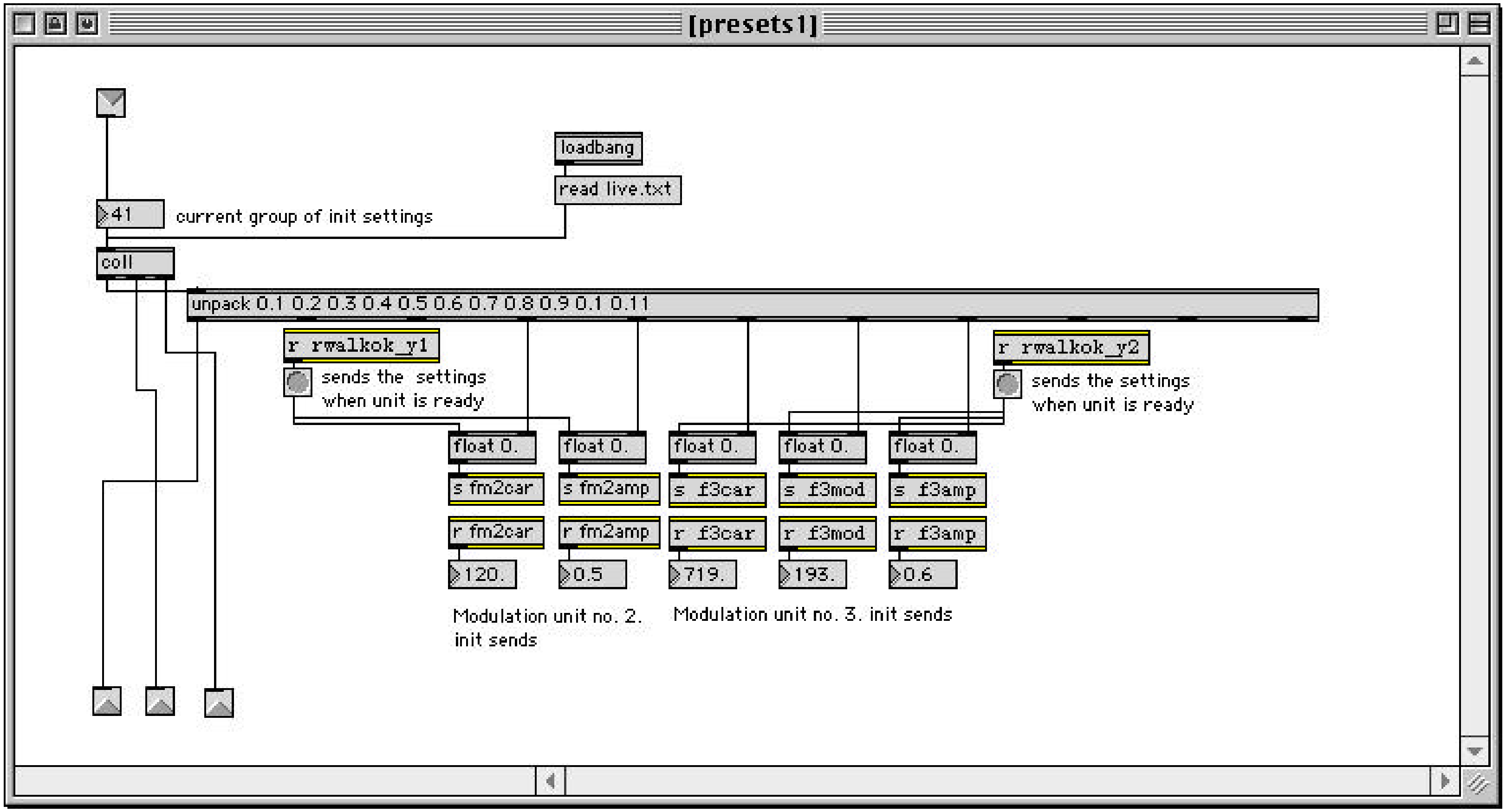


Fig. 14. The “presets1” routine used to change the initialization parameter settings for modulation units two and three. When the <<current group of init settings>> is changed, the test file line information is sent to the proper parameter inputs.

AIFF Files

Audio file playback is provided in two major forms; first, pre-composed quadraphonic soundtracks provide structural divisions and second, shorter monophonic files are used with direct control from the sensors. The quadraphonic soundtracks vary in duration and are composed from sonic material that is used to create the monophonic files. These blocks of composed material are designed to provide formal cues as well as to add structural stability to the work. Figure 15 shows the quadraphonic file playback module. The files are chosen and loaded using the lingo script. The files range in length from 30-seconds to longer 2-minute sections. The playback files are not only used for structural aspects of the work, but, they also have a more ambient role in the creation of the over all fabric in the sonic environment. The composed recordings are compiled into four-track AIFF files and are played at their original speed with no interference from the rug or panning and effect modules.

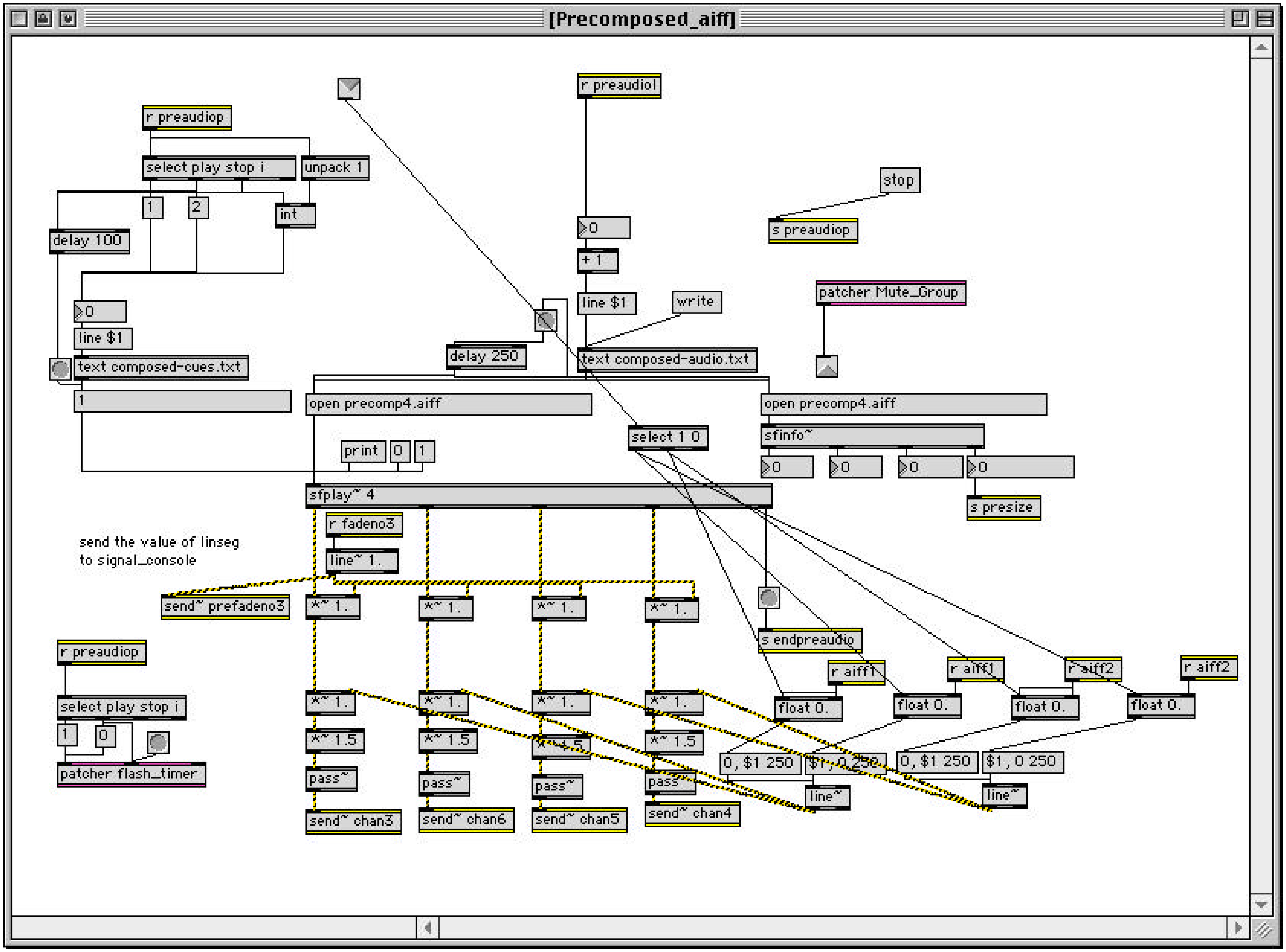


Fig. 15. The quadraphonic AIFF file playback routine.

Shorter monophonic AIFF files, however, are subject to signal processing and placement and are also directly initiated by the rug sensor triggers. Sixteen separate buffers are used to create a dynamic playback module providing sixteen discrete audio signals that are sent to the panning and effects routines. Line segment generators in each of the file buffers drive the speed and direction of playback. The duration of the line segment is a variable that is associated with timings from the rug. Each sensor on the rug triggers one of the file buffers directly, yet, the duration of the line segment is taken from the global x or y mapping group to which the sensor belongs. A filter is also applied to this sensor data. This filter sets a minimum duration, providing enough time for an efficient audio rate envelope to function, and also will not allow the repetition of a sensor until another sensor in the group has been triggered. Therefore, the rate at which one file is being read can be adjusted or changed during playback by another in the same group. The filter not only reduces unwanted noise from sounds that quickly stop and then return, but at the same time allow the dancer to reposition or adjust balance without sporadically re-triggering the sampled sounds.

A large number of monophonic samples are used in this composition. These samples are loaded into the file buffers and are grouped as sets of short, medium and long lengths. These sets are then associated with areas of the sensor device (see figs. 16 and 17). For example, at times the center of the carpet is set to trigger short samples while medium and longer samples can be utilized as the dancer moves to the outer regions of the carpet.

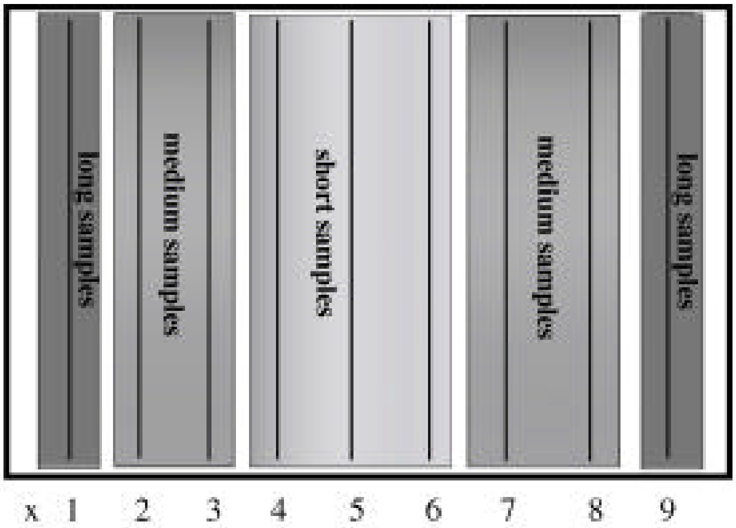


Fig. 16. The mapping of the x-sensors places control over longer sample lengths at the edges of the carpet.

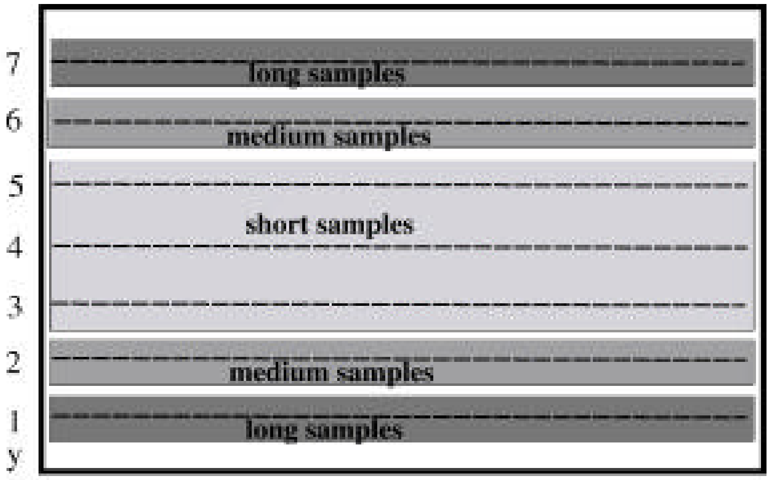


Fig. 17. The mapping of the y-sensors places control over longer sample lengths at the edges of the carpet.

In this way, the center of the carpet will be more “live” and the dancer can expect to have an immediate consequence to movement. The outer regions, in this arrangement, will react more slowly and are designed to have a longer and more evolving sonic character that can be associated with sweeping or pausing gestures. This format of mapping gives the dancer more control over the triggered sounds in relation to their pace and movement. The arrangement of sensor to sample length is controlled by loading the buffers with banks of AIFF files specified in the lingo script. The direction and shape of the line segments that drive the audio buffers are changed in this way and the line segments can be reset as a group or individually.

Placement of audio signals.

The panning routines place the audio signals into one of four stereo fields using values that are directly related to areas on the rug. The center of the carpet represents even panning in the various stereo fields. Movement on the carpet from one sensor to another, as represented in the generic mapping routine, results in linear connection of the sensor points. In the panning routines this linear data drives another non-linear table that calculates a panning crossover at the audio rate (see fig. 18). The routines place the monophonic signals between two loudspeakers considering the point of crossover to an attenuated signal of approximately 70%. This power crossover creates a dynamic and efficient panning unit with little loss of power during fades between loudspeakers. In other words this produces logarithmic amplitude fades to eliminate a perceived loss of amplitude in the middle.

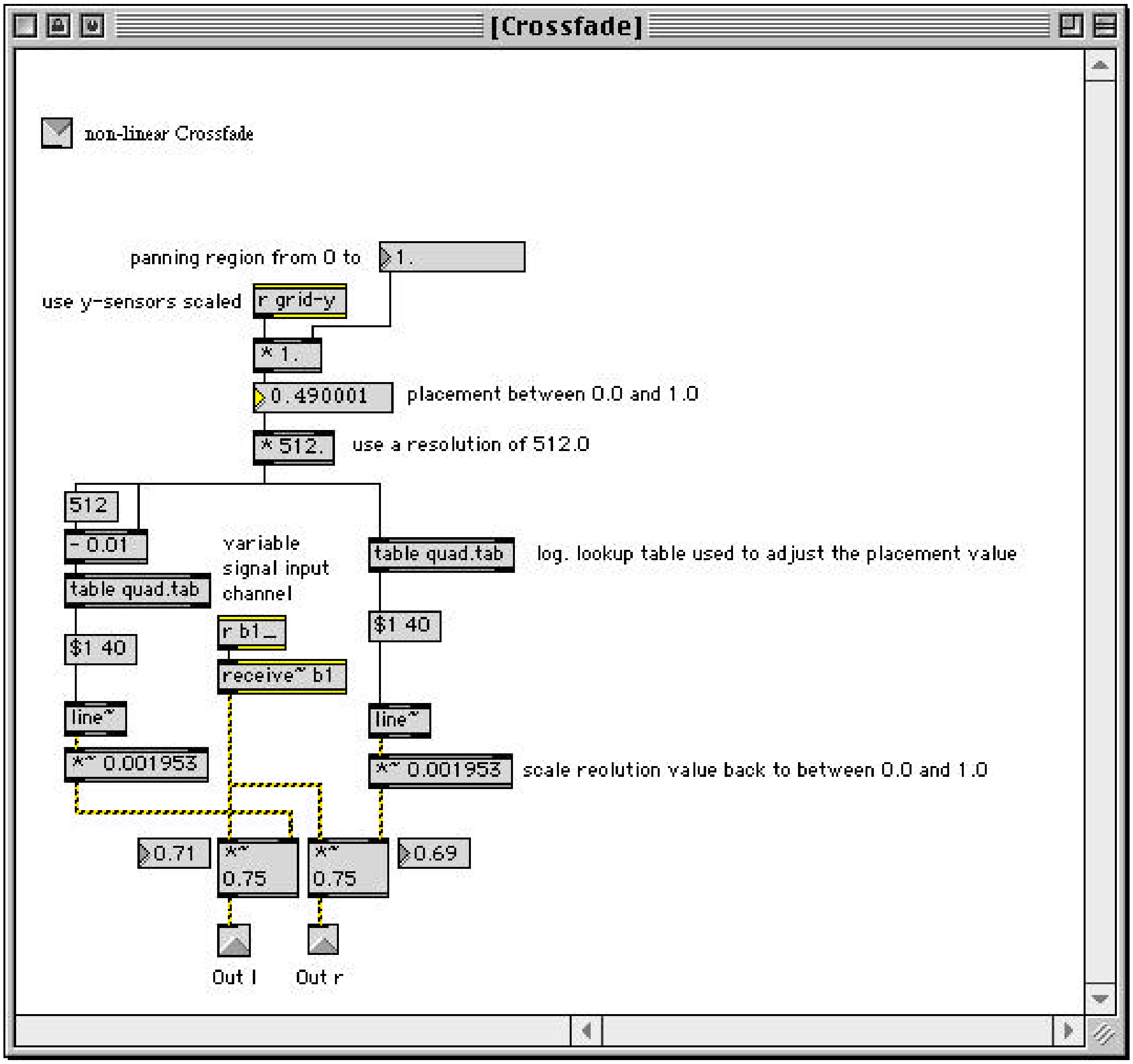


Fig. 18. One of the crossfade routines used to control the placement of the audio signal between two loudspeakers.

The duration at which the points on the rug are connected by the line segments is a variable related to the rate of sensor change from the carpet. When there is a lot of activity on the rug and the duration of sensor down states within a sensor group is short, the line segment duration is also short. When more time separates the movement on the carpet, the line segment has a longer duration. In this way the dancer can prepare the panning routines for a designed gesture that has an expected outcome. If the dancer leaps from one point to another, the panning routine will have a far different outcome depending on the amount of time between the last and previous action.

Signal Processing

A series of four similar effect racks, each made of several processing units, perform similar sonic production while maintaining differing mapping strategies (see fig. 19). The separate bank of filters, which can be adjusted and specified individually and or globally during performance by the lingo script, send their output to the panning routines for placement. The output from each rack is sent to one of the panning routines for placement in the sonic environment. Each of the effects is dynamically controlled by the rug sensor data. For instance, the normalized rug coordinates are used to control the frequency attenuation in the filter banks. However, the normalized data is mapped differently in each of the units. In this situation the frequency of each audio signal is subject to a filter sweep that is uniquely adjusted with the sensor data from each step. The duration of delay, amplitude settings, filter coefficients and feedback levels are also dynamically changed in real time in relation to movement on the carpet.

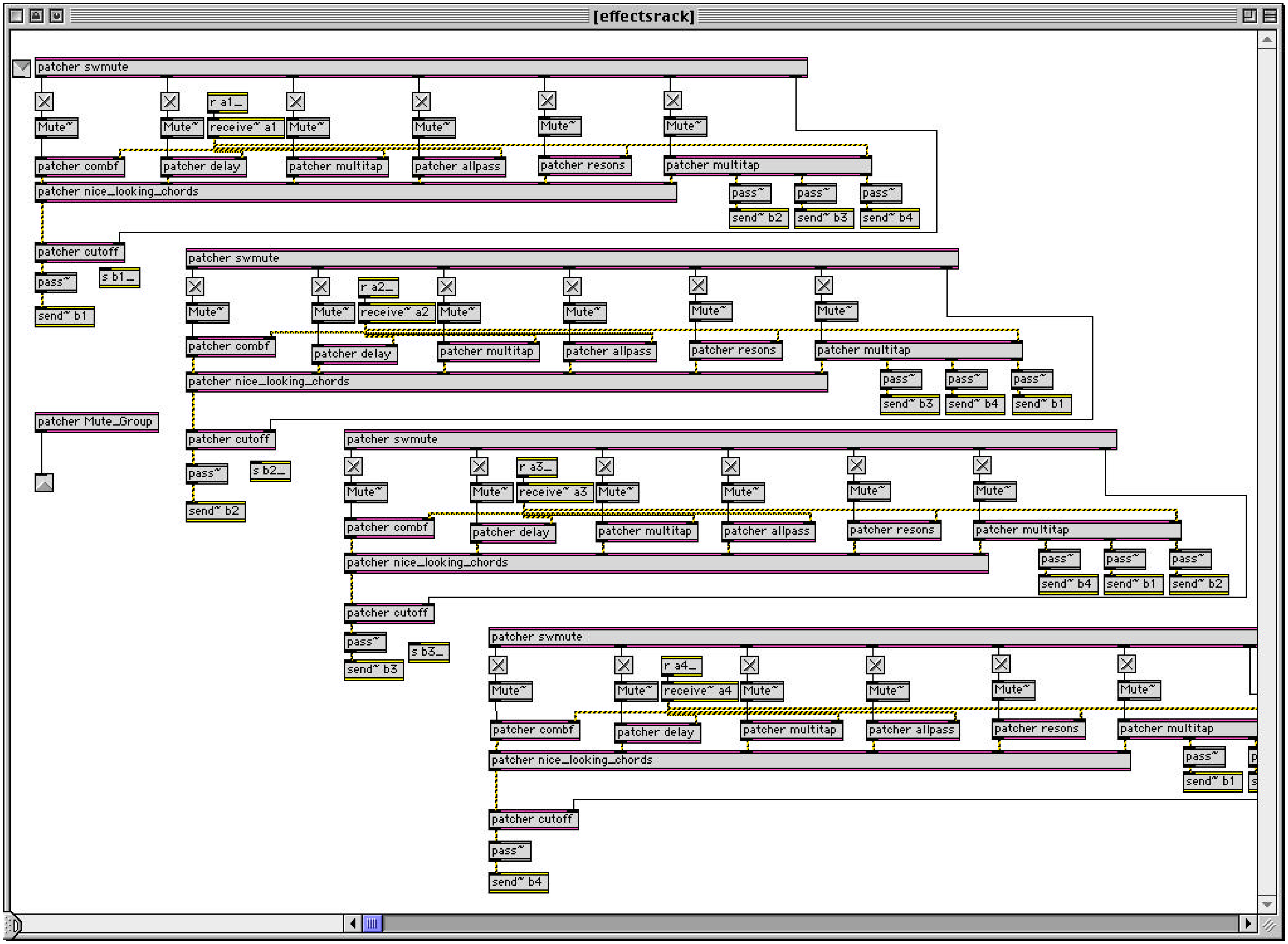


Fig. 19. the effects window housing the racks of available effects.

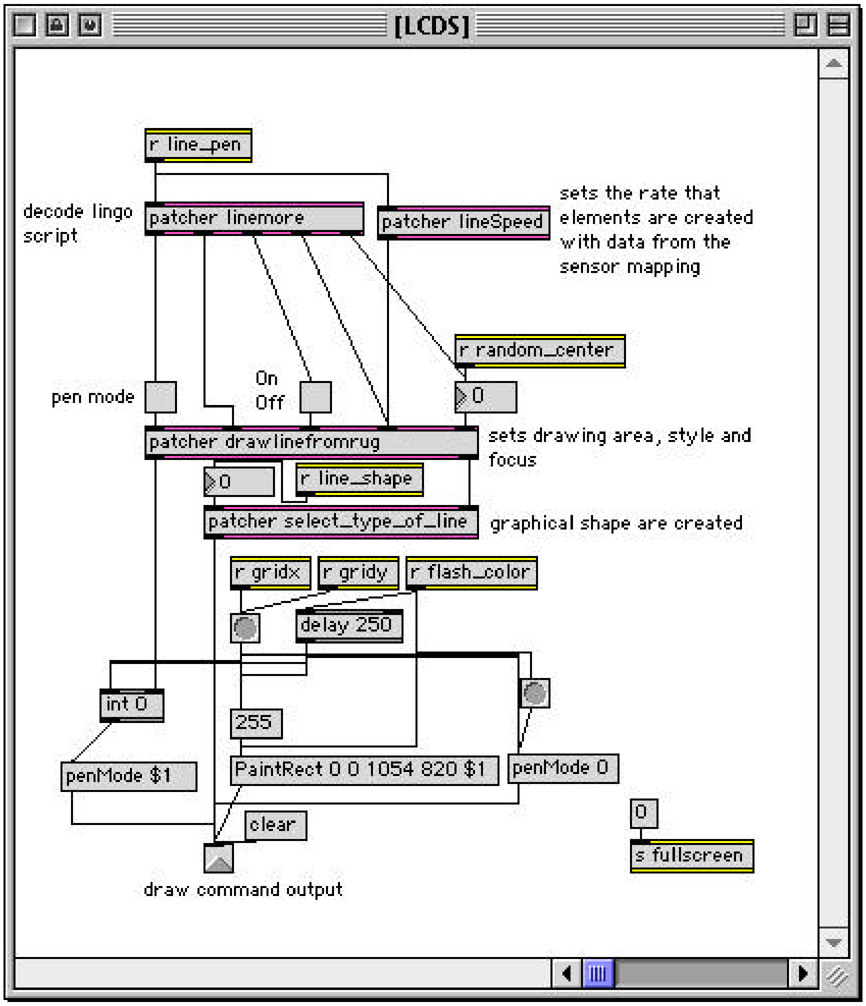
The effect routines enhance the aural sensation of the placement in the sonic environment. For instance, the frequencies of a signal can be adjusted to create a sense of height in the stereo fields and employed as the dancer moves away from the audience. Using another effect routine it is possible to vary the delay time and feedback levels of the signal in relation to sensor activity. Each of the audio modules has a certain level of autonomy. That is, many functions are designed to continue running when the unit is active. The lingo script serves to communicate commands to the separate units and these commands are stored in an editable text file. Using combinations of these audio units, the sonic material is created as each unit contributes to the color, placement, shape and texture of musical elements in the sonic environment.

The Visuals

Two main modules that are under the supervision of the score produce the visual materials. Graphic and movie elements are created by routines that utilize mathematical logarithms to draw shapes and elements on the correct area of the screen. The visual modules utilize the sensor mapping in similar ways to the audio modules. The quadrants of the screen are mapped to the regions on the carpet using the global sensor data. The sensors are also mapped to control the rate, shape, lengths and color of the generated material. The mapping designs that control both the frame rate at which the digital movies are read and the screen quadrants that are used for their display are similar to those used to control the long and short audio samples. Filters are installed with these mapping routines in order to control the rate at which the graphic elements are re-drawn to the screen, providing an opportunity for the dancer to adjust footing.

The visual elements are drawn to a specially designed window that uses MAX display objects placed in layers. These layers enable the movies and graphic elements to be drawn simultaneously. The foreground and background status of each of the layers are changed when the objects receive input from the visual modules. The display resolution of the computer should be set to 1024 * 768. The display window is framed using a black background to surround the display objects and darken the outer regions of the window. In this way, the display window can be used at higher screen resolutions if the correct setting is not available for the projection units. Also, all of the normal control bars that are present with computer display are removed by sending the window a "full screen" message. Then, to darken the projection field a large black rectangle is drawn on the top layer of the projection window covering all other display objects.

At other times, using the LCD module seen in figure 20, colors are displayed to fill the projection field and illuminate the dancer. The LCD module is synchronized to the quadraphonic file playback causing the projection field to be filled with a flash of color that is programmed to coincide with the progression of the recorded files. The synchronization between these audiovisual materials helps to provide structural cues to the performer. The technique of redrawing the top layer of the projection window is also used to erase the visual material from the display window.   As the dancer moves from one projection quadrant to another, the display screen is re-drawn, in a similar fashion to a blackboard being erased. This removes any trace visual material from the old projection field and provides a clean slate for new graphic material that is focused on a new portion of the display window.



decode lingo script

sets the rate that elements are created with data from the sensor mapping

sets drawing area, style and focus

graphical shape are created

draw command output

Fig. 20. The LCD graphic routine.

The audiovisual modules are designed to have autonomous features and to run simultaneously with one another. The modules depend on the lingo script for their direction. For instance, the logarithms in the LCD module that are used to generate graphic material will continue to draw line segments even when other layers of the projection window are currently displaying other graphics. In the case of movie elements, the line segments that are being generated by the LCD module are drawn over the current frame of the movie and then replaced by the next frame. The amount of overlapping graphic material is dependent on the speed that each of the layers is being drawn.

The technique of overlapping elements is used for visual transition between the various graphic materials. Transition between movie fragments is achieved by gradually changing the color of the elements being produced by the LCD module. These graphic elements interrupt the frames of the movie ultimately replacing them. The next movie segment is prepared by changing the color of the LCD's material and then introducing the new frames. Like wise, the generation of synthesized audio will run through its cycle of re-initialization settings until the unit is successfully turned off. The transitions and cross fading of sonic structures are managed by the attenuation of the audio unit's output and this is achieved with commands sent via lingo script. The composition is formed through time by controlling these dynamic modules by controlling the scheduled delivery of the lingo script. The sonic and visual material's timbre, color, shape, tempo and texture are combined and associated with each other to create an interactive environment that progresses through time with a formal goal.

CHAPTER 6

CONCLUSION

This work represents a new approach to programming and compositional control in my series of *jn.gesture* rug compositions for dance. The software has been designed to allow the artistic concerns to have precedence over the technical elements associated with software design in this style of composition. The modules, which function simultaneously with one another, are controlled by the lingo script, which allows immediate re-configuration and adjustment of their output. The delivery of the lingo is directed by the GUI score, which executes the script at a prescribed rate of progress. Each of the multimedia modules is designed to be flexible and comprehensive, allowing the script exacting control over the parameters used to configure its role in the composition. The lingo script is the score and for performance it has been automated as an interface directing the user to ensure that the lingo script is executed at the correct points in time. The software has proven to be flexible and adjustable allowing for a practical rehearsal situation as well as interface that is adaptable to the dancer's physical dimensions and expressive style.

The software and lingo scripts have been developed with a concept that will provide a flexible engine for the creation of future compositions in this venue. By creating new media files and initialization settings the program will generate new audiovisual material. Since the lingo script synchronizes and controls the production of media elements, I can create new GUI scores and utilize the software for unique compositions that are not restricted by previous configurations and mapping scenarios. This will allow me to focus on more artistic concerns and utilize the scripting language to combine and bind the media elements in formal organization. Also, new lingo script, modules and routines can be added to the program with out compromising the existing modules and lingo. In this way, the series of compositions can incorporate new technology, as it becomes available, by complementing and replacing inadequate modules as deemed necessary.

The rug's 16 stationary sensors form a grid that has proven to be an efficient form of communication between the device and the computer. Complex mapping scenarios can be achieved with this minimal amount of sensor data. The data is stable, predictable, fast and accurate and the device is not effected by the lighting and visual aspects of the work. In the future, I would like to design some new tracking devices to be incorporated with this system. The use of an optical tracking device, however, is not desirable. This is because the graphic elements are being projected directly on the subject who is being tracked. The change of ambient and direct light during performance causes video capture to become unreliable. Another concern is not restricting the movement of the dancer; so, an umbilical connection between the dancer and the computer is not practical. The device must be wireless and attach comfortably to the dancer.

A device that is able to measure the altitude of a body's extremity and send a continual stream of this measurement could provide useful z data that would supply a third dimension in the tracking paradigm. The z data can be associated with sweeping arm movements and directly control glissandi and other audiovisual elements. The combination of x, y, and z sensor data can be interpreted by the routines to have simultaneous mapping functions, by extending the generic mapping routines that are currently implemented in the software. For instance, sensor data can be used to create an azimuth-zenith coordinate that then controls the depth and height of projection, and at the same time it can be used as individual or group data to provide the immediate and direct control over the audiovisual parameters.

In future compositions it may prove desirable to use more than one computer for the generation of the media elements. If separate computers are used the role, quality and interaction of the media elements may be enhanced. The use of multiple computers may not be necessary as the power of the personal computer increases, but at this point the scenario does offer some interesting possibilities. If the visual elements are produced on a separate computer and built in a language that is specifically designed for the manipulation of graphic elements, more artistic control would be possible. The computer running MAX/MSP©[19] could then utilize more of its CPU time for the generation of real-time audio, as the role of visual generation in MAX©[20] would then lessen to the point of support.

Adding modules designed to communicate the lingo script to the other computers will synchronize creation of the media elements. The score will control lingo-time relationship and the production of material on the separate computers without having to synchronize of their internal clocks. A concern in this situation is the distribution of the sensor data to the separate computers. If one computer is used to interpret and generically map the sensor data, Ethernet or MIDI protocols can be used to communicate this information to the other computers. This represents a potentially large amount of data using a MIDI interface and may overload the communication between the separate computers. This is especially true if MIDI is also being used for the transmission of lingo related messages. One solution is to have the USB communication from the sensors sent to each computer being utilized for the composition. In this case, each computer would be in charge of mapping the sensor data that is received at a similar time through its serial connection.

[19] MSP © 1997 David Zicarelli—All rights reserved based on Pd by Miller Puckette © 1997 The Regents of the University of California MSP and Pd are based on ideas in Max/FTS, an advanced DSP platform © IRCAM.

[20] MAX copyright © 1900-1997 Opcode systems/IRCAM.

In all of these cases the lingo script will be used to synchronize the media modules running on any number of computer simultaneously. The lingo script provides a flexible outline of the computational processes that can be edited as a text file. This is an important feature of the software because the composition can be adjusted without re-programming the software itself. A compositional concept can be created with the input from a collaborative performer through the use of lingo. Each of the elements in the composition can then be adjusted quickly with the performer present by adding new commands to the script file. This allows the artistic concerns of the performer to be introduce into the compositional process using the existing software. The amount of collaboration will vary depending on the composition, but, at the same time, by editing the lingo script one can refine the reactionary outcome of the software to the physical activity on the rug sensor.

APPENDIX

The lingo script used for the performance of jn4.gesture.

```
------------------------ 0;
------------------------ 1 dac on;
1 dac on;
------------------------ 2 ready init;
2 fadeno1 0;
fadeno2 0;
fadeno3 0;
fadeno4 0;
fullscreen 1;
preaudiol 1;
effectno on, effall 1;
line_shape 5;
dot_size 5;
set_dot set 1;
set_dot forback;
set_dot rand_off;
mute_group precomp_on, bufsnd_on;
colorname red;
fadeno3 1.;
trig_buf 0;
init_linseg all 1 0 1 1. 0.2 0.5;
flash_color 255;
------------------------ 3 Introduction;
3 fadeno1 0, 1 100;
fadeno3 0, 1 100;
preaudiop play;
------------------------ 4 after flashes start transition circles;
4 trig_buf 1;
dot_size 5, 67 2000;
set_dot set 5, forback, rand_off;
line_pen dur_rug 1, rcenter 1, lm_on 1;
------------------------ 5 tran part two;
5 line_pen dur_exp 1;
colorname gray;
dot_size 68, 1 9000;
------------------------ 6 end transition;
6 colorname black;
------------------------ 7 stop intro;
7 trig_buf 0;
random_center 0;
------------------------ 8 general pause (end Intro);
8 preaudiop stop;
```

```
preaudiol 2;
colorname red;
movieload graything.mov;
init_linseg all 2 0 1 1. 0.2 0.5;
set_dot rand_off, forback, set 5;
dot_size 5;
effectno effall 12;
flash_color 255;
loadbuffers 0;
------------------------ 9 start first section;
9 preaudiop play;
------------------------ 10 after flash start audio and transition circles;
10 trig_buf 1;
fadeno4 0, 1 2000;
dot_size 1, 67 4000 1 3500;
line_pen dur_std 1, rcenter 1, lm_on 1;
------------------------ 11 start transition;
11 line_pen dur_exp 1, pen_on;
set_dot set 25, forback, rand_on;
colorname orange;
dot_size 1;
effectno effall 12;
loadbuffers 20;
init_linseg all 2 0 1 1. 0.2 0.5;
------------------------ 12 transition part two;
12 colorname gray;
line_pen pen_off 0;
set_dot set 25, forback, rand_on;
dot_size 5, 55 1000 1 4000;
------------------------ 13 transition part three;
13 random_center 0;
set_dot rand_off, forback, set 5;
movierun on;
effectno effa1 2, effa2, 3, effa3 4, effa4 5;
------------------------ 14 transition part four;
14 random_center 0;
effectno effa1 3, effa2, 1, effa3 3, effa4 4;
------------------------ 15 transition part five;
15 movierun off;
set_dot set 5, forback, rand_off;
line_pen dur_exp 1;
dot_size 1, 67 4000;
line_pen dur_exp 1, rcenter 1, lm_on;
------------------------ 16 transition part six;
16 movieload monolith.mov;
dot_size 70, 5 2000;
```

```
effectno effall 23;
flash_color 255;
------------------------ 17 transition part seven;
17 colorname red;
dot_size 5, 25 1000 10 100 50 1500 1 5000;
------------------------ 18 Start improvisation section;
18 random_center 0;
line_pen lm_off;
movierun on;
preaudiop stop;
loadbuffers 40;
effectno effall 56;
------------------------ 19 start transition;
19 colorname red;
set_dot rand_on, forback, set 10;
dot_size 5;
line_shape 5;
line_pen dur_exp 1, rcenter 1, lm_on 1;
dot_size 1;
------------------------ 20 transition part one;
20 movierun off;
movieload graything.mov;
set_dot rand_off, forback, set 30;
dot_size 30, 1 2000;
flash_color 255;
line_shape 2;
------------------------ 21 end transition;
21 colorname gray;
random_center 1;
dot_size 1, 90 2000;
line_pen ldur_std 60;
preaudiol 3;
------------------------ 22 start ending part of section one;
22 line_pen ldur_std 20;
dot_size 90, 1 3000;
effectno effall 13;
preaudiop play;
loadbuffers 40;
line_shape 5;
------------------------ 23;
23 random_center 0;
movierun on;
line_pen lm_off;
------------------------ 24;
24 loadbuffers 20;
effectno effall 12;
```

```
------------------------ 25;
25 colorname gray;
dot_size 5;
movierun off;
line_pen ldur_exp, lm_on, rcenter 1;
line_shape 5;
------------------------ 26;
26 colorname rand;
loadbuffers 0;
effectno effall 15;
------------------------ 27;
27 colorname gray;
dot_size 15, 1 2000;
fadeno2 1., 0 2000;
------------------------ 28 end first section pause;
28 random_center 0;
fadeno3 1, 0 50;
fadeno4 1, 0 50;
trig_buf 0;
set_dot rand_off;
mute_group livefm_on;
effectno effa1 15, effa2 25, effa3 35, effa4 45;
line_pen ldur_exp, lm_on, ldur_exp;
line_shape 3;
colorname purple;
movieload greenthing.mov;
flash_color 255;
------------------------ 29 Start the second section fm;
29 mute_group bufsnd_off, precomp_off;
fadeno2 0., 1. 1000;
livefm/aiff 0 1;
fm_rwalk 1;
gliss 1;
random_center 1;
------------------------ 30 start transition shape changes;
30 line_shape 4;
------------------------ 31 transition part two centered lines walk = 11;
31 line_shape 1;
random_center 2;
fm_rwalk 11;
gliss 2;
effectno effa1 25, effa2 35, effa3 45, effa4 16;
------------------------ 32 transition part three change to green lines;
32 gliss 1;
colorname green;
------------------------ 33 start movie;
```

```
33 random_center 0;
line_pen lm_off;
movierun on;
fm_rwalk 21;
gliss 1;
effectno effa1 16, effa2 2, effa3 3, effa4 34;
------------------------ 34 end lines;
34 random_center 0;
------------------------ 35 start transition number two;
35 movierun off;
line_pen ldur_exp, lm_on;
line_shape 4;
random_center 1;
gliss 1;
------------------------ 36;
36 line_shape 3;
colorname yellow;
movieload yellowthing.mov;
flash_color 255;
------------------------ 37;
37 line_pen ldur_exp, lm_off;
random_center 0;
movierun on;
fm_rwalk 31;
gliss 2;
effectno effa1 25, effa2 56, effa3 56, effa4 25;
------------------------ 38;
38 random_center 0;
------------------------ 39;
39 fm_rwalk 11;
effectno effa1 56, effa2 25, effa3 35, effa4 56;
gliss 1;
colorname yellow;
line_shape 1;
line_pen ldur_exp, lm_on;
movierun off;
random_center 1;
------------------------ 40;
40 effectno effa1 16, effa2 5, effa3 4, effa4 1;
------------------------ 41;
41 colorname gray;
fm_rwalk 1;
gliss 1;
fadeno2 1., 0. 12000;
------------------------ 42;
42 colorname black;
```

```
preaudiol 4;
------------------------ 43 end fm part of section 2;
43 line_pen ldur_exp, lm_on, rcenter 1;
mute_group precomp_on, bufsnd_on;
movieload bluething.mov;
flash_color 255;
loadbuffers 80;
------------------------ 44 start the second part second section;
44 fadeno3 0, 1 200;
preaudiop play;
effectno effa1 15, effa2 56, effa3 24, effa4 13;
------------------------ 45;
45 colorname blue;
line_shape 6;
set_dot set 200, rand_on, forback;
line_pen ldur_exp, lm_on, rcenter 2;
init_linseg all 1 0 1 1. 0.2 0.5;
fadeno4 0, 1 2000;
trig_buf 1;
fadeno2 0;
mute_group livefm-off;
------------------------ 46;
46 effectno effa1 23, effa2 14, effa3 13, effa4 56;
------------------------ 47;
47 random_center 0;
set_dot rand_off;
line_pen lm_off;
movierun on;
------------------------ 48;
48 colorname gray;
line_pen ldur_exp, lm_on, pen_on;
line_shape 6;
set_dot forback, set 15;
dot_size 1;
movierun off;
random_center 1;
------------------------ 49;
49 movieload marble.mov;
colorname orange;
line_pen pen_off 1;
effectno effa1 16, effa2 45, effa3 14, effa4 16;
flash_color 255;
------------------------ 50;
50 random_center 0;
set_dot rand_off;
line_pen lm_off;
```

```
movierun on;
effectno effa1 45, effa2 16, effa3 16, effa4 14;
colorname black;
line_shape 5;
set_dot set 300;
------------------------ 51;
51 random_center 0;
------------------------ 52;
52 movierun off;
set_dot rand_on;
line_pen ldur_exp, lm_on;
random_center 2;
effectno effa1 25, effa2 35, effa3 12, effa4 24;
------------------------ 53;
53 movieload graything.mov;
flash_color 255;
random_center 1;
set_dot set 5, rand_off;
dot_size 200, 5 2000;
colorname gray;
effectno effall 3;
------------------------ 54;
54 line_pen pen_on;
loadbuffers 20;
movierun on;
dot_size 1, 100 4000;
set_dot rand_on, forback, set 10;
effectno effall 6;
colorname rand;
------------------------ 55;
55 effectno effall 3;
dot_size 80, 20 1500 loadbuffers 0;
mute_group livefm_on;
fadeno2 0, 0.7 1000 0 2000;
colorname purple;
------------------------ 56;
56 movierun off;
effectno effall 2;
line_pen pen_off 0;
------------------------ 57;
57 trig_buf 0;
colorname gray;
dot_size 20, 1 4000;
fadeno4 1, 0 4000;
------------------------ 58;
58 fadeno1 0, 1 100;
```

```
random_center 0;
mute_group bufsnd_off, precomp_off, erack_off, livefm_off;
------------------------ 59;
59;
------------------------ 60;
60 random_center 0;
```

REFERENCE LIST

Books and Articles

Bongers, Bert. "An interview with Sensorband." Computer Music Journal Vol. 22, No. 1 (Spring 1998): Pages 13-24.

Brewster, Stephen A. "Using Nonspeech Sounds to Provide Navigation Cues." ACM Transactions on Computer-Human Interaction, Vol. 5, No. 3 (Sept. 1998): Pages 224-259.

Camurri, Antonio and Riccardo Trocca. "Movement and Gesture in Intelligent Interactive Music Systems." Trends in Gestural Control of Music. M. Wanderley and M. Battier, Eds. Ircam - Centre Pompidou, 2000, Pages 95-109.

Cascone, Kim. "The Aesthetics of Failure: "Post-Digital" Tendencies in Contemporary Computer Music." Computer Music Journal Vol. 24, No. 4: Pages 12-18.

Choi, Insook. "Gestural Primitives and the context for computational processing in an interactive performance system." Trends in Gestural Control of Music. M. Wanderley and M. Battier, Eds. Ircam - Centre Pompidou, 2000, Pages 139-172.

Cope, David. New Directions in Music. Dubuque, IA.: Brown and Benchmark, 1993.

Dix, Alan. Et all. "Exploiting Space and Location as a Design Framework for Interactive Mobile Systems." ACM Transactions on Computer-Human Interaction, Vol. 7, No. 3 (Sept. 2000): Pages 285-321.

Dodge, Charles and Thomas A. Jerse. Computer Music. New York: Schirmer Books, 1985.

Fedorkow, Guy, William Buxton, and K.C. Smith. "A Computer-Controlled Sound Distribution System for the Performance of Electroacoustic Music." Computer Music Journal Vol. 2 No. 3: 33-42.

Garnett, Guy E., and Carmille. Goudeseune. "Performance Factors in Control of High-Dimensional Spaces." In Proceedings of the 1999 International Computer Music Conference, Beijing China. International Computer Music Association, pages. 268 - 271.

Goldberg, RoseLee. Laurie Anderson. New York : Harry N. Abrams, c2000.

Goldberg, RoseLee. Performance Art. New York: Harry N. Abrams, Inc., 1988.

Hubona, Geoffrey s., Philip N. Wheeler, Gregory W. Shirah and Matthew Brandt. "The Relative Contributions of Stereo Lighting, and Background Scenes in Promoting 3D Depth Visualization." ACM Transactions on Computer-Human Interaction, Vol. 6, No. 3 (Sept. 1999): Pages 214-242.

Kessler, Drew G., Larry F. Hodges, and Neff Walker. “Evaluation of the CyberGlove as a Whole-Hand Input Device.” ACM Transactions on Computer-Human Interaction, Vol. 2, No. 4 (Dec. 1995): Pages 263-283.

Mulder, Axel. Human movement tracking technology. Hand centered studies of human movement Project, Technical report 94-1., Copyright 1994 Simon Fraser University. ftp://fas.sfu.ca/pub/cs/graphics/vmi/HMTT.pub.ps.gz (11 Jan 2003).

Pressing, J. 1990. "Cybernetic Issues in Interactive Performance Systems." Computer Music Journal Vol. 14, No. 1: Pages 12-25.

Roads, Curtis. The Computer Music Tutorial. Massachusetts: Massachusetts Institute of Technology, 1996.

Rowe, Robert. Machine Musicianship. Cambridge, Mass.; London : MIT Press, c2001.

Ryan, J. "Effort and Expression." In Proceedings of the 1992 International Computer Music Conference. San Francisco, International Computer Music Association, Pages 414-416.

Siegel, Wayne and Jens Jacobsen. “Composing for the Digital Dance Interface.” In Proceedings of the 1999 International Computer Music Conference, Beijing China. International Computer Music Association, Pages 276-7.

Stockhausen, Karlheinz. "Music in space." die Reihe, Report Analyses. n.s. 5 (1959). Translated by Ruth Koening. New Jersey: Universal Edition Publishing, Inc., 1975: 67-82.

Tanaka, A. "Musical Performance Practice on Sensor-based Instruments." Trends in Gestural Control of Music. M. Wanderley and M. Battier, Eds. Ircam - Centre Pompidou, 2000.

Wanderley, M. and M. Battier, eds. Trends in Gestural Control of Music. Ircam - Centre Pompidou, 2000.

Ware, Colin and Kathy Lowther. “Selection Using a One-Eyed Cursor in a Fish Tank VR Environment.” ACM Transactions on Computer-Human Interaction, Vol. 4, No. 4 (Dec. 1997): Pages 309-322.

Wishart, Trevor edited by Simon Emmerson. On Sonic Art. Netherlands: Hardwood Academic Publishers GMBH., 1996.

Computer Programs

Csound 3.50, Massachusetts Institute of Technology.

Extreme 3D 2, Macromedia inc. 600 Townsend St., San Francisco, CA 94103.

Max 3.5.9, Opcode Systems, inc., Palo Alto, Ca.

Pro Tools, Digidesign inc., 3401-A Hillview Avenue, Palo Alto, CA. 94304.

Strata Videoshop, 2 West St. George Blvd, St. George, UT. 84770.

Compositions and Scores

Anderson, Laurie. Puppet Motel [interactive multimedia] / Laurie Anderson; with Hsin-chien Huang. New York: Voyager, c1995.

Lanza, Alcides. Penetrations V (1970-IV) for a minimum of 10 sound sources, with voices, lights, electronic sounds and electronic extensions. New York: Boosey & Hawkes, 1972

Stockhausen, Karlheinz. Spiral. Wien: Universal Edition, 1973.