TOWARDS FACILITATING GRAPHICAL INTERACTION: SOME EXAMPLES FROM COMPUTER-AIDED MUSICAL COMPOSITION

Ronald Baecker
William Buxton
William Reeves

Dynamic Graphics Project & Structured Sound Synthesis Project
Computer Systems Research Group
University of Toronto

Abstract

This paper sketches some of the issues involved in designing user interfaces to interactive graphics systems. The ideas are presented and motivated through a series of examples that arise in an interactive graphical editor for musical scores expressed in common music notation.

LA FACILITATION DES ACTIONS AVEC DES SYSTEMES GRAPHIQUES: QUELQUES EXEMPLES QUE VIENNENT DE LA COMPOSITION MUSICALE A L'AIDE DE L'ORDINATEUR

Résumé

Cette communication traite de quelqu'unes des solutions que se présentent à l'égard de la construction des interfaces pour l'usager d'un système graphique interactif. Les idées se sont présentées et inspirées au moyen des questions que se sont posées en ce qui concerne un éditeur graphique interactif pour les partitions exprimées à la notation commune de musique.

TOWARDS FACILITATING GRAPHICAL INTERACTION: SOME EXAMPLES FROM COMPUTER-AIDED MUSICAL COMPOSITION

1. *Introduction*

The development of rich, highly-responsive user-oriented interactive graphics systems has been hindered by the lack of adequate conceptual frameworks for describing graphical input and interaction, and for characterizing and evaluating user interfaces. Primitive functions and descriptive formalisms for graphical output have been well understood and commonly accepted for a number of years. Corresponding proposals have been made on the input side ([Foley 74], [Wallace 76], [Kasik 76], [Deecker 77], [Core 77], [Van den Bos 78], [Green 79], and [Baecker 79], among others), but both understanding and agreement are less widespread. This is due in part to the fact that our languages for describing input are very hardware-dependent, and a great diversity of devices are in common use.

Our understanding of *graphical interaction* is even less advanced than that of input. Graphical interaction is a set of actions of a computer system and its user on each other. The user provides input to the system via a set of devices; the system provides output to the user via a set of displays. These inputs and outputs must be reciprocal; that is, they must be related to one another. Graphical interaction is then a succession of interrelated actions and reactions. If an interactive graphics system is to be responsive, these actions and reactions must be tightly coupled; in fact, they should usually be perceived as simultaneous.

We currently have great difficulty in describing graphical interaction due to the lack of an appropriate language. We have even more difficulty in evaluating the effectiveness of graphical interfaces, and in explaining with any rigor why one is better than another. And we have the greatest difficulty in designing effective interfaces, an activity much better characterized as an art or a craft than as a science or an engineering discipline.

This paper sketches some of the issues involved in designing user interfaces to interactive graphics systems. Most of the ideas will be presented and motivated through a series of examples. The examples will come from one domain — the application of interactive graphics in facilitating the use of an advanced real-time digital sound synthesizer for musical composition. In so doing, we shall also summarize the major goals and accomplishments to date of the Structured Sound Synthesis Project, which has designed and constructed the synthesizer and the graphic interfaces to it.

2. *Background*

The Structured Sound Synthesis Project (SSSP) is an interdisciplinary project whose aim is to conduct research into problems and benefits arising from the use of computers in musical composition [Buxton 78a,b,c]. This research can be considered in terms of two main areas: the investigation of new representations of musical data and processes, and the study of man-machine communication as it relates to music.

Integral to the research is the evolution of a computerized environment which one could term a "composer's assistant." Therefore, the development of an interactive music system — in combination with observing its use by trained composers — constitutes one of the main aspects of our research. In order to achieve a high degree of interaction required in such a system, one of our key activities has been the development of a computer-controlled digital synthesizer [Buxton 78d]. Another main area of activity has been the development of high-level "front-end" programs which would give the musically sophisticated (but technologically naive) user a high degree of access to the potential offered by the computer-synthesizer combination.

The basis for our approach to this second problem has been the consideration of behavioral issues arising in the course of composition. As a result, we view the composer's activities in terms of a task taxonomy consisting of four basic tasks:

1. Definition of the palette of timbres to be available; what we call *object definition*, which is analogous to choosing the instruments which are to comprise the composer's orchestra.

2. Definition of the pitch-time structure of a composition, a process which we can call *score definition*. In conventional music, this task would be roughly analogous to composing a piano version of a score.

3. The *orchestration* of the "score". Generally stated, attaching attributes (such as *objects* defined in Step 1) to *scores* defined in Step 2.

4. The *performance* of the material developed thus far, whether an entire (orchestrated or unorchestrated) score, or simply a single note (to audition a particular object, for example).

From the above taxonomy of tasks derives one of our first major decisions: to have two major data types, *objects* and *scores*, which relate to the sonic level and deeper structural level, respectively. Secondly — since composers work in different ways — we recognize that there should be no order imposed on the sequence in which the user undertakes the above four tasks. A composer should, for example, be permitted to perform a score before it is orchestrated. The implication of this is that the system should be capable of coping with incompletely specified data (through system defaults, for example). Finally, the composer must be provided a "handle" onto his data which goes beyond the note-by-note approach prevalent in most systems today; one which enables him to address data in "chunks" corresponding to the units within which he conceptualizes them.

Several modes of interaction have previously been used in music systems, such as alphanumeric text as in MUS10 [Smith 78], voice recognition [Tucker 77], and piano-type keyboards [New England Digital 78]. In our work we have adopted a bias towards graphics-based interaction [Newman 79], in the belief that this approach can make a significant contribution towards an effective human interface. First, music lends itself well to representations in the visual domain. Second, the task of editing music is complex in the sense that there are many parameters and commands to be manipulated and controlled; complexity which can be reduced by the graphic representation of information. Third, previous work [Pulfer 70], [Tanner 72], and [Vercoe 75] indicates that more congenial interfaces can be constructed using dynamic graphics techniques.

3. *Some Problems in Graphical Interaction and Their Solutions*

In this section, we shall present some aspects of the development of an interactive graphical editor for scores in traditional music notation, *ludwig*.

3.1. *Vocabulary and Display Organization*

The graphical representation of musical events used in *ludwig* is common music notation (CMN — see Figure 1). Our use of CMN does not imply an affinity towards it. Rather, CMN was used for two reasons. The new computer-related concepts are presented to the computer-naive composers within an environment familiar to all musicians. Techniques learned by the composer in using CMN are then applicable to other notational schemes. Second, by using an established representation it was possible to begin our research on interaction immediately. Many of the interactive methodologies evolved for CMN have been applied to other representations such as piano roll or time-line notation.
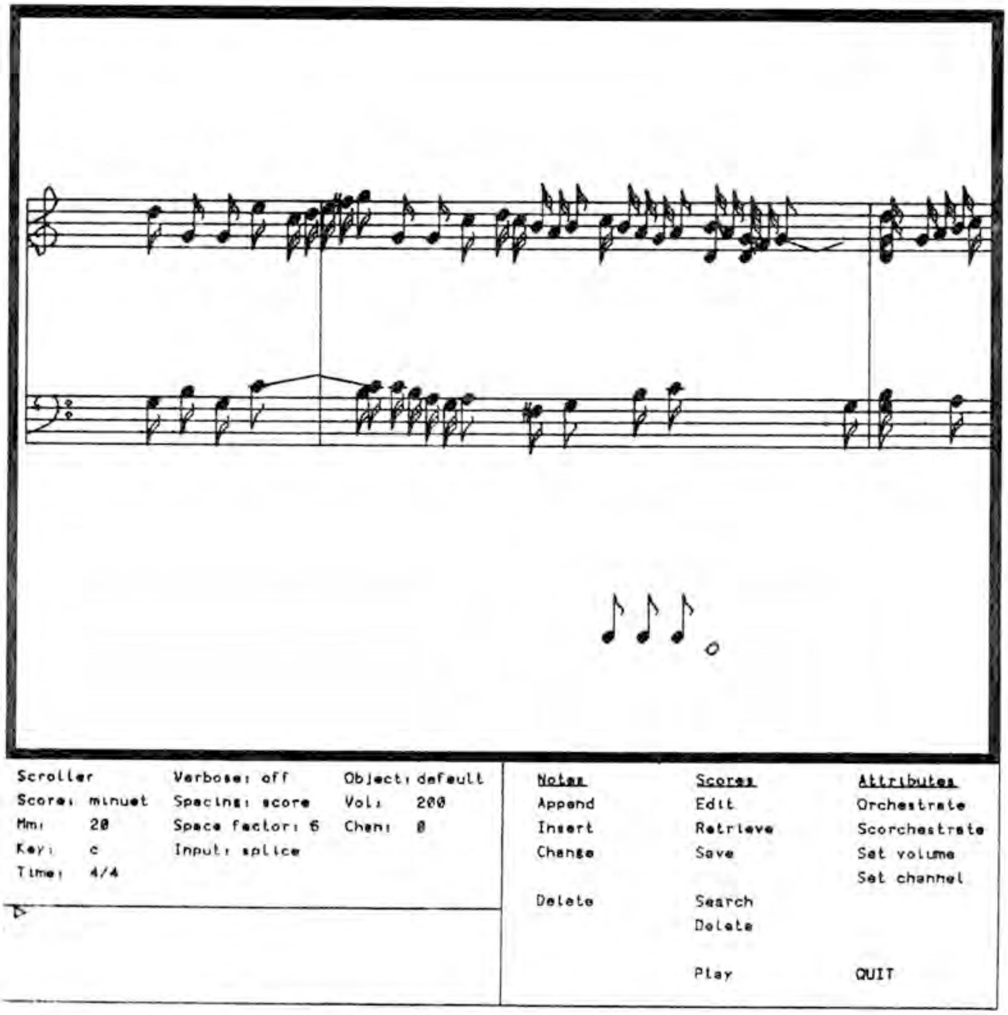
Figure 1 — Display Organization

Early versions of *ludwig* were influenced strongly by the similarity between a score editor and a text editor. This influence extended even as far as the naming of the commands available to the composer. We found, however, that composers could not understand the computer science related nomenclature, so we switched to much more user and application-oriented terminology.

As shown in Figure 1, we have also tried to logically organize the visual presentation of the commands on the display screen. Commands pertaining to notes are in one column, those pertaining to scores in another, and those pertaining to musical attributes in a third. The purpose is to provide a basic hierarchy or grouping of commands to aid the composer in remembering each command's function.

3.2. *Iconic Trackers*

The composer begins to select a command by positioning the *tracking-cross (tracker)* over the command name in the *menu*. This is accomplished simply by moving a *cursor* on the *graphics tablet*. On depressing a button on the cursor, the command is initiated. As feedback, the intensity of the command name in the menu is brightened.

Throughout *ludwig*, the tracking-cross is often replaced by an "icon" in order to convey some information in a pictorial manner as opposed to a wordy message. Some of the iconic trackers used in *ludwig* are shown in the figures of this paper. Among many others that are not shown are a "buddha" icon to request patience from the user when the system is busy, a question mark when the user attempts an erroneous or unknown interaction, and a "tty" icon when the user is to type some information.

The graphics package used [Reeves 78b], enables us to make great use of iconic trackers by providing mechanisms for rapid tracker switching. They are utilized in many other systems implemented in our environment.

Besides indicating system state and prompting the composer in a very concise manner, the information is displayed at the user's highest point of concentration on the screen — at the tracker. This allows the user to focus his attention, avoiding wasted hand and eye movements. Because this is a very rapid feedback mechanism, the composer is never left having performed some interaction and wondering whether the system received the message or not.

3.3. *The Note Input Tool*

Many music systems utilize piano-like keyboards as input devices (eg., [Vercoe 75]). We feel that too heavy a reliance on such transducers "locks the composer in" to a particular mode and notation in composing. We decided to investigate input techniques lending themselves to different forms of notation. Basing the techniques on graphical interaction where many different interactive tools can be constructed with software has greatly helped our investigations.

In this section, we will illustrate one such interactive technique — the note input tool — as implemented in *ludwig*. Figure 2a shows the note input tracking cross being positioned over the desired pitch. On depressing the button on the cursor, a "marker note" symbol appears at the indicated pitch. Concurrently, the tracker is replaced by a sequence of notes. This is shown in Figure 2b. This sequence of notes "tracks" or follows the motion of the cursor on the tablet. The roles of the tracking-cross and the menu are now reversed. Instead of the conventional stationary menu and moving pointing tool, we have a moving menu and a stationary pointer. By placing the note of the desired duration over the marker note symbol (i.e., moving the menu), and releasing the cursor button, a note is input. This is shown in Figure 2c. The ledger lines, tail direction, bar lines, and note spacing are automatically handled. While our long-winded explanation may indicate otherwise, a new composer can quickly learn to input notes as fast or faster than on manuscript paper.

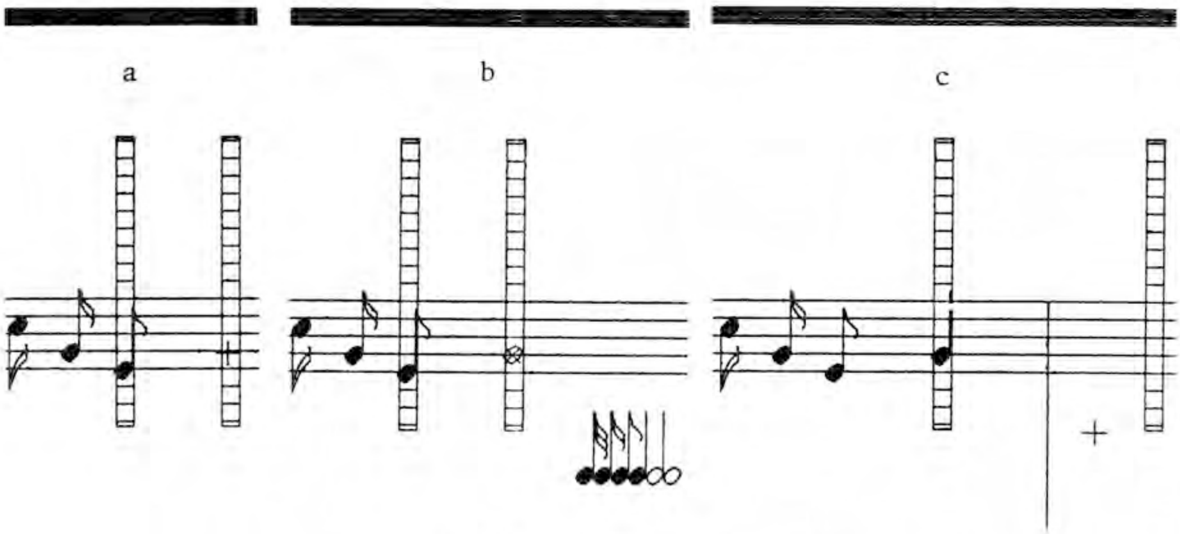

Figure 2 — Input of a Note

Our note input tool is also very flexible. By positioning the marker note in the first set of ledger lines, the new note is chorded with all notes above or below it. This allows polyphonic scores to be edited with *ludwig*. By manipulating other input transducers, the composer can also enter ties, delete the last note and make pitch corrections on the last note. A rest input tool is also available — the protocol is identical to the note tool except the sequence of notes becomes a sequence of rests.

3.4. *Orchestration*

One of the best aids in presenting new concepts is a good analogy. This is especially true for concepts used in human interfaces to computers. Thus one technique that we use is often called the "paint pot" technique. We use it to orchestrate the notes of a score (i.e., assign instruments to the notes). After selecting the "orchestrate" lightbutton, the tracking-cross becomes a paint-brush, and our palette of timbral colours (instruments) appears as a menu at the bottom of the score viewport. The "colour" currently on the brush is highlighted. Simply pointing at notes with the brush, and depressing the button, will orchestrate them with the current instrument. At any time, the composer is able to change the "colour" on his brush either by dipping into his palette (i.e., pointing at the desired instrument in the instrument menu), or scrolling through the list of instruments — using a hardware *slider* — until the desired instrument is the one that is highlighted.

Why is this a successful interactive tool? First, in accessing the various instrument files, no typing is done. Second, there is no burden on the composer's memory to recall the names of the instruments in his directory or their spelling. *Ludwig* is able to extract all instrument files from the directory — and them alone — and list them in the "palette", because of the underlying musical data structures [Buxton 78a], and [Buxton 78e]. The use of sliders to change the "current" instrument in the palette means that the cursor need not move down. Orchestration is carried out with the cursor in one hand, and the sliders — to do instrument selection — in the other. The resulting economy of motion results in a more smooth, efficient, and congenial interface. Finally, it is important to note that what was described is an instance of a general protocol. Once learned, the same technique can be used for several other parameters, for example, all of the commands in the "Attribute" list shown in Figure 1.

3.5. *Score Navigation*

It is important to enable the composer to "get around" the score, the notes of which may or may not be within the current *viewport* (i.e., what is displayed on the screen). In *ludwig,* there are three navigation techniques. To begin with, any note in the current viewport can — for editing purposes — become the *current* note by pointing at it and depressing a button on the cursor. That is to say, the graphics system is flexible enough to do hit detection with immediate response. If, on the other hand, the note we want is not in the current viewport, we can *scroll* the score across the screen in real-time at the touch of one of the hardware *sliders.* This ability to scroll through the score — especially during performance — is a good example of the importance of *dynamic* (as opposed to static) graphics techniques. A storage tube or most video devices simply could not support this type of interaction.

In the third navigation tool, we again want to examine part of the score not in the current viewport. This time we take an alternative approach. We point at the *light button* "search", and press a button on the cursor. What appears on the screen (seen in Figure 3) is a "time-line" representing the entire duration of the score. On this time-line, we see two "angle brackets" which indicate what portion of the score is in the current viewport. By placing the tracking-cross (which has become a magnifying-glass icon — to indicate "searching") anywhere on the time-line, and depressing the cursor button, the viewport will move and the portion of the score within it will become visible. The use of labels (not shown) further strengthens this technique, providing something analogous to orchestral rehearsal marks. A simple addition to *ludwig* would display quantitative labelling on the time-line.

The first two navigational techniques exploit locality of search. They will be used most often and hence their protocols have been designed to be very simple and efficient. The third technique solves a harder problem by presenting the composer with a simple yet sufficient graphical representation of the entire score and a very simple tool to use with it. Significantly, all of these techniques also apply to forms of notation other than CMN.

3.6. *Performance*

At any time during the development of a score, the composer can attain acoustic feedback by executing the "play" command. The score is performed on the digital synthesizer. Such instantaneous feedback in the sonic domain is very important to complement the graphical feedback in the visual domain. We believe interactive systems of the future will develop multi-dimensional feedback techniques such as this to communicate as much information as possible to their users.
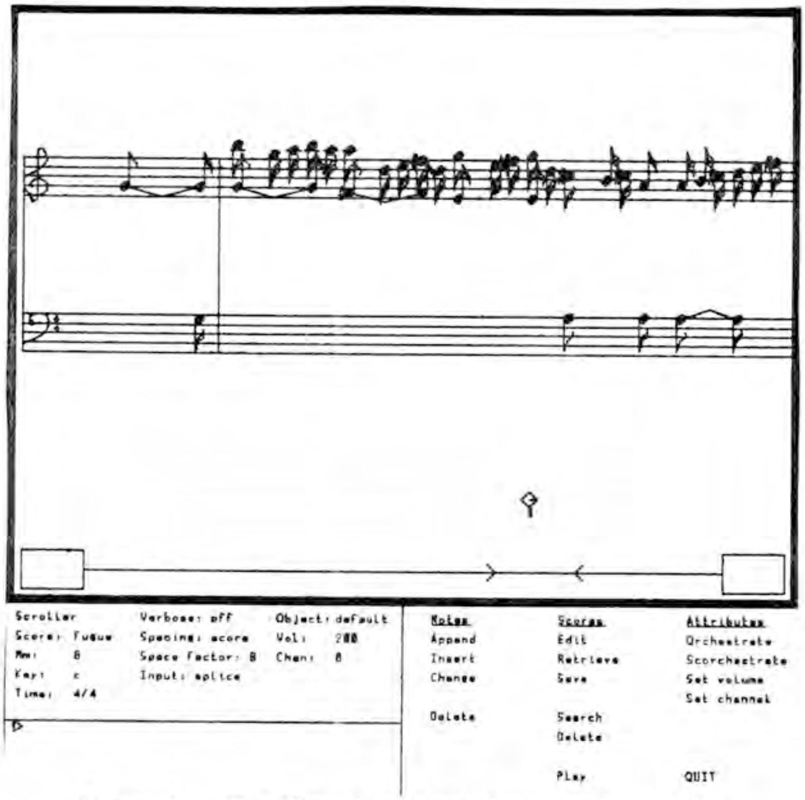
Figure 3 — The Time-Line Used for Score Navigation

4. *Some Characteristics of Good Interactive Techniques*

The above techniques work well because they adhere to a number of principles which seem to apply to the design of good interactive dialogues:

1. The nomenclature used is oriented towards and appropriate for the application.

2. The techniques are refined through careful observation of their use by real users, that is, the intended users of the ultimate system.

3. Screen layouts are very carefully designed and refined.

4. A small but effective set of input transducers is used. Too many can lead to wasted actions; too few can lead to cumbersome interactions.

5. The techniques are natural, easy to learn, not cumbersome. Musicians can begin using *ludwig* within an hour after being introduced to the system.

6. The feedback given in response to user input is iconic and is appropriate for the task at hand. The note input tool is a good example of this.

7. The feedback occurs rapidly. One would use different techniques if the system could not respond instantaneously to user input.

8. The feedback occurs predictably. Unpredictable response is even worse than predictably slow response, leading to frustration, tension, and anxiety.

9. The technique implemented is a powerful one, giving the user many degrees of freedom

and control. The navigation tool, for example, allows the user to move with ease anywhere in the score.

10. The technique allows the user to focus his attention, avoiding wasted hand and eye movements. The changing tracker and note entry techniques are good examples of applications of this principle.

11. The proper visual ground or context is presented. Thus with the score navigation tool we know where we are relative to where we could be.

12. It is easy to escape from or abort the action.

13. It is difficult to make mistakes, and the system is robust enough to minimize the damage from mistakes that are made. The changing tracker technique presents diagnostic information to the user very quickly. The note entry technique helps guarantee legitimate input.

14. As few demands as possible are made on the user's memory. In the "orchestrate" command, for example, he need not remember specific file names, and can find an instrument, albeit inefficiently, even if he forgets where in the file system it is located.

15. The various techniques embedded in the system share a unity of protocol — a common syntax, set of visual conventions, and interactive style. This, along with some of the other characteristics listed above, allows the user to focus on the application, not the communication.

16. Finally, the techniques are very device dependent. Why should we not expect good interactive mechanisms to be so? Do we expect good device-independent flute-like instruments, chiseling tools, or flying vehicles? Do we expect works of art to be independent of the medium in which they are created?

5. *Notes and Acknowledgements*

Readers interested in graphical interaction should see [Martin 73], [Nickerson 76], [Treu 76], and especially [Baecker 79], which lists relevant recent papers and which takes issue with the current dogma of "device-independence." Examples in other domains of the kind of interactive techniques described above can be found in [Baecker 76], [Tilbrook 76], [Tuori 77], and [Crossey 77]. The authors are indebted to the National Research Council and to the Social Sciences and Humanities Research Council for financial support, and to the following who are among our many collaborators who have made enthusiastic contributions: Guy Fedorkow, Al Fogels, Mark Green, Les Mezei, Tom O'Dell, Sanand Patel, Robert Pike, Larry Sasaki, Dave Sherman, K.C. Smith, Mike Tilson, and The Canadian Electronic Ensemble.

6. *Bibliography/References*

[Baecker 76] Ronald M. Baecker, David M. Tilbrook, and Martin Tuori, NEWSWHOLE: A Newspaper Page Layout System, 10 minute 3/4" black-and-white video cartridge, Dynamic Graphics Project, Computer Systems Research Group, University of Toronto.

[Baecker 79] Baecker, R., Towards an Effective Characterization of Graphical Interaction. To be presented at IFIP W.G. 5.2 Workshop on Methodology of Interaction, Seillac, France.

[Buxton 77a] Buxton, W., A Composer's Introduction to Computer Music. *Interface 6:* 57-72.

[Buxton 77b] Buxton, W., *Computer Music 1976/77: a directory to current work.* Ottawa: The Canadian Commission for Unesco.

[Buxton 78a] Buxton, W., Design Issues in the Foundation of a Computer-Based Tool for Music Composition. *Technical Report CSRG-97.* Toronto: University of Toronto.

[Buxton 78b] Buxton, W., Fedorkow, G., The Structured Sound Synthesis Project (SSSP): an Introduction. *Technical Report CSRG-92,* Toronto: University of Toronto.

[Buxton 78c] Buxton, W., Fedorkow, G., Baecker, R., Reeves, W., Smith, K.C., Ciamaga, G., Mezei, L., An Overview of the Structured Sound Synthesis Project. Paper presented at the Third International Conference on Computer Music, Dept. of Music, Northwestern University, Evanston Illinois.

[Buxton 78d] Buxton, W., Fogels, A., Fedorkow, G., Sasaki, L., Smith, K. C., An Introduction to the SSSP Digital Synthesizer. *The Computer Music Journal 2.4, 28-38.*

[Buxton 78e] Buxton, W., Reeves, W., Baecker, R., Mezei, L., The Use of Hierarchy and Instance in a Data Structure for Computer Music. *The Computer Music Journal 2.4, 10-20.*

[Core 77] Status Report of the Graphic Standards Planning Commitee of ACM/SIGGRAPH, *Computer Graphics,* Vol. 11, No. 3.

[Crossey 77] Crossey, S., An Interactive Graphical Source Language Debugging System, M.Sc. Thesis, Department of Computer Science, University of Toronto.

[Deecker 77] G.F.P. Deecker and J.P. Penny, Standard Input Forms for Interactive Computer Graphics, *Computer Graphics,* Vol. 11, No. 1, pp. 32-40.

[Fedorkow 78] Fedorkow, G., Buxton, W., Smith, K. C., A Computer Controlled Sound Distribution System for the Performance of Electroacoustic Music. *The Computer Music Journal* 2.3: 33-42.

[Foley 74] James D. Foley and Victor L. Wallace, The Art of Natural Graphic Man-Machine Conversation, *Proceedings of the IEEE,* Vol. 62, No. 4, pp. 462-471.

[Green 79] Mark Green, A Graphical Input Programming System, M.Sc. Thesis, Department of Computer Science, University of Toronto.

[Kasik 76] David J. Kasik, Controlling User Interaction, *Computer Graphics,* Vol. 10, No. 2, pp. 109-115.

[Martin 73] James Martin, *Design of Man-Computer Dialogues,* Prentice-Hall, Englewood Cliffs, N.J.

[New England Digital 78] New England Digital Corp., Synclavier Instruction Manual. Norwich Vermont: New England Digital Corp., PO Box 305.

[Newman 79] Newman, W., Sproull, R., *Principles of Interactive Computer Graphics (Second Edition).* New York: Mc Graw-Hill.

[Nickerson 76] R.S. Nickerson, On Coversational Interaction with Computers, appears in [Treu 76], pp. 101-113.

[Pulfer 70] Pulfer, J., K., Computer Aid for Musical Composers. *Bulletin of Radio and Electrical Engineering Division* 20.2: 44-48.

[Reeves 78a] Reeves, W. T., A Device-Independent General Purpose Graphics System in a Minicomputer Time-Sharing Environment. *Technical Report CSRG-93,* Toronto: University of Toronto.

[Reeves 78b] Reeves, W., Buxton, W., Pike, R., Baecker, R., Ludwig: an Example of Interactive Computer Graphics in a Score Editor. Paper presented at the Third International Conference on Computer Music, Dept. of Music, Northwestern University, Evanston Illinois.

[Smith 78] Smith, L., MUS10 User Manual. Stanford: work in progress, CCRMA, Dept. of Music, Stanford University.

[Tanner 72] Tanner, P., MUSICOMP, an Experimental Aid for the Composition and Production of Music. *ERB-869.* Ottawa: N.R.C. Radio and Electrical Engineering Division.

[Tilbrook 76] Tilbrook, D., A Newspaper Pagination System, M.Sc. Thesis, Department of Computer Science, University of Toronto.

[Treu 76] Siegfried Treu [Editor], User-Oriented Design of Interactive Graphics Systems, Proceedings of the ACM/SIGGRAPH Workshop, Pittsburgh Pa., Oct. 14-15.

[Tucker 77] Tucker, W. H., Bates, R. H. T., Frykberg, S. D., Howarth, R. J., Kennedy, W. K., Lamb, M. R., Vaughan, R. G., An Interactive Aid for Musicians. *Int. J. Man-Machine Studies* 9: 635-651.

[Tuori 77] Tuori M., Tools and Techniques for Computer-Aided Animation, M.Sc. Thesis, Department of Computer Science, University of Toronto.

[Van Den Bos 78] Jan Van Den Bos, Definition and Use of Higher-level Graphics Input Tools, *Computer Graphics*, Vol. 12, No. 3, pp. 38-42.

[Vercoe 75] Vercoe, B., Man-Computer Interaction in Creative Applications. Cambridge: unpublished manuscript, Studio for Experimental Music, MIT.

[Wallace 76] Victor L. Wallace, The Semantics of Graphic Input Devices, *Computer Graphics*, Vol. 10, No. 1, pp. 61-65.