

The Music Notepad

Andrew Forsberg, Mark Dieterich, and Robert Zeleznik
Brown University
Department of Computer Science
Providence, RI 02912
(401) 863-7693; {asf,mkd,bcz}@cs.brown.edu

ABSTRACT

We present a system for entering common music notation based on 2D gestural input. The key feature of the system is the *look-and-feel* of the interface which approximates sketching music with paper and pencil. A probability-based interpreter integrates sequences of gestural input to perform the most common notation and editing operations. In this paper, we present the user's model of the system, the components of the high-level recognition system, and a discussion of the evolution of the system including user feedback.

KEYWORDS: user interface, interaction, music notation, gestural input, gesture recognition, handwriting recognition, direct displays.

INTRODUCTION

There are a number of situations that revolve around informally notating music¹, such as when a composer wants to jot down an idea, when a teacher explains theory to students, or when a musician wants to visualize (i.e., notate) a musical idea. Despite the many advantages of applying computers to music notation (e.g., for synthesizing sound, neatly formatting and rendering notation), people in fact resort to using just paper and pencil for many tasks even when computer solutions are available. This paradox derives from the nature of the interfaces for typical music applications.

Most computerized music notation systems employ standard windows, icons, menus, and point-and-click (WIMP) user interfaces (UI's) as well as a transition to keyboard "hotkeys" for frequently used functions. In some cases, these systems offer advantages over paper and pencil notation such as rapid data entry, editing flexibility, automatic formatting, synthesized sound, and high-quality printing. However, the user's model for computerized systems is very different from the model of paper and pencil notation. Based on discussions

¹ See [1] and [13] for further information on musical terms and common music notation

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

UIST '98, San Francisco, CA

© 1998 ACM 0-58113-034-1/98/11... \$5.00

with a number of musicians and composers we believe a fundamentally different music notation interface based on a pen-based UI will be more desirable and of equal or greater value than a WIMP-based UI.

The Music Notepad attempts to be an interactive electronic sheet of music paper. Unlike WIMP UIs, the Music Notepad is characterized by a portable display surface (a Wacom PL-300 Display Tablet) that can be directly drawn upon with a stylus. To support what can be done with pencil and paper interfaces, the Music Notepad interprets gestures specified by the user with a stylus to create notation. Moreover, gestures can also be used to perform more powerful editing operations, to professionally format notation, and to synthesize instrumental sounds based on the notation.

The following sections present previous work, the user's model of the system, the details of our recognition methodology, and a discussion of the formative design of the system through user feedback.

PREVIOUS WORK

There are two common approaches to music notation: paper and pencil, and computer-based systems. Paper and pencil has many advantages— notably low cost, simplicity, and portability. Music can be notated by drawing symbols on inexpensive paper. Sheets of paper can be copied and distributed very easily. However, producing high quality publishable documents by hand requires great skill and editing operations that are difficult to perform. Other desirable concepts such as automatically performing written notes are not possible.²

There are two main flavors of computer-based systems: sequencer and notation systems. The goal of sequencers is to enter and perform synthesized music, whereas notation programs are intended only to produce high-quality printed scores. Both are successful in addressing some aspects of the problems of paper and pencil systems such as improved editing operations (e.g., editing individual or groups of symbols and transposing), and synthesizing or printing a high-quality version of the music that has been entered.

²Written music can be performed by one or more skilled performers. However, there are often significant barriers to becoming a skilled performer such as years of practice and expense.

However, the handwriting techniques for creating standard music notation [13] learned by many musicians bears little resemblance to the music software UIs. Instead, they tend to use the well-established WIMP-based UIs. These UIs are sometimes augmented with a MIDI input device such as a piano keyboard. The primary advantage of a WIMP interface is its simplicity and learnability. In addition, many applications such as Finale [6] or Cakewalk [3] provide mechanisms that allow users to transition from using the WIMP interface to using keyboard “shortcuts.” Shortcuts are typically learned over time by displaying each shortcut key next to the equivalent WIMP interface command. Gradually, users tend to transition to using only the shortcuts resulting in very fast, although indirect, user input. The use of MIDI devices for input to a notation system at first seems appropriate, but is still not ideal because nearly all input from MIDI devices requires editing. This problem is rooted in the need to be skilled performer of a particular MIDI device.

There have been several research systems for music notation. The Mockingbird system [12] was a pioneer in the use of a graphical UI and MIDI keyboard. However, because this system relies heavily on MIDI keyboard input nearly all input requires skilled performance and editing. The system also depends on a WIMP interface-style to edit notation.

Buxton [2] developed a system which included a set of gestures for specifying notes and rests (see Figure 1). While this is an effective set of gestures for very basic note entry, the system does not provide gestures for other fundamental notations such as stem direction, accidentals, and beams. Although we have integrated Buxton’s gestures in the Music Notepad, there are some situations where the gesture scheme is cumbersome. For example, multiple short notes often appear in long sequences, but the gesture for creating short notes is unfortunately relatively complicated.

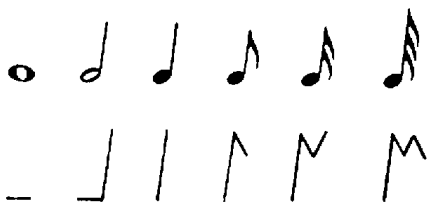


Figure 1: A set of gestures developed in [2] for creating notes of various durations. Rests are created by mirroring these same gestures around the horizontal axis.

A similar gestural component is embedded in an otherwise conventional WIMP interface in both the NoteWriter and the NoteAbility systems [14]. These systems provide simple gestural alphabets that are related to Buxton’s gestures both in functionality and limitations.

As we prepared the final version of this paper, we learned about a similar system GSCORE [17]. GSCORE provided both WIMP based and gestural based methods for music no-

tion. Although both systems share many comparable features, the Music Notepad provides unique functionality, such as allowing the user to retain a “non-finished” look (making it appear closer to a pen and paper look), extensive support for editing notations, and score playback.

SYSTEM DESCRIPTION

This section describes the user model of the system followed by the details of the components used to integrate the various types of input.

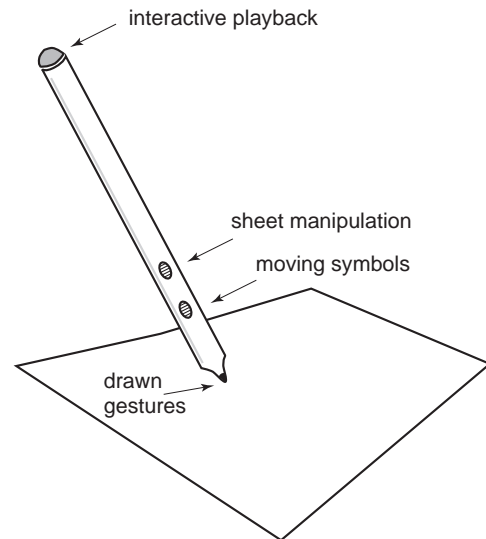


Figure 2: The stylus has four buttons for controlling all Notepad operations.

User Model


Music Notepad is intended to appear to the user as an interactive sheet of music paper. Thus the user accesses all functionality gesturally with a stylus. There are four classes of gestural operations that correspond to the four buttons of the stylus (see Figure 2). Marking gestures, drawn with just the tip of the stylus, leave ink trails on the display; sequences of these gestures are interpreted as either handwritten commands or as operations for creation and deletion of musical notations, marking menu [9] invocation and selection, as well as region selection. The lower button of the pen is used to perform direct manipulation operations for changing note pitches and graphical placement of symbols. The second lowest button allows the user to slide the “music paper” across the display screen. Finally, the eraser button of the pen is used for playback of the entire score or for interactive playback of regions of musical notation.

Creating notation symbols The most basic operation in the Music Notepad is the creation of *notes*. Users can create notes using the gestures shown in Figure 3. Gestures convey both spatial and symbolic information. Note creation gestures consisting of only distinct line segments are centered on the first point of the gesture. Creation gestures involving drawn noteheads (called “scribbled” noteheads) are placed

Legend:

- drawn by computer
- drawn by user

Example Gesture:



dot signifies start of gesture





























































	=>		create new note		=>		delete note with squiggle
	=>		create new note with sharp		=>		delete note with lasso
	=>		create new note with flat		=>		delete note with lasso & squiggle
	=>		create new note with low volume		=>		delete group of beamed notes
	=>		create new note with medium volume		=>		delete single notes from a group of beamed notes
	=>		create new note with loud volume		=>		delete beam over group of notes
	=>		create new whole note		=>		halve note duration
	=>		create new whole note		=>		double note duration
	=>		create new half note		=>		beam all notes, change note durations, match beam slope to gesture
	=>		create new quarter note		=>		halve beamed notes duration
	=>		create new eighth note		=>		double beamed note duration
	=>		create new sixteenth note				
	=>		create new thirtysecond note				
	=>		create new whole rest	piano			
	=>		create new half rest				name of instrument for staff
	=>		create new quarter rest				
	=>		create new eighth rest	piano	oboe		
	=>		create new sixteenth rest				writing "oboe" on staff changes the instrument
	=>		create new thirtysecond rest	oboe	=>		

Figure 3: A taxonomy of marking gestures and their associated operations.

based on the computed center of the notehead. Scribbled notes default in duration to a quarter note, however, the size of the drawn notehead also determines the note's volume. That is, a small notehead plays softer than a fat notehead. If a scribbled notehead has a tail that extends above or below the notehead, then a sharp or flat is associated with that note. After a new note is created, the synthesizer sounds that note. If the note is created above other notes, a chord is recognized and all notes are played back simultaneously.

The duration of an existing note can be modified by drawing a slash gesture through its stem. A gesture from the left to the right will half its duration, whereas a gesture from the right to the left will double its duration.

A quick tap on a notehead will mark that notehead as staccato. The staccato marking of a notehead will disappear if it is already notated to play staccato and is tapped with the stylus. This quick tap gesture is similar to the meaning of the staccato marking. Pressing on a notehead for a longer period of time will toggle whether that notehead has a "dotted" value.

Rests can be created by using the gestures from Figure 1 mirrored about the horizontal (to distinguish from the "create-a-note gestures".) Rest durations can be doubled or halved in the same way note durations are modified; by drawing a short gesture through the rest.

Beams for a set of notes can be created by drawing a straight line above or below a set of notes. This position and angle of the gesture line determines the distance of the beam from the noteheads as well as the angle of the beam. In some situations, a specific set of notes can be accurately beamed by first lassoing the desired notes and then drawing the line above or below the set of notes. Once a group of notes has been "beamed," drawing a gesture line through the stems of notes in the group will half or double the duration of those notes, depending on the direction of the gesture line. Scribbling on a beam erases a beam and ungroups the notes associated with it. Users can also adjust the slant of a beam by directly manipulating either the left or right side of the beam. If the user grabs the middle of the beam, then the height of the beam is adjusted.

Accidentals for an existing note can be added using a marking menu. If the user holds the stylus on a notehead for some short period of time, a radial menu appears showing the marking menu choices for accidentals. By dragging and releasing in the direction of a menu item, a sharp, flat, or natural is associated with the note. The final menu option clears all accidentals. If the user does not wait for the menu, the operation is still performed as with any other marking gesture. New notes with accidentals can also be created using a single gesture as previously described in the creating notation section.

Clefs and key signatures are created by drawing a dot or lasso

to indicate a location followed by a command or text. Commands are specified through handwriting or speech. For example, to change the key signature the user draws a dot at the desired location and writes "D major".

Editing The Music Notepad currently supports the most commonly used music notation editing operations: *deleting*, *copying*, *pasting*, and *region selection*. One or more objects can be deleted by scribbling on top of them. To *select a region*, the user draws a lasso around the region. Multiple regions can be specified by drawing multiple lassos. To *delete a selected region*, the user scribbles inside a lasso. There are also two single-stroke delete gestures derived from text editing notation (see Figure 4).



Figure 4: Two single-stroke delete gestures derived from text editing.

To copy a region, the user holds the stylus just above a lassoed region, presses the move button on the stylus (which creates a ghosted copy of the symbols in the lasso), and then releases the button at the location he or she wishes to place the copied symbols. To move a region, the user touches the tablet surface and presses the move button on the stylus, moves to the new location (dragging a copy of the lassoed symbols), and then releases the move button to place the lassoed symbols. This style of pen-based interaction is similar to Pick-And-Drop [15].

Instrument Assignment To assign a musical instrument to a staff, the user draws a gesture line from an instrument shown in the "instrument picker" (see Figure 5) to the staff the instrument is to be assigned to. Alternatively, the user can draw a dot on a staff and then write or speak the name of the desired instrument (see Figure 3).

Visualization Typically, music is printed on roughly 8.5" by 11" paper and two or more sheets are arranged in a row for viewing. However, our Wacom display tablet has a total display area of only 8" by 6". To increase the effective display area of our tablet, the user can create a division in the "music paper" on either side of the display tablet in order to display more of the musical score (see Figure 6). This is based on the perspective wall metaphor [11]. As a result, the user has increased context for the current working area and also makes it easier to navigate through the score.

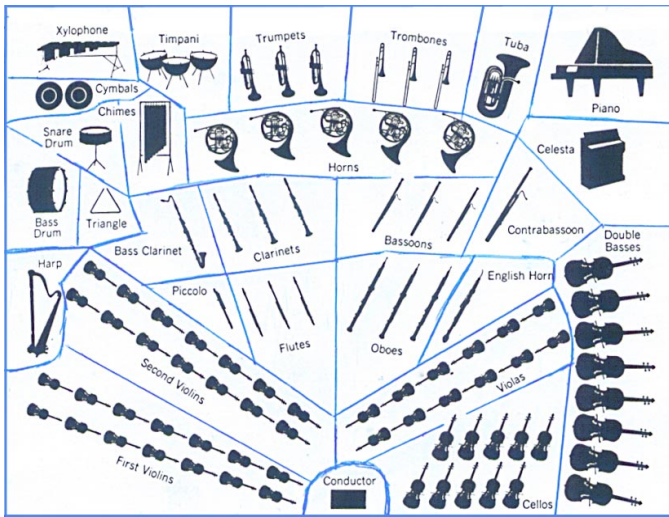


Figure 5: An orchestral instrument picker—the user draws a gesture line from an instrument to a staff to assign instruments.

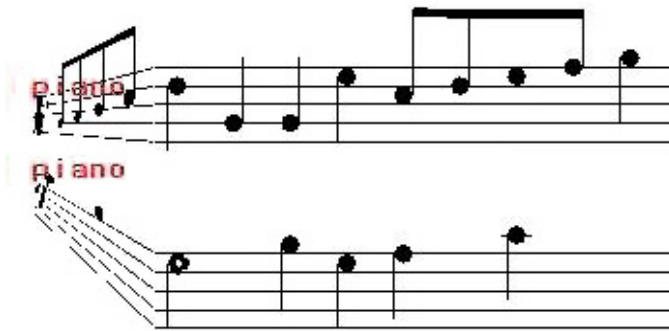


Figure 6: A Perspective Wall of music.

Audio Feedback In addition to hearing notes or chords when adding new notes, there are two other types of audio feedback in Music Notepad. First, the user can drag the other end (the eraser side) of the stylus over notes. As the stylus passes over notes, they are played back. Second, using a different gesture with the eraser end of the stylus, the entire piece will be played back.

UI System Components

This section describes the Music Notepad’s UI system components. The flow of input through the system is illustrated in Figure 7. The tokenizer converts user input into feature vectors, called *tokens*, data structures that describe an input’s spatial and textual content. Each token consists of 25 features including, for example, the time to enter the user input, the length of a stroke, the object under the starting point of a stroke, and the number of self intersections. Tokens are posted to the accumulator where they are examined by command recognizers, called *RECOGs*. Each RECOG has a *customized procedure* for estimating the probability that the posted tokens are “what it is looking for” by analyzing

the token’s features, often in the context of the existing notations. For example, a RECOG that erases lassoed objects returns a probability based on how well the RECOG considers that one token describes a lasso around the objects and that the other describes a squiggle within the lasso.³ If a RECOG’s probability is greater than all other RECOG probabilities and it is also greater than some threshold (we used 50% certainty), then the RECOG is executed and the accumulator is cleared. Conceptually, our approach is similar to the unification-based multi-modal integration from [8].

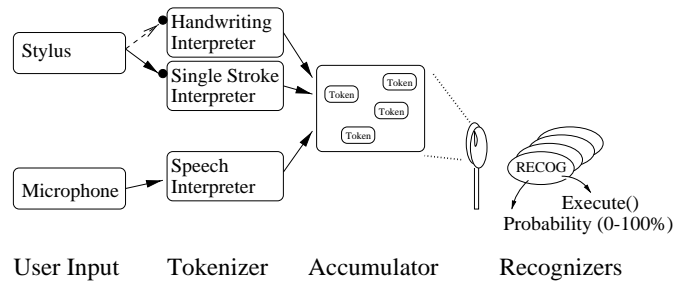


Figure 7: Overview of system components: User input is transformed to a token data structure that describes the spatial and textual information of input. Command recognizers (called “RECOG”s) examine the accumulated tokens. If the top probability of a RECOG exceeds some threshold, its function is executed.

We employed the concept of an *Arbitrator* for resolving situations where multiple RECOGs reported high probability that they were recognized. For example, the RECOGs that identify the gesture to insert an end-of-measure symbol and to insert a quarter note (based on gestures in Figure 1) both look for nearly the same input tokens. We chose to keep individual RECOGs simple and in the case where there is a conflict, the RECOGs that reported high recognition probabilities were passed to the Arbitrator which is responsible to resolve the conflict. The Arbitrator consists of case-specific code for expected conflicts. For example, in the end-of-measure and quarter note scenario, the Arbitrator has a resolution handler that has been hand-coded to inquire additional context to resolve the situation.

There are some important details to the UI system components. Stylus input can be directed to one of two tokenizers—a single stroke interpreter or a handwriting interpreter. By default, all stylus input is directed to the single stroke interpreter. If the first single stroke gesture drawn is a “dot,” then the system recognizes this as an out-of-band gesture and channels all remaining stylus input through a handwriting recognizer. The “dot” gesture generally acts as a spatial marker for the subsequent handwritten command.

Our tokenizer interpreters are Rubine’s gesture recognition system [16] for single strokes (such as those from Figure 1),

³Since probabilities are in essence arbitrary, we had to iteratively refine different techniques for computing and combining probabilities, with the result being effective but “ad hoc.”

the Calligrapher [4] system for handwriting recognition, and the In-Cube speech recognition system [7]. Additional support code translates the interpreted input to our token data structures which are then posted to the accumulator.

DISCUSSION and USER FEEDBACK

The design of the Music Notepad has undergone a number of iterations guided by formative evaluations by small groups of musicians and composers. The following discussion presents some of the results of our user experiences and a description of how our designs changed in response to that feedback.

A variety of user gestures

Although all users were instructed with both a demonstration and a description of the gestures used in the system, we found wide variation between individual performances of the same gesture. Figure 8 illustrates some of the varieties of four common, simple gestures.

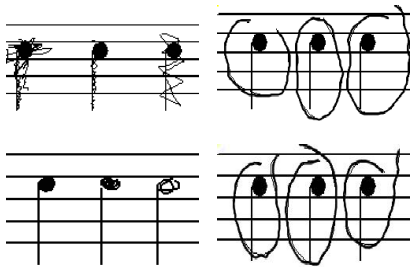


Figure 8: Different user styles for drawing the same gesture. Clockwise from top left: Erasing notes by squiggling on them, selecting notes with a lasso, scribbling to create noteheads, erasing notes with a single gesture lasso.

In response to the range of gesture styles, we were able to redesign our gestures and develop more robust gesture recognition algorithms. By analyzing the actual drawn gestures from a number of users, we identified what types of features were important in different situations and used this information to fine tune the recognition probabilities in specific gesture RECOGs.

Based on informal interviews of users who had tried the Music Notepad, we found no negative reactions to learning and using the gestural style of interaction. In addition, some users, including musicians, indicated they would prefer using a completed version of the Music Notepad over existing alternatives. The PalmPilot reflects similar learnability issues and has gained widespread acceptance despite its use of the idiosyncratic graffiti alphabet and the time required to master it. Based on these reactions and the similar idiosyncratic style of gesturing between the two systems, we believe the Music Notepad would have similar acceptance.

Accurate placement of symbols

Some music notation symbols such as noteheads must be accurately positioned. If the user specifies a position with a single mouse sample (e.g., by pointing and clicking), we found

they often misplace the notehead. Since users want staves to be drawn at a standard printed size, the spacing between staff lines is relatively small. Consequently, as predicted by Fitt's law, users have increasing difficulty in placing notes accurately as they work faster. Each time a note is misplaced, the user must perform at least one additional editing operation to correct the mistake.

In response to our early user experiences with gestures like those of Buxton (see Figure 1), we developed an alternate method for entering notes. With this method, users position a notehead by "scribbling in" a gesture that looks like a notehead. There are several differences in this method for entering notes. First, this gesture is more accurate than the point-and-click approach because the position of the note is specified by the *average* position of the multiple samples defining the scribbled notehead. Second, this gesture maps directly to how a notehead is drawn on paper. Third, since this gesture can act as the image for a notehead, we can avoid the distraction of replacing the drawn gesture with a different image. Last, this technique can be slower than the point and click technique and it does not convey the note duration.

Visual Representation

In addition to supporting What-you-see-is-what-you-get (WYSIWYG), the Music Notepad also supports What-you-see-is-what-you-entered (WYSIWYE). The goal of WYSIWYE is to minimize the time a user spends understanding the effects of an action. There are two instances of WYSIWYE in the Music Notepad: sketchy noteheads and delayed auto-formatting.

Sketchy noteheads When notes are entered by sketching a notehead, the notehead is represented by *exactly the line the user sketched* instead of replacing the line with a perfect notehead.

Delayed auto-formatting In our evaluation of existing music software systems, we found that each one reformats some subset of the notation every time a new symbol is created. While automatic reformatting is a useful feature, it can also be distracting—especially since the look of the document is often irrelevant during informal music entry. In response to the reaction of some users, we delay this automatic formatting until the user specifically requests it.

A major issue with WYSIWYE is ensuring that gestures are initially interpreted correctly. If not, this may lead to reformatting errors that must be painstakingly located by the user. Filled and hollow notes, for example, can be difficult to distinguish even by humans from a sketched notehead. We think it may be effective to incorporate feedback for marking ambiguous notes in a similar way to Microsoft Word's "squiggly underline" technique for highlighting misspelled words.

Marking Menus and Direct Input

There are many advantages to a direct draw environment, however, one disadvantage is that the user's hand can block

part of the display. This is a problem when using marking menus in the traditional manner because candidate menu items are obstructed by the user's hand.

We propose two solutions to this problem: first, allow the user to lift their hand and the stylus tip from the tablet to view the choices of the radial menu. After viewing the choices, the user can select an item by touching the item with the stylus tip or cancel the operation with a different gesture. The second solution is not to display items underneath the user's hand. This requires sensing where the user's hand is and might be accomplished with a Wacom tablet by using the data that reports the orientation of the stylus.

Mouse versus Stylus Input

In order to validate the need for a stylus-based gestural interface, we also prototyped our system using a three-button mouse for input. We found that although some aspects of the mouse-based interface proved beneficial (e.g., the user's hand does not occlude the display), users considered most gestural interactions to be more difficult. Users had particular difficulty drawing gestures with the mouse that involved curved lines, especially handwriting and lassoing. Despite these difficulties, users found that the character of pencil and paper sketching was still preserved.

FUTURE WORK

There are many areas of future work for the Music Notepad:

- approaches to learning the gestural interface
- accounting for many different styles of drawn gestures
- more extensive use of natural speech
- apply framework to other 2D applications
- incorporate MIDI keyboard interface, voice input
- greater music functionality in order to perform user studies
 - supporting multiple voices per staff
 - visual management of score
 - better playback that incorporates notation / dynamics
- user studies
 - comparison with traditional GUI (e.g., paper and pencil and Finale)
- apply to specific area of music: e.g., jazz

CONCLUSIONS

The Music Notepad demonstrates a paper and pencil look-and-feel to a powerful computer music notation engine. Thus users can informally jot down music as well as edit, professionally format, and synthesize the music. Although the system is still incomplete, it has benefited from multiple stages of formative evaluation and development. Musicians and composers provided feedback that was used to redesign our gesture recognition algorithms and to improve access to the available functionality.

ACKNOWLEDGMENTS

This work is supported in part by the NSF Graphics and Visualization Center, Advanced Networks and Services, Alias/Wavefront, Autodesk, Microsoft, Sun Microsystems, and TACO.

REFERENCES

1. Ammer, C., "The Harper Collins Music Dictionary," New York: Harper Perennial, 1991.
2. Buxton, W., Sniderman, R., Reeves, W., Patel, S., and Baecker R., "The Evolution of the SSSP Score Editing Tools," *Computer Music Journal*, Issue No. 12, 3:(4), pp. 14-25, 1979.
3. Cakewalk Pro Audio, Cakewalk, Inc., <http://www.cakewalk.com/>.
4. Calligrapher, ParaGraph International, Inc., <http://www.paragraph.com/>.
5. "Common Music Notation," A free western music notation package written in Common Lisp, The Stanford University Center for Computer Research in Music and Acoustics (CCRMA), <http://ccrma-www.stanford.edu/>.
6. *Finale*, Coda Music Technology, <http://www.codamusic.com/>.
7. The In-Cube User Guide, available from Command Corporation, Inc., Atlanta, GA, 1997.
8. Johnston, M., Cohen, P. R., McGee D., Oviatt, S. L., Pittman, J. A., Smith I., "Unification-based Multimodal Integration", *35th Annual Meeting of the Assoc. for Computational Linguistics and 8th Conference of the European Chapter of the Assoc. for Computational Linguistics*, Madrid, Spain, July 7-12, 1997.
9. Kurtenbach, G. and Buxton, W. "User learning and performance with marking menus," *In Proceedings of ACM CHI '94 Conference on Human Factors in Computing Systems*, pp. 258-264, 1994.
10. MacKenzie, S., Sellen, A., and Buxton, W., "A comparison of input devices in elemental pointing and dragging tasks," *ACM SIGCHI - Human Factors in Computing Systems*, pp. 161-166, 1991.
11. Mackinlay, J. D., Robertson, G. G., Card, S. K., "The Perspective Wall: Detail and Context Smoothly Integrated," *In Proceedings of ACM CHI'91 Conference on Human Factors in Computing Systems*, pp. 173-179, 1991.
12. Maxwell, J. T. and Ornstein, S. M., "Mockingbird: A Composer's Amanuensis," Available as technical report from Xerox Corporation, January 1983.
13. McGrain, M., "Music Notation," Berklee Press Publications, Milwaukee, 1986.
14. Personal conversation with Keith Hamel, Opus1 Music Software, <http://debussy.music.ubc.ca/opus1/>.

15. Rekimoto, J., "Pick-And-Drop: A Direct Manipulation Technique for Multiple Computer Environment," *In Proceedings of UIST '97*, pp. 31-39, October, 1997.
16. Rubine, D., "Specifying Gestures by Example" *In Proceedings of ACM SIGGRAPH '91*, pp. 329-337, July, 1991.
17. Rubine, D., *The Automatic Recognition of Gestures*, PhD Thesis, School of Computer Science, Carnegie Mellon University, December, 1991.
18. M.T. Vo and C. Wood, "Building an Application Framework for Speech and Pen Input Integration in Multimodal Learning Interfaces," *In Proceedings ICASSP'96*, Atlanta, GA, May 1996.