CENTER FOR COMPUTER RESEARCH IN MUSIC AND ACOUSTICS
SEPTEMBER 1989

Department of Music
Report No. STAN-M-59

NUTATION: STRUCTURAL ORGANIZATION VERSUS GRAPHICAL GENERALITY IN A COMMON MUSIC NOTATION PROGRAM

Glendon R. Diener

CCRMA
DEPARTMENT OF MUSIC
Stanford University
Stanford, California 94305

NUTATION:
STRUCTURAL ORGANIZATION VERSUS GRAPHICAL GENERALITY IN A COMMON MUSIC NOTATION PROGRAM

Glendon R. Diener
Center for Computer Research in Music and Acoustics (CCRMA)
Stanford University
GRD%CCRMA-F4@SAIL.STANFORD.EDU

Abstract

This paper addresses the implementation tradeoff between structural organization and graphical generality in common music notation programs. It introduces the concept of *three-dimensional, transparent glyphs*: transparent, rectangular areas of drawing which may be arranged hierarchically into complex images and structures. Nutation, a glyph-based, object-oriented program for the NeXT™ computer, is described.

Introduction

Striking the delicate balance between *structural organization* on the one hand and *graphical generality* on the other is a major issue in the design of common music notation (CMN) systems. This problem, described by Donald Byrd as the "fundamental tradeoff" between *semantics* and *graphics* (Byrd 1986), is readily understood by imagining musical versions of the 'draw' and 'paint' programs available on many small computers. A musical draw program could facilitate high-level editing and performance operations by means of the data structure analogs of notes, staves, parts, and the like, but as a consequence would limit its visual universe to some finite collection of pre-defined symbols. By contrast, a musical paint program, by imposing no further organization on its data than that of a two-dimensional array of pixels, would gain graphical generality at the expense of its ability to perform musically meaningful operations on that data.

The thesis of this paper is that structural organization and graphical generality are by no means incompatible objectives, and that an adequate synthesis of the two is technologically feasible, at least as far as CMN is concerned. The paper introduces the program *Nutation*: the result of an on-going research effort in which a restricted form of three-dimensionality coupled with transparency combine to bring CMN to the NeXT™ computer.

The Imaging Model

Absolutely everything in a Nutation score is made out of basic building blocks called *glyphs*. As far as their visual aspect is concerned, these glyphs are simply

transparent, rectangular areas on the computer screen which may contain drawing. A note, a rest--any familiar image from musical notation--can be enclosed in a bounding rectangle and displayed in a glyph. But glyphs may also display user-created symbols, or multiple symbols, or, for that matter, no symbols at all.

Glyphs may be be dragged and repositioned on the screen by means of either user interaction or program control. Because they draw on a transparent background, whenever an individual glyph is dragged over top of another, the other's image shows through where not obstructed by the drawing of the first, with both glyphs' bounding rectangles remaining invisible. Just as in most draw programs, this process allows complex images to be built up by combining simpler ones. But Nutation goes one step further: glyphs can be *stacked* on top of other glyphs, new glyphs stacked on top of these, and so on. The model is that of a three-dimensional world[1] populated by piles of two-dimensional glyphs.

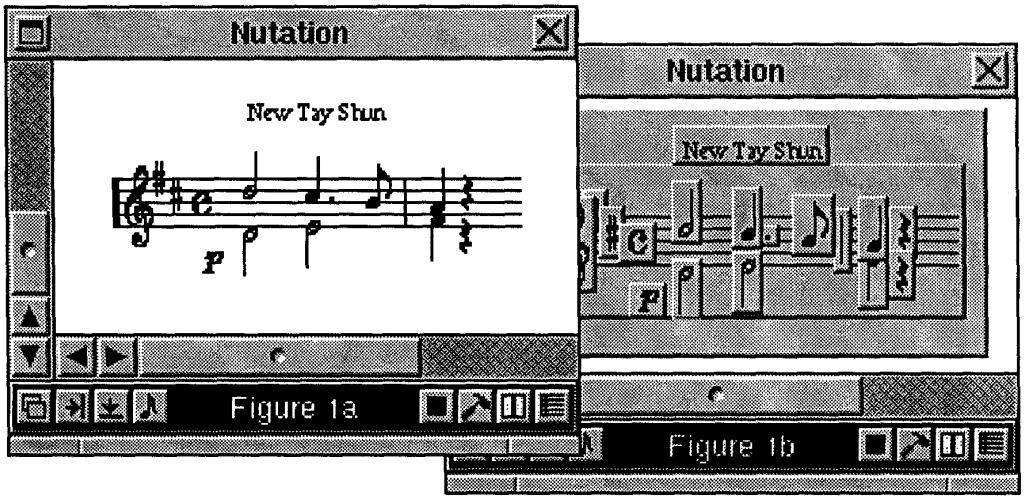

Figure 1

Combined with transparency, this three-dimensionality yields a simple yet powerful metaphor for score construction. To illustrate, suppose we have a set of glyphs representing pages, staffs, clefs, notes, and so on. Pages, usually opaque, are lain in the Nutation window. Typically, staffs are placed on top of pages. A transparent staff allows the page underneath to show through: opaque staffs occlude the underlying page, which itself may contain drawing and text. Clefs, notes, rests--all the trappings of traditional music notation--can then be stacked onto their appropriate staffs. Figures 1a and 1b show Nutation windows in which a short passage is depicted first as a flat, two-dimensional image, and then in a three-dimensional representation, with all glyphs made opaque and given 'thickness' to emphasize their underlying three-dimensionality.

[1]Actually, a two-and-a-half dimensionsal world, i.e. two continuous dimensions and one discrete dimension.

Structural Organization

Nutation is an object-oriented system, and glyphs communicate with each other by sending *messages*. The individual glyph, then, in addition to displaying its image, can respond to incoming messages by executing the corresponding *methods* (procedures). These methods may be used to modify the way the glyph displays itself, to change its location on the screen, to alter the way it is performed, and so on. Because methods are completely programmable, the Glyph's behavior is limited only by the skill and imagination of the programmer.

Nutation's three-dimensionality is responsible for more than just its imaging model. When one or more glyphs are piled on top of another, those glyphs are installed as subglyphs of the glyph they are piled upon, which is in turn a subglyph of the glyph it is piled upon, and so on, to the depth of recursion. These 'piles' of glyphs, then, are simply the visual analog of an underlying tree structure, maintained by Nutation in the form of a TTree (Diener 1989).

Nutation gains considerable power through this underlying hierarchical structure,[2] for the hierarchy functions as a network for channeling messages among glyphs. Messages sent to an individual glyph have the potential of being relayed automatically to the entire glyph subtree rooted in that individual. The program's user-interface component, for example, gains a ready-made mechanism for specifying the scope of such operations as moving, copying, and deletion: whenever one of these operations is done to a single glyph, it is done to the entire pile of glyphs resting upon it.

Another beneficiary of this hierarchical message-channeling scheme is Nutation's performance mechanism. When the root of the hierarchy is sent a *play* message, the message eventually trickles down through the tree until it bottoms out at the leaves (generally, some kind of 'note' glyph). Along the way, the flow of the message can be controlled by the glyphs it passes through. A 'page' glyph, for example, might relay the message to its subglyphs (usually 'staff' glyphs) successively from the top of the page to the bottom. Each 'staff' subglyph, upon receipt of the 'play' message, responds in turn by relaying the message from its leftmost subglyph to its rightmost. These subglyhs, generally 'note' glyphs, respond by instructing the Music Kit (Jaffe and Boynton 1989) to invoke the NeXT machine's Digital Signal Processor. The mechanism ensures that the score is performed in the correct top-to-bottom, left-to-right order, and allows for the accumulation of such global state as current key and time signature, dynamic level, etc., in a manner analogous to the way in which a human performer would read the score.

Note that the program's hierarchical scheme is by no means fixed. The number of hierarchical levels and their interrelationships constitute data which can be crafted by the musician as desired. Nutation's role is to provide a convenient framework for building such structures and for associating them with graphic representations. Given the openness of this architecture, it is easy to extend the system to embrace non-standard notational schemes. In fact, Nutation gives musicians the power to define their own notational worlds: worlds which may bear little or no resemblance to common notational practice.

---

[2]This hierarchy of glyphs is a so-called *instance* hierarchy, and should not be confused with the *class* hierarchy of by many object-oriented systems.

State of the Project

As of this writing, Nutation's imaging model is implemented, and a provisional set of glyphs has been programmed in Objective-C for the NeXT computer's 0.9 software release. The system can input, display, and perform simple scores, and can parse a subset of the *Music Kit*'s score file language (Jaffe and Boynton, 1989). Unfortunately, the user-interface to programming glyph behavior is hampered by Objective-C's edit-compile-test cycle. Clearly, a more interactive, conversational programming style is needed in order to facilitate creative work, and some other solution needs to be found. The final choice of language base for this aspect of Nutation, however, has been postponed until the NeXT computer's release 1.0 is available (due Fall of 1989).

Conclusions

Keith Hamel, in a discussion of his model of notational syntax, points out that "The limitations encountered in most music-printing software inevitably result from restrictions imposed by the syntax inherent in the program design" (Hamel, 1989). Nutation, however, by making both image and behavior a matter of data, imposes no syntactic restrictions whatever beyond that of a tree graph, while allowing the creation of arbitrarily deep hierarchical data structures without sacrificing the full graphic potential available on the NeXT computer.

Acknowledgements

The author gratefully acknowledges the Social Sciences and Humanities Research Council of Canada for their support of his research at Stanford University.

References

Byrd, Donald. 1986. "User Interfaces in Music-Notation Systems". In *Proceedings of the 1986 International Computer Music Conference*. San Francisco: Computer Music Association, pp.145-150.

Hamel, Keith A. 1989. "A Design for Music Editing and Printing Software Based On Notational Syntax". *Perspectives of New Music*, 27(1), 70-83.

Diener, Glendon. 1989. "TTrees: A Tool for the Compositional Environment". *Computer Music Journal* 13(2): 77-85.

Jaffe, David and Lee Boynton. 1989. "An Overview of the Sound and Music Kits for the NeXT Computer". *Computer Music Journal* 13(2): 48-55.