CyberBand: A "Hands-On" Music Composition Program

James Wright, Daniel V. Oppenheim, David Jameson
Don Pazel, Robert M. Fuhrer
{ jwright, music, dhj, pazel, rfuhrer, @watson.ibm.com }

Computer Music Center
T. J. Watson Research Center, IBM
P.O. Box 218, Yorktown Heights, NY 10598
www.research.ibm.com/music

Abstract

CyberBand provides a novel and powerful approach to composing music with computers. It is based on the concept that many kinds of music can be constructed by assembling, modifying, and transforming musical fragments through direct visual manipulation. We describe our approach to music representation, the user interface, and choice of user metaphors, while emphasizing the elements that enable non musicians to creatively compose music. We outline issues that influence our ongoing work in determining both usability and look and feel. A working prototype of CyberBand will be demonstrated.

1. Introduction

Music is abstract and complex. Composing instrumental music requires skills in various areas of music theory such as harmony, counterpoint, musical form, and orchestration. The user interface in systems for computer assisted composition often adopts metaphors that are based on music theory. Only a skilled musician can understand these metaphors and use the provided compositional tools effectively. Furthermore, systems for composing music tend to become large and complex, are hard to use, and have a significant learning curve. (For a survey of systems see Loy 1989; for examples of academic systems see Mathews 1969, Schottstaedt 1984, Puckette 1988/1991, Ames 1990, Taube 1991 and Oppenheim 1996; for examples of commercial systems see *Vision* 1995, *FreeStyle* 1995, and *Symbolic Composer* 1995).

We would like to bring the joy of creating and performing music to a wide range of users that may have little or no formal musical training. Our premise is that, while acquiring true musical competence requires extensive formal training, the musical experience itself, much like spoken language, is learned through a natural process that is a part of our social and cultural environment. We seek to empower users to create music through software that provides the basic musical competence needed to realize their musical ideas, without unduly restricting them. Users should be able to take charge of the compositional process and assume responsibility for making important compositional decisions.

Several commercial applications for the non professional musician have appeared in recent years. Our own *KidRiffs* is geared towards a young age group (ages 5-10). It provides a creative musical experience, but focuses primarily on performance rather than composition. Microsoft has recently released *Music Producer*. Here, composition is a black box process whereby a user specifics a style with some parameters, after which the application generates the music. In this system the user is passive and has little involvement in the creative process after initial decisions are made.

2. CyberBand

CyberBand is centered around the concept that musical riffs (thematic or idiomatic musical fragments) can be combined and modified in ways that enable the composition of music both by people with musical training and skills, and by those with no musical experience whatever. Composition on this level focuses on arrangements, combinations and interactions of relatively opaque elements. This allows users to create and refine a broad range of musical effects very quickly. When finer control is desired, individual elements can be "unfolded" to specify modifications precisely or perform detailed content editing. Because of this multi-level approach, CyberBand is surprisingly easy to learn and yields results quickly. As users start working at deeper levels, their options expand dramatically.

CyberBand has a base set of riffs that people can use individually or in combination to create music. (Users can also create, import and edit riffs when so desired.) The application provides great flexibility in manipulating the riffs to meet individual tastes and musical contexts, while taking care of many support functions needed for music composition and

Reprinted from ICMC 1997

performance. These features can be used as automatic facilitators for those without musical experience, as well as sophisticated assistants by those with musical training. This permits individuals to focus on creating music, rather than being concerned about, or inhibited by, the mechanics of music production.

3. The Music Representation

The music representation developed for the CyberBand prototype evolved from earlier work in DMIX (Oppenheim 1989, 1996), research into musical expression and the LeNNY framework (Oppenheim 1992), experience with commercial music sequencer software (Wright et al 1985) and mix automation software (Wright, Rayna 1991, 1993), and Nuance, a time-based media representation currently under development (Oppenheim and Wright, 1996). The Nuance model is significantly different from the model used in the CyberBand Smalltalk prototype (described below), and we are investigating the feasibility of using Nuance for the C++ version now in progress.

3.1 The Event structure

Music is typically modeled as a collection of note events. Each note has a set of attributes, such as pitch, onset, amplitude, and duration. Composition is carried out by specifying each note and its attributes. Notes may be further grouped into hierarchical structures that typically model instrumental parts or musical sections. We refer to this as the *Event Structure*.

Dealing with music on the level of individual notes and their attributes requires considerable musical skill and is also quite tedious. We therefore introduce in CyberBand a higher-level concept that we call a *Music Block*, or simply Block. A Block holds an ordered set of notes or percussive events, often corresponding to a musical phrase, motif or "riff" (though Blocks may also contain entire sections or compositions). The user creates music by selecting Blocks from a palette and placing them on a canvas we call a *Score Sheet*. An important feature of Blocks is that they can be manipulated as a discrete entity. For example, to set the duration of a Block, the user can drag the Block's bottom right corner to the desired length.

3.2 The modification structure

The shaping of musical gestures and expressive nuance is commonly achieved by setting new attribute values in all notes included in the desired region. Determining the new values usually requires a highly skilled musician. This can be a lengthy process that involves several iterations of trial and error, and can be tedious even with the aid of a graphical interface. Moreover, a successful change often requires additional modification after some other musical aspect has been adjusted. For example, if a user first sets an amplitude curve and then changes articulation from *legato* to *staccato*, the original amplitude curve may have to be readjusted.

In CyberBand we take a very different approach to the problem of editing and refining the music. Rather than changing the values of individual events directly, we introduce high level objects that model the desired changes, called *Modifiers*. For example, to set the amplitude curve of a Music Block, one would attach to it a Volume Modifier with the desired curve. Modifiers can be stacked together in any number and in any order (changing the order may change the effect). Attribute values for a given note are calculated just before it is scheduled to play, taking into account the effect of all associated Modifiers.

Our approach also differs from traditional computer models for music representation. Rather than have a single Event Structure of notes and attribute values, we introduce a second structure of Modifiers that we call the *Modification Structure*. Each Modifier is linked to one or more Music Blocks (or individual notes) in the Event structure. This double structure maps well onto instrumental music where the Event Structure models the music score and the Modification Structure models the musical changes that take place as the score is performed.

There are several distinct advantages to having a Modification Structure. First, it is easy to describe a desired musical result. For example, drawing an amplitude curve and attaching that to a single Music Block is easier than changing attribute values of many individual notes. Moreover, this operation can easily be carried out by non musicians. Second, it is easy to change a result after it is applied. In the example above one need only edit the amplitude curve in order to modify the audible result. Finally, it is easy to understand and manage interdependencies between different Modifiers. A Modifier can be muted to disable its effect, the order of modifications can be adjusted, and modifiers that are not longer needed can be removed.

4. The CyberBand User Interface

The main components of the CyberBand user interface comprise a Score Sheet, Music Blocks, Modifiers, and Catalogs. Catalogs are used to store and access persistent content and are not further described in this paper. We are experimenting with several user metaphors based on these components, namely the paint and word processor metaphors.

4.1 The Score Sheet

The Score Sheet from our prototype version can be seen in Figure 1. This is the main area in which composition takes place. The horizontal axis represents time. The vertical axis has no special significance in the initial prototype, but we plan to use it to distinguish regions with specialized roles.

Composition is done by selecting Blocks from a Catalog, and placing them on the score sheet. Blocks can be moved on the score sheet and their musical attributes can be manipulated graphically.

For example, the duration of a block can be set by dragging its right side, the volume can be set by dragging its upper side, fade-in and fade-out can be set by moving the top left and top right corners respectively, and the block can be transposed by dragging a special transposition handle. Other attributes of the block, such as its instrument, harmony, scale, and pan position, can be set by using an inspector or other tool. Advanced users can also gain access to the underlying musical content of a given block, where they can edit individual notes through a graphical "piano roll" editor or other editor.

The score sheet provides the user with a high level visual representation of the music. As such, it is a powerful tool that allows users to shape and structure their work, providing both audio and visual feedback. Unlike conventional scores, individual notes are not displayed.

We believe that that non musicians will benefit from a system that focuses on higher-level musical characteristics. For them a primary emphasis on low-level detail may obscure the underlying compositional intent. Most users, most of the time, will benefit from a system that focuses on higher-level musical characteristics. We expect that even professional musicians will find this representation extremely useful and will not, for many activities, require a more detailed representation (although an event-level piano roll view is available). In fact, such a visualization, with some refinements, has been suggested for advanced users (Oppenheim 1987).

4.2 Modifiers

Modifiers are one of the most powerful concepts of CyberBand, as they allow a user to control almost any aspect of the music easily and precisely. Modifiers are represented as independent visual objects that can be directly manipulated to change their behavior, and which are attached to the content elements they modify. Modifiers can have a very subtle effect, typically used to provide expressive nuance, or a very dramatic effect, often used to completely change the way a Riff is rendered. Modifiers also provide tools that non musicians can use to create new source materials via transformation.

There are many kinds of Modifiers. For example, Rhythm Grids set the rhythms of the blocks to which they are attached. Beat Emphasis Modifiers create

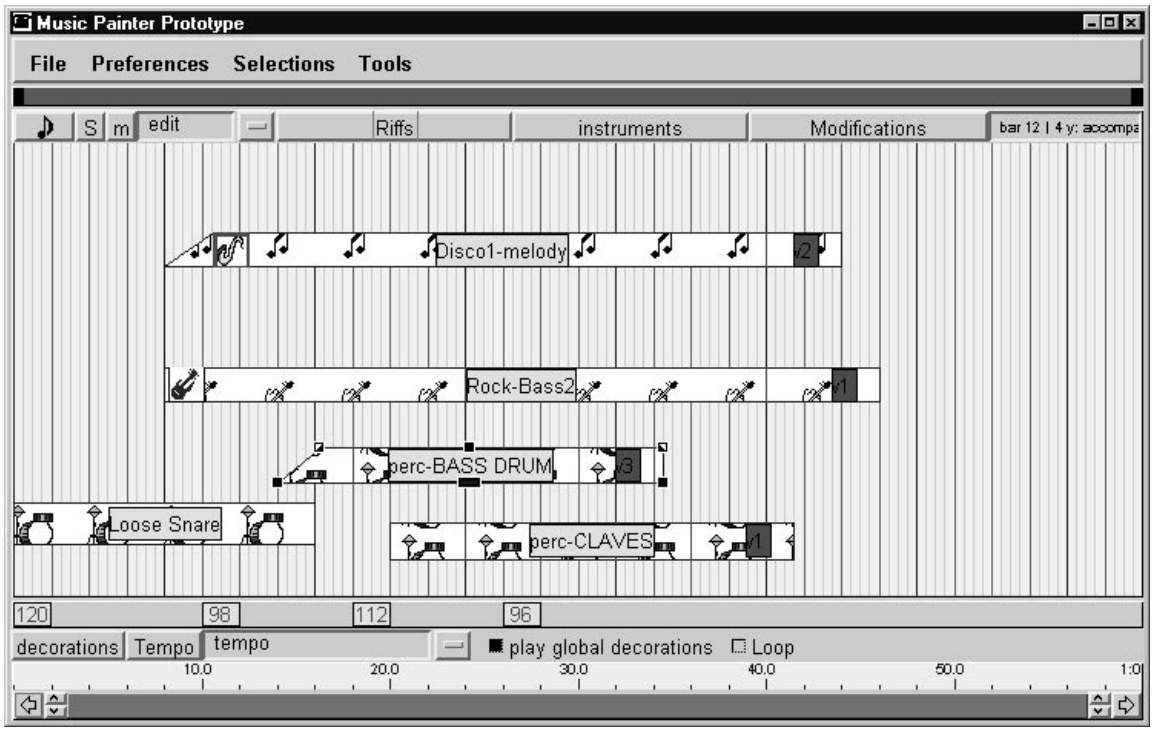

Figure. 1 An early prototype of the CyberBand Score Sheet.

the feel of a live performer. Scale modifiers change the scale in which music is playing. In addition there are higher level Modifiers, such as interpolators which define how several subordinate Modifiers affect the music over time, and Morphing Modifiers which blend musical characteristics from different music Riffs (Oppenheim 1995).

Modifiers are particularly powerful in that they can be applied to any hierarchical level within a composition: individual blocks, a selection containing several blocks, a complete musical part (such as all the drum blocks in a composition), or the entire score sheet. In each case, a given Modifier will affect all the music within its scope. Because a Block can be affected by a number of Modifiers associated with different levels of the composition, the corresponding modification structure can become very complex.

5. Conclusion

A working prototype of CyberBand was developed in Smalltalk by one of the authors (Oppenheim) in 1996. In January 1997 we begun work on a C++ version that will be released to a wider audience.

The utility and effectiveness of this approach depends on the user's ability to have fine-grained control over musical detail when needed. Without such control, the program would be little more than a scrapbook for music clips. The basic solution is twofold. First, users can work at finer levels when desired, by opening a block or decoration and modifying its contents or behavior. Second, as noted earlier, mechanisms are provided for creating hierarchies of modifiers that can alter the way the music is rendered (performed) in subtle or extreme ways.

Since much of the construction and refinement of the final composition is done at a high level, without having to deal with individual notes, several goals are accomplished. Users with no formal education are able to create music with a high degree of compositional freedom. On the other hand, accomplished computer musicians have a set of high level tools that support fluid working sessions, provide much improved control over the fine detail, and let them easily accomplish tasks which are quite difficult using standard user interfaces and environments.

6. References

"*FreeStyle*" (1995). Software for Windows and Macintosh, published by *Mark of the Unicorn*, Cambridge, Massachusetts

"*Symbolic Composer, 6th Generation Common Music Language Reference*" (1995). Software for the Macintosh, published by *Tonality Systems*.

"*Vision*" (1996). Software for Windows and Macintosh, published by *Opcode Software Systems*, Menlo Park, California.

Ames, C. (1990). "Introduction to COMPOSE: An editor and interpreter for automated score generation and score processing," Interface: Journal of New Music Research, 20:3-4, p. 181.

Loy, G. (1989). "Composing with Computers—a Survey," in *Current Directions in Computer Music Research*, Mathews M. and Pierce J., editors, MIT Press, Cambridge, Massachusetts.

Mathews, M. V. (1969). "The Technology of Computer Music," MIT Press, Cambridge, MA

Oppenheim, D. (1987). "The P-G-G Environment for Music Composition." *Proceedings of the International Computer Music Conference (ICMC)*, Illinois.

Oppenheim, D. (1989). "DMIX: An Environment for Composition." *Proceedings of the ICMC*, Ohio.

Oppenheim, D. (1992). "Compositional Tools for Adding Expression to Music." *Proceedings of the ICMC*, San Jose, California.

Oppenheim, D. (1993) "Slappability: A New Metaphor for Human Computer Interaction." in: *Music Education: An Artificial Intelligence Perspective*, Springer Verlag, London.

Oppenheim, D. (1995). "Demonstrating Mmorph: A System for Morphing Music in Real-Time." *Proceedings of the ICMC*, Banff, Canada.

Oppenheim, D. (1996) "DMIX—A Multi Faceted Environment for Composing and Performing Computer." Computers and Mathematics with Applications, 32:1, pp. 117-135, 1996.

Oppenheim, D., Wright, J. (1996) "Towards a Framework for Handling Musical Expression." *Proceedings of the ICMC*, Hong Kong.

Puckette, M. (1988). "The Patcher," *Proceedings of the ICMC*, Cologne.

Puckette, M. (1991). "Combining Event and Signal Processing in the MAX Graphical Programming Environment", Computer Music Journal 15/3, MIT Press, Cambridge, Massachusetts.

Schottstaedt, B. (1984). "PLA - A Tutorial and Reference Manual," CCRMA report No. STAN-M-24, Dept. of Music, Stanford University, CA.

Taube H. (1991) "Common Music: A Music Composition Language in Common-Lisp and Clos," CMJ 15/2. MIT Press, Cambridge, MA

Wright, Frazier and Steele, (1985), *Sequencer Plus* music sequencer software for PC, published by *Voyetra Technologies*, Yonkers, New York.

Wright, J., Rayna, D., (1991). *Diskmix III V4* Mix Automation software, published by *Otari Console Products Group*, Hauppauge, NY.

Wright, J., Rayna, D., (1993). Concept I Automated Audio Mixing Console software, ibid.