Composing Interactive Music

Techniques and Ideas Using Max

Todd Winkler

Composing Interactive Music

Techniques and Ideas Using Max

Todd Winkler

The MIT Press
Cambridge, Massachusetts
London, England

First MIT Press paperback edition, 2001

This book was set in Stone Serif and Stone Sans by Graphic Composition, Inc. and was printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Winkler, Todd, 1958–
Composing interactive music : techniques and ideas using Max /
Todd Winkler.
p. cm.
Includes bibliographical references and index.
ISBN 0-262-23193-X (hc : alk. paper), 0-262-73139-8 (pb)
1. Computer composition. 2. Max (Computer program language)
I. Title.
MT56.W5 1998
781.3'45268—dc21 97-34535
CIP
MN

10 9 8 7 6 5

Preface

Composing Interactive Music: Techniques and Ideas Using Max, is a book about the technical and aesthetic possibilities of interactive music: a music composition or improvisation where software interprets a live performance to affect music generated or modified by computers. It describes musical concepts and programming techniques, and presents some insights into the development and research efforts leading up to this increasingly popular area of computer music.

The initial research for this book was conducted for my dissertation at Stanford University, *Three Interactive Etudes for Clarinet and Computer: Technology and Aesthetics in Interactive Composition* (1992). The dissertation, supervised by John Chowning, included a description of a Max program called *FollowPlay,* a large collection of software modules designed as a general-purpose package for interactive composition. Much of *FollowPlay* was conceived of and created while I was a visiting researcher at Institute de Recherche et Coordination Acoustique/Musique (IRCAM) during 1990–1991, with guidance from Miller Puckette, the original author of Max, and Cort Lippe and Zack Settel, who also contributed to the creation of Max.

The numerous compositions I have created with *FollowPlay* have been the laboratory for ongoing research and development for this book. Most of the programming examples presented here were taken directly from these works. However, the *FollowPlay* program is not presented in its entirety since it was never meant to be a program for public distribution. Instead, the principles and techniques I encountered while designing and redesigning the software are described here,

with the idea that readers will create their own software to reflect their personal approach to composition.

How to Use This Book

Max is a graphical programming language, designed by Miller Puckette, for interactive composition. A version of Max, further developed by David Zicarelli, is commercially available from Opcode Systems, Inc. The techniques and ideas presented in this book are demonstrated through programming examples using Max. The examples are described in the text, accompanied by a picture of how they appear on the computer screen. The same examples are included as software on the accompanying CD-ROM, playable on a Macintosh computer, which may be copied and edited for further study. Whenever possible, examples and discussions will show the musical relevance of programming techniques, giving the reader firsthand experience in making and understanding interactive music. In the few software examples that are not my own, I have given credit to the contributor, whose name appears above their work. Because of space limitations, some of the examples on CD-ROM are not printed in their entirety in the book. An index of all examples, cross-referenced with the Opcode's Max manual, appears in appendix A.

Although this book is targeted at composers who will be writing music and software using Max, it has been written so that a casual reader might learn the basic concepts of interactive composition by just reading the text, without running any software at all. For readers who have access to a MIDI setup who do not own Opcode's Max programming environment, the accompanying software will run by itself, in a non-editable form, on most Macintosh computers. However, the complete Max application is highly recommended to accompany the text since it is necessary for editing the examples and for creating new software. Programming interactive works using Max is like learning to play an instrument: a thorough understanding will come only with firsthand experience. There is no substitute for practice.

The examples for this book require a basic MIDI studio, consisting of a Macintosh computer with at least 4 MB of RAM, a MIDI keyboard, and a MIDI interface. Keyboards should send and receive on MIDI channel 1. Although the examples are designed for keyboards, other

MIDI controllers may be used in conjunction with a MIDI sound module.

While no knowledge of Max programming is assumed, it would be helpful for readers to familiarize themselves with the Max tutorial and the manual before or while reading the text. (These are provided by Opcode Systems, Inc.) The information on Max and many of the examples presented here are indebted to these excellent resources. Although some overlap between the Max tutorial and the first chapters of this book is unavoidable, presenting two views of similar information will provide a solid foundation for more advanced work. Experienced Max programmers may want to skip chapters 3 and 4, which provide an introduction to Max programming.

The structure of the book follows a logical progression, keeping in mind composers with few programming skills. The ten chapters were conceived of as four large sections: Introduction, History, and Theory; Programming Foundation; Core Components; and Advanced Techniques. An overview of the chapters is as follows:

I Introduction, History, and Theory

Chapter 1, *Introduction to Interactive Composition,* presents basic concepts regarding interactive music and continues with a brief history of interactive music compositions and systems.

Chapter 2, *Interactive Music Performance and Theory,* discusses models for interactions, issues of freedom and control, and human/computer relationships.

II Programming Foundation

Chapter 3, *Graphic Programming using Max,* is designed to build a foundation for writing software using Max. It contains essential information for anyone new to Max and new to programming. It begins with an overview of various programming languages and programming concepts as they relate to Max, continues with an introduction to the basic materials and operations used in Max, and concludes with a section on MIDI.

Chapter 4, *Program Structure and Design,* is a continuation of chapter 3, and delves into more general programming principles. The approach used in chapter 4 (and throughout the book in general) is designed to

demonstrate fundamental computer science concepts through musical examples, so that musicians new to programming will learn directly by creating music rather than by writing programs unrelated to their interests.

Chapters 3 and 4 establish a baseline of understanding with which to proceed to more advanced topics, and equips composers with the minimal understanding of computer concepts needed to produce interactive work.

Chapter 5, *Interface Design,* covers principles of interface design and reviews Max's interface objects.

III Core Components

Chapter 6, *Listener Objects,* shows techniques for analyzing and storing performance information.

Chapter 7, *Composer Objects,* introduces methods for generating and processing computer music.

IV Advanced Techniques and Concepts

Chapter 8, *Sound Design,* explores the use of sound through synthesis, orchestration, mixing, and computer-controlled signal processing.

Chapter 9, *Performance Structures,* offers suggestions for compositional strategies to create larger works, and contains descriptions of score objects that can be used for coordinating events during a performance.

Chapter 10, *Multimedia Extensions and New Controllers,* focuses on multimedia applications of interactive composition, Max's QuickTime extensions, and new controllers.

Appendix A, contains an index of the examples used in the book, cross referenced with related topics in the Max manual. This provides a single source for all information regarding software examples. Also useful will be Max's on-line help files, a software feature that calls up a description and working example of an object when the option key is held down while any object is clicked.

Acknowledgements

I am deeply grateful to Miller Puckette, David Zicarelli, Cort Lippe, Alan Strange, Dale Stammen, and Jason Vantomme for their thorough feed-

back at various stages of this book. I would also like to thank the following reviewers: Michael Pelz-Sherman, John P. Lamar, Burton Beerman, and Julian Richards. Dale Stammen and Michael Pelz-Sherman helped to test and edit the examples for the book. Special thanks to my student research assistants, Alex Gottschalk, Jason Duva, Brian Lee, and Scott Pagano.

Funding for this project was provided by the Center for Computer Research in Music and Acoustics (CCRMA), a Stanford University Paris Scholarship, an Oberlin College Faculty Research Grant, and Brown University. Special thanks goes to Tim Self at Opcode for software support.

The single source that proved the most useful to me was Robert Rowe's *Interactive Computer Music Systems* (MIT Press). It summarizes much of the research related to the development of interactive music, and explains Rowe's *Cypher* program. In addition, Rowe lays out a conceptual framework that was the starting point for this text. It would be an ideal reference for readers seeking a more in-depth study of various types of interactive music systems.

Finally, I must acknowledge the help of my wife, Karina Lutz, whose unflagging support was not just moral, but tangible as the first editor of this book.

Contents

I Introduction, History, and Theory

1 Introduction and Background

Interaction is a two-way street. Nothing is more interactive than a good conversation: two people sharing words and thoughts, both parties engaged. Ideas seem to fly. One thought spontaneously affects the next. Participants in conversation assume much past experience and find excitement in shared experience. Conversations stay within a consistent context that creates a feeling of mutual understanding without being predictable. On the other hand, when only one person does the talking it isn't interactive—it is a lecture, a soliloquy.

Computers simulate interaction. Computers continue to move into the home and workplace, not because they are capable of millions of calculations per second but because they are clever mimics, able to represent images, sounds, and actions from the real world and imagined worlds. Computers simulate interaction in this constructed world by allowing users to change aspects of their current state and behavior. This interactive loop is completed when the computers, in turn, affect the further actions of the users.

Interaction means action. Computer programs are more or less interactive, depending on how they respond to human actions and how they engage human response. Interactivity comes from a feeling of participation, where the range of possible actions is known or intuited, and the results have significant and obvious effects, yet there is enough mystery maintained to spark curiosity and exploration. Television is not very interactive (yet). The viewer does not have any way to change aspects of a show, except for switching channels, changing the volume, or altering the controls for color, tint, and horizontal hold. The medium will become more interactive when viewer actions have a meaningful impact on the content and structure of the work.

This book describes techniques for creating interactive computer music (hereafter referred to simply as interactive music). Interactive music is defined here as a music composition or improvisation where software interprets a live performance to affect music generated or modified by computers. Usually this involves a performer playing an instrument while a computer creates music that is in some way shaped by the performance. This is a broad definition that encompasses a wide range of techniques, from simple triggers of predetermined musical material, to highly interactive improvisational systems that change their behavior from one performance to the next. Interactive music may also have applications in commercial multimedia software and CD-ROM titles, such as educational or entertainment titles, where the user (the "performer") controls aspects of music selection and compositional processes using the computer keyboard and mouse.

Performers participate in the creation of an interactive work, in part, by the amount of freedom they have to produce significant results in the computer's response. For example, a fully notated, predetermined score could be made slightly interactive by allowing a performer to control a single parameter of the computer-generated music, such as tempo. In more sophisticated interactive pieces, performers control many significant musical parameters and the composition can change dramatically according to their interpretation. In the most extreme examples, a performer is free to play any kind of music, and the computer has enough "intelligence" to respond in a way that makes sense and naturally encourages the performer's continuation. Like good conversations, interactive compositions succeed by encouraging spontaneity while residing within the boundaries of a dynamic context that seems whole and engaging.

Interactive music is a natural extension of a long history of collaborations. Music has always been an interactive art in which musicians respond to each other as they play, whether it is a conductor with an orchestra, a lead singer with a rock band, or the members of a jazz combo or a string quartet. Performers listen to each other while they play, continuously altering and shaping their performance according to what they hear. Many traditional musical relationships can be simulated with the computer. These models can be valuable starting points for an interactive work. More importantly, interactive techniques may suggest a new musical genre, one where the computer's capabilities are

used to create new musical relationships that may exist only between humans and computers in a digital world.

The adjective "virtual" is a current buzzword that describes computer simulations of things that behave like real-world objects, situations, and phenomena. Describing such a simulation, Brenda Laurel writes, "A virtual world may not look anything like the one we know, but the persuasiveness of its representation allows us to respond to it *as if it* were real." (1993) Interactive music techniques can be used to model many of the key elements in making and listening to music: instruments, performers, composers, and listeners. *Virtual instruments* behave somewhat like real instruments, and can be played from the computer keyboard, a MIDI controller, or as an extension of a traditional instrument. *Virtual performers* may play with human performers, interacting with "musical intelligence" in a duet, combo, or accompaniment role. *Virtual composers* create original music based on flexible and sometimes unpredictable processes specified by a real composer. These processes might represent the same musical ideas that a composer may choose in order to create a piece for acoustic instruments. Other processes may be specifically designed to take advantage of the capabilities of the computer. Finally, a *virtual listener* or *virtual critic* may pass judgment by reacting to and altering the final outcome of a performance (Rowe 1993). Such a critic might be designed to analyze the accuracy of a performance, or steer the output of a composition away from too much repetition.

The behavior of these virtual entities must be described by the programmer, who must answer the questions: How are virtual instruments played? How does a virtual performer respond to musical input? How do virtual composers generate, process, and structure musical material? What are the criteria for a virtual critic to judge the success or failure of a performance or or a piece of music? The opinions of the programmer are impossible to separate from the final outcome. There is a compelling reason why composers and other artists need to become involved in the creation of software, and why programs like Max are needed to make software creation more accessible to artists. Among other things, artists are experts at creating vivid imaginary worlds that engage the mind and the senses. These talents are especially needed in creating rich and engaging computer environments. Computers are not intelligent. They derive their appearance of intelligence only from

the knowledge and experience of the person who creates the software they run.

Components of an Interactive System

Broadly speaking, interactive music works by having a computer interpret a performer's actions in order to alter musical parameters, such as tempo, rhythm, or orchestration. These parameters are controlled by computer music processes immediately responsive to musical data (such as individual notes or dynamics) or gestural information (such as key pressure, foot pedals, or computer mouse movements). Interactive software simulates intelligent behavior by modeling human hearing, understanding, and response (Rowe 1993). The response must be believable in the sense that it seems appropriate for the action taken, and appropriate for the style of music. This process is somewhat analogous to the distinct activities that take place during a jazz improvisation or other musical dialogue: listening, interpreting, composing, and performing. Figure 1.1 describes five steps to creating an interactive piece:

1. Human input, instruments—Human activity is translated into digital information and sent to the computer.

2. Computer listening, performance analysis—The computer receives the human input and analyzes the performance information for timing, pitch, dynamics, or other musical characteristics.

3. Interpretation—The software interprets the computer listener information, generating data that will influence the composition.

4. Computer composition—Computer processes, responsible for all aspects of the computer generated music, are based on the results of the computer's interpretation of the performance.

5. Sound generation and output, performance—The computer plays the music, using sounds created internally, or by sending musical information to devices that generate sound.

The first two steps are very practical, and limited, and deal mainly with facts. The software should be accurate in its analysis and, therefore, must understand certain things about human performance. The last three steps are artistic decisions limited only by a composer's skill and imagination, since there are countless ways that performance information can be interpreted to create original music and sound.