DIGITIZING NORTH INDIAN MUSIC:

Preservation and Extension using Multimodal Sensor Systems, Machine Learning and Robotics

by

Ajay Kapur
B.S.E., Princeton University, 2002

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

In Interdisciplinary Studies involving Departments of Computer Science, Music, Electrical and Computer Engineering, Mechanical Engineering, & Psychology

University of Victoria

DIGITIZING NORTH INDIAN MUSIC:

Preservation and Extension using Multimodal Sensor Systems, Machine Learning and Robotics

by

Ajay Kapur
B.S.E., Princeton University, 2002

Supervisory Committee

Dr. G. Tzanetakis (Department of Computer Science, Electrical Engineering & Music)
Supervisor

Dr. P. R. Cook (Princeton University Department of Computer Science & Music)
Co- Supervisor

Dr. W. A. Schloss (School of Music & Department of Computer Science)
Co- Supervisor

Dr. P. F. Driessen (Department of Electrical and Computer Engineering & Music)
Co- Supervisor

Dr. A. Suleman (Department of Mechanical Engineering)
Outside Member

Dr. N. Virji-Babul (Department of Psychology)
Outside Member

Supervisory Committee

Dr. G. Tzanetakis (Department of Computer Science, Electrical Engineering & Music)
Supervisor

Dr. P. R. Cook (Princeton University Department of Computer Science & Music)
Co- Supervisor

Dr. W. A. Schloss (School of Music & Department of Computer Science)
Co- Supervisor

Dr. P. F. Driessen (Department of Electrical and Computer Engineering & Music)
Co- Supervisor

Dr. A. Suleman (Department of Mechanical Engineering)
Outside Member

Dr. N. Virji-Babul (Department of Psychology)
Outside Member

ABSTRACT

This dissertation describes how state of the art computer music technology can be used to digitize, analyze, preserve and extend North Indian classical music performance. Custom built controllers, influenced by the Human Computer Interaction (HCI) community, serve as new interfaces to gather musical gestures from a performing artist. Designs on how to modify a Tabla, Dholak, and Sitar with sensors and electronics are described. Experiments using wearable sensors to capture ancillary gestures of a human performer are also included. A twelve-armed solenoid-based robotic drummer was built to perform on a variety of traditional percussion instruments from around India. The dissertation also describes experimentation on interfacing a human sitar performer with the robotic drummer. Experiments include automatic tempo tracking and accompaniment methods. A framework is described for digitally transcribing performances of masters using custom designed hardware and software to aid in preservation. This work draws on knowledge from many disciplines including: music, computer science, electrical engineering, mechanical engineering and psychology. The goal is to set a paradigm on how to use technology to aid in the preservation of traditional art and culture.

Table of Contents

SUPERVISORY COMMITTEE II
ABSTRACT III
TABLE OF CONTENTS IV
LIST OF FIGURES VII
ACKNOWLEDGEMENTS XIV

1 INTRODUCTION 1
1.1 Motivation *2*
1.2 Overview *3*
1.3 Key Contributions *7*

RELATED WORK 9

2 A HISTORY OF MUSICAL GESTURE EXTRACTION 10
2.1 Keyboard Controllers *11*
2.2 Drum Controllers *12*
2.3 String Controllers *15*
2.4 Wind Controllers *16*
2.5 Body Controllers *18*
2.6 Summary *19*
3 A HISTORY OF MUSICAL ROBOTICS 20
3.1 Piano Robots *22*
3.2 Turntable Robots *24*
3.3 Percussion Robots *25*
3.4 String Robots *30*
3.5 Wind Robots *33*
3.6 Summary *34*
4 A HISTORY OF MACHINE MUSICIANSHIP 37
4.1 Algorithmic Analysis *38*
4.2 Retrieval-Based Algorithms *39*
4.3 Stage Ready Systems *42*
4.4 Summary *42*

MUSICAL GESTURE EXTRACTION 44

5 THE ELECTRONIC TABLA 45
5.1 Evolution of the Tabla with Technology *46*
5.2 Tabla Strokes *48*
5.3 The MIDI Tabla Controller *50*
5.4 Sound Simulation *54*
5.5 Graphic Feedback *57*
5.6 User Study of the ETabla Sensors *59*
5.7 Summary *61*
6 THE ELECTRONIC DHOLAK 63
6.1 Background *64*
6.2 GIGAPOPR: Networked Media Performance Framework *67*
6.3 The Electronic Dholak Controller *73*
6.4 veldt: Networked Visual Feedback Software *76*
6.5 Summary *77*

7 THE ELECTRONIC SITAR 82
7.1 Evolution of the Sitar 83
7.2 Traditional Sitar Technique 84
7.3 The MIDI Sitar Controllers 86
7.4 Graphic Feedback 91
7.5 Summary 93
8 WEARABLE SENSORS 94
8.1 Motion Capture for Musical Analysis 95
8.2 The KiOm Wearable Sensor 101
8.3 The WISP Wearable Sensors 105
8.4 Summary 107

MUSICAL ROBOTICS 109

9 THE *MAHADEVIBOT* 110
9.1 Design 112
9.2 Experimental Evaluation 118
9.3 Summary 119

MACHINE MUSICIANSHIP 122

10 TEMPO TRACKING EXPERIMENTS 123
10.1 Method 125
10.2 Experimental Results 127
10.3 Summary 129
11 RHYTHM ACCOMPANIMENT EXPERIMENTS 131
11.1 Applications 132
11.2 Method 134
11.3 Experimental Results 138
11.4 Summary 141
12 PITCH & TRANSCRIPTION EXPERIMENTS 142
12.1 Method 143
12.2 Sheet Music 146
12.3 Summary 146
13 "VIRTUAL-SENSOR" GESTURE EXTRACTION 147
13.1 Method 149
13.2 Experimental Results 153
13.3 Summary 155
14 AFFECTIVE COMPUTING EXPERIMENTS 156
14.1 Background 157
14.2 Method 158
14.3 Experimental Results 159
14.4 Summary 163

INTEGRATION AND CONCLUSIONS 165

15 INTEGRATION AND MUSIC PERFORMANCE 166
15.1 April 12, 2002 - ETabla in Live Performance 166
15.2 June 3, 2003 - The Gigapop Ritual 168
15.3 June 4th, 2004 – ESitar Live in Japan 171
15.4 November 18, 2004 - ESitar and Eight Robotic Turntables 174
15.5 April 18th, 2006 – ESitar with DeviBot 175
15.6 November 6th, 2006 – ESitar 2.0 with MahaDeviBot 177
15.7 February 5th, 2007 – Meeting with Trimpin 178
15.8 March 11th, 2007 – National University of Singapore Concert 180
15.9 June 9th, 2007 – ESitar and MahaDeviBot Live in New York City 182

16 CONCLUSIONS 184
16.1 Summary of Contributions 185
16.2 Discussions on Techniques 186
16.3 Challenges of Interdisciplinary Research 195
16.4 Future Work 196

APPENDIX 198

A AN INTRODUCTION TO NORTH INDIAN CLASSICAL MUSIC 199
A.1 Nad 200
A.2 The Drone 202
A.3 The Raga System 204
A.4 Theka 210
B PHYSICAL COMPUTING 214
B.1 Microcontrollers 214
B.2 Sensors 217
B.3 Actuators 219
B.4 Music Protocols 221
C MACHINE LEARNING 224
C.1 ZeroR Classifier 225
C.2 k-Nearest Neighbor 225
C.3 Decision Trees 226
C.4 Artificial Neural Networks 229
D FEATURE EXTRACTION 234
D.1 Audio-Based Feature Extraction 234
E COMPUTER MUSIC LANGUAGES 242
E.1 STK Toolkit 242
E.2 ChucK 243
E.3 Marsyas 243
E.4 Pure Data (pd) 243
E.5 Max/MSP 244
F PUBLICATIONS 245
F.1 Refereed Academic Publications 245
F.2 Publications by Chapter 250
F.3 Interdisciplinary Chart 250

BIBLIOGRAPHY 251

List of Figures

Figure 1 - Radio Baton/Drum used by Max Mathews, Andrew Schloss, and Richard Boulanger. 13
Figure 2 - D'CuCKOO 6 piece drum interfaces on stage (left). BeatBugs being performed in Toy Symphony in Glasgow, UK (right). 14
Figure 3 - Hypercello (Machover), SBass (Bahn), and Rbow (Trueman). 15
Figure 4 - Wind Hyperinstruments: (left) Trimpin's Saxophone, (middle) Cook/Morrill trumpet, (right) HIRN wind controller 17
Figure 5 - Body Controllers: (left to right) Tomie Hahn as PikaPika, Cook's Pico Glove, Cook's TapShoe, Paradiso's wireless sensor shoes. 18
Figure 6 - Trimpin's automatic piano instruments (a) contraption Instant Prepared Piano 71512[175] (b) piano adaptop that strikes keys automatically 23
Figure 7 - Trimpin's eight robotic turntables displayed in his studio in Seattle Washington. 24
Figure 8 - Williamson's "Cog" robot playing drums. [195] 26
Figure 9 - (a) Chico MacMurtie Amorphic Drummer[104], (b) N.A Baginsky's robotic rototom "Thelxiepeia"[8], (c) JBot's *Captured by Robots'* "Automation" [75] . 27
Figure 10 - Trimpin's robotic Idiophones.[174] 27
Figure 11 - LEMUR's TibetBot [159] 28
Figure 12 - Trimpin's "Conloninpurple" [174] 29
Figure 13 - (a) Gordon Monahan's "Multiple Machine Matrix" [115] (b) LEMUR's !rBot [159] 30
Figure 14 - (a) N.A Baginsky's "Aglaopheme" [8] (b) Sergi Jorda's Afasia Electric Guitar Robot [77] (c) LEMUR's Guitar Bot. [160] 31
Figure 15 - (a) Krautkontrol [174] (b) "If VI was IX" [174] at the Experience Music Project, Seattle, USA. 32
Figure 16 - (a) Makoto Kajitani's Mubot [80], (b) N.A. Baginsky's "Peisinoe" bowing bass [8] (c) Sergi Jorda's Afasia Violin Robot [77]. 33
Figure 17 - (a) Makoto Kajitani's Mubot [80], (b) Sergi Jorda's Afasia Pipes Robot [77] (c) Roger Dannenberg's "McBlare" robotic bagpipes. 34

Figure 18 - (a) Example of robotic percussive toy: friendly monkey playing cymbals. (b) Maywa Denki's "Tsukuba Series" [46], (c) "Enchanted Tiki Room" at Walt Disney World, Orlando, Florida, USA. 36
Figure 19 - (left) Mari Kamura and Eric Singer's GuitarBot; (right) Gil Wienberg's Haile 42
Figure 20 - North Indian Tabla. The *Bayan* is the silver drum on the left. The *Dahina* is the wooden drum on the right. 46
Figure 21 - The Mrindangam, a drum of the Pakhawaj family of instruments. 47
Figure 22 - Tabla pudi (drumhead) with three parts: Chat, Maidan and Syahi. 48
Figure 23 - Ga and Ka Strokes on the *Bayan* 49
Figure 24 - Na, Ta and Ti strokes on the *Dahina*. 50
Figure 25 - Tu, Tit, and Tira strokes on the *Dahina*. 50
Figure 26 - Electronic *Bayan* Sensor Layout. 52
Figure 27 - Circuit diagram of *Bayan* Controller. The *Dahina* Controller uses similar logic 52
Figure 28 - Electronic *Dahina* Sensor Layout. 53
Figure 29 - The Electronic Tabla Controller 54
Figure 30 - Banded Waveguide Schematic. 55
Figure 31 - Figures showing construction of paths that close onto themselves 56
Figure 32 - Sonograms comparing recorded (left) and simulated (right) Ga strike 57
Figure 33 - Different modes of visual feedback. 58
Figure 34 - User testing results of the *ETabla* of User Test A and B. The tests measured maximum strike rate for each sensor as evaluated by an expert performer… 60
Figure 35 - Flow Control and Sequencing of GIGAPOPR 69
Figure 36 - Traditional Dholak. 74
Figure 37 – Two-player *EDholak* with Piezo sensors, digital spoon, CBox and custom built MIDI Control Software. 75
Figure 38 – (Left) Layering text elements (Hindi) over several sparse structures using *veldt*. (Middle) Example screenshot of structure evolved from a drumming sequence generated by veldt. (Right) A more cohesive structure generated by a variation on the rule set. 77

Figure 39 - A traditional Sitar. 82
Figure 40 - (a) A stick zither vina [7], (b) A vina made of bamboo [157], (c) A sehtar [157], (d) A 7-stringed sitar [157]. 83
Figure 41 - Traditional Sitar Playing Technique. 86
Figure 42 – Atmel Controller Box Encasement of *ESitar* 1.0 (left, middle). PIC Controller Box Encasement on *ESitar* 2.0 (right). 87
Figure 43 - The network of resistors on the frets of the *ESitar* 1.0 (left, middle). The *ESitar* 2.0 full body view (right). 89
Figure 44 - FSR sensor used to measure thumb pressure on *ESitar* 1.0 (left) and *ESitar* 2.0 (right). 90
Figure 45 - Gesture capturing sensors at base of *ESitar* 1.0. 90
Figure 46 - Headset with accelerometer chip. 91
Figure 47 – (left) Roll of swara rendered over video stream. (right) Animation scrubbing from thumb pressure. 92
Figure 48 – (left) Screenshot of data capturing process for tabla performance. (right) Screenshot of data capturing process for violin performance. 97
Figure 49 - Vicon Sonification Framework. 98
Figure 50 - The following code sketch shows has the 3 markers of the x,y,z wrist Position can be used to control 3 sinusoidal oscillators in Marsyas. 98
Figure 51 - Example template ChucK code for sonification of body motion data. 99
Figure 52 - The *KiOm* Circuit Boards and Encasement. 102
Figure 53 - Diagram of synchronized audio and gesture capture system. 102
Figure 54 – Wearable sensors used for a drum set performance. 104
Figure 55 - Wearable sensor used for a Tabla performance (left). Set up for using on head of performer (right). 105
Figure 56 - Wearable sensor used to capture scratching gesture of turntablist (left). Wearable sensor used in conjunction with live Computer Music Performance using ChucK (right).. 105
Figure 57 - Wireless Inertial Sensor Package (*WISP*) 106
Figure 58 - Comparison of Acquisition Methods 108
Figure 59 - Kapur Finger using a grommet and padding. 113

Figure 60 - Singer Hammer with added ball-joint striking mechanism..... 114
Figure 61 - Trimpin Hammer modified to fit the *MahaDeviBot* schematic. 115
Figure 62 - Trimpin Hammers use on *MahaDeviBot*. 115
Figure 63 - Trimpin BellHop outside shell tubing (left) and inside extended tubing (right). 116
Figure 64 - Trimpin BellHops used on *MahaDeviBot*. 116
Figure 65 - The bouncing head of *MahaDeviBot*. 117
Figure 66 - Maximum speeds attainable by each robotic device. 118
Figure 67 - Dynamic Range Testing Results. 119
Figure 68 - Evolution of *MahaDeviBot* from wooden frames to the sleek 12-armed percussion playing device. 120
Figure 69 - Multimodal Sensors for Sitar Performance Perception. 125
Figure 70 - Comparison of Acquisition Methods. 127
Figure 71 - Normalized Histogram of Tempo Estimation of Audio (left) and Fused Audio and Thumb (right) 128
Figure 72 - Kalman Tempo Tracking with decreasing onset periods. 129
Figure 73 - Symbolic MIR-based approach showing how *ESitar* sensors are used as queries to multiple robotic drum rhythm databases. 133
Figure 74 - Audio Driven Retrieval Approach. 134
Figure 75 - Wavelet Front End for Drum Sound Detection..... 136
Figure 76 - Audio Signal “Boom-Chick” Decomposition. 137
Figure 77 - (left) Transcribed Drum Loop (Bass and Snare). (right) Transcribed Tabla Loop in Hindi (Dadra – 6 Beat). 138
Figure 78 - Comparison of Filter and Wavelet Front-end. 140
Figure 79 - “Chick” hit detection results for Filter Front End (left). "Boom" hit detection results for Filter Front End (right). 140
Figure 80 - Block Diagram of Transcription Method..... 144
Figure 81 - Fret data (red), Audio pitches (green) , and the resulting detected notes (blue lines). 145
Figure 82 - Sheet music of Sitar Performance. The top notes are the audible notes, while the lower notes are the fret position. Notice the final three notes were pulled.. 146

Figure 83 - Graph of Audio-Based Pitch extraction on an ascending diatonic scale without drone strings being played. 151
Figure 84 - Graph showing the effect of texture window size and regression method. . 154
Figure 85 - Screenshot of the data capturing process. The dots on the screen correspond to the markers taped onto the human body. 159
Figure 86 - Confusion matrix of human perception of 40 point light displays portraying 4 different emotions. Average recognition rate is 93%. 160
Figure 87 - Recognition results for 5 different classifiers. 162
Figure 88 - Graph showing “Leave One Out” classification results for each subject using multiplayer perceptron and support vector machine learning classifiers..... 162
Figure 89 - Confusion matrix for “subject independent” experiment using support vector machine classifier..... 163
Figure 90 - What emotion is the violin player portraying?..... 164
Figure 91 - The *ETabla* in a live concert. Taplin Auditorium, Princeton University, April 25, 2002. 167
Figure 92 - Diagram of Gigapop Ritual setup. 169
Figure 93 - Gigapop Ritual Live Performance at McGill University with left screen showing live feed from Princeton University and right screen showing real time visual feedback of veldt. 170
Figure 94 - "Saraswati's ElectroMagic" Performances at Princeton NJ and Hamamatsu, Japan. 172
Figure 95 - *ESitar* Interfaced with Trimpin's Eight Robotic Turntables. 174
Figure 96 - DeviBot and *ESitar* on Stage for First Time April 18th, 2006. 175
Figure 97 - MIR Framework for Human/Robot performance. 176
Figure 98 - *MahaDeviBot* and *ESitar* in Concert on November 4th, 2006 in MISTIC Annual Concert Victoria BC..... 177
Figure 99 - *MahaDeviBot* being controlled with the *ESitar* at NUS Arts Festival in Singapore. 181
Figure 100 - Performance at New York University at the International Conference on New Interfaces for Musical Expression June 11, 2007..... 183
Figure 101 - The seven main *swara-s* shown using a C Scale..... 200

Figure 102 - Notation of 12 basic *swara-s*..... 201
Figure 103 - The Tanpura and names of the parts of the instrument [7]. 203
Figure 104 - Thaat system. with appropriate scales..... 208
Figure 105 - Chart of Raga-s corresponding to Season (Time of year)..... 209
Figure 106 - Chart describing correspondence between Raga-s, Thaat-s and time of day [7]. 210
Figure 107 - Common *Thekas* with *Bol* patterns [7]..... 212
Figure 108 - PIC 18f2320 Pin Diagram. 216
Figure 109 - Basic Stamp II and Basic Stamp IIsx. 216
Figure 110 - Stepper Motor Circuit Diagram. 220
Figure 111 - Circuit Diagram for using a Solenoid 221
Figure 112 - MIDI Out Circuit Diagram..... 223
Figure 113 - MIDI In Circuit Diagram. 223
Figure 114 - Illustration of kNN Classifier showing class 1 and class 2 points with 2 features and a prediction point which would be classified as class 1 based on the proximity. 226
Figure 115 - Illustration of Decision Tree for example problem of classifying Traditional Indian Classical Music and Western Music using attributes of drums, strings, wind instrument type..... 227
Figure 116 - Illustration of neuron in brain: nucleus, cell body, dendrites and axon. 229
Figure 117 - Illustration of synapse converting signals from an axon to a dendrite..... 230
Figure 118 - Illustration of an artificial neuron. 231
Figure 119 - Illustration of a weighted artificial neuron. 232
Figure 120 - Illustration of three layer architecture of an artificial neural network. 232
Figure 121 - Graph of sound of the *Bayan* as a function of time..... 235
Figure 122 - Graph showing where ramptime finds maximum value of first peak and returns number of samples to point R. 237
Figure 123 - Graph showing spectral centroid for one frame of a signal in the frequency domain. [208]..... 238
Figure 124 - Graph showing zero crossing feature finding eight points where time domain signal crosses zero..... 239

Figure 125 - Diagram of a four level filter bank wavelet transform using FIR filter pairs H0 (low pass) and H1 (high pass)..... 241

Acknowledgments

I would not be in graduate school if I had not met my first mentor in Computer Music at Princeton University, Dr. Perry R. Cook. He showed me how to combine my skills as an engineer with my passion for the musical arts. Many thanks to Ge Wang, Ananya Misra, Perry R. Cook and others at the Princeton SoundLab involved in creating the ChucK music language. Huge thanks to Dan Trueman, Tae Hong Park, Georg Essl and Manjul Bhargava who helped in early experiments that took place at Princeton. I am forever grateful to George Tzanetakis, Peter Driessen, Andrew Schloss, Naznin Virji-Babul, and Afzal Suleman for advising me throughout my Ph.D. research at University of Victoria. Huge thanks to Manjinder Benning, Adam Tindale, Kirk McNally, Richard McWalter, Randy Jones, Mathieu Lagrange, Graham Percival, Donna Shannon, Jennifer Murdoch and the rest of the MISTIC team at University of Victoria for their constant support. Special thanks to Arif Babul, a Physics professor at University of Victoria for his guidance and inspiration. Special thanks to Stephanie Kellett for hand drawing the *"Artificial Saraswati"* front cover image. The robotics research included in this dissertation was accomplished with guidance from Trimpin, Eric Singer, Rodney Katz and Afzal Suleman. Analysis experimentation was accomplished using Marsyas software framework, developed at University of Victoria by George Tzanetakis, Mathieu Lagrange, Luis Gustavo Martins and others at MISTIC. Many thanks to Curtis Bahn for helping compose initial performance experiments for the *MahaDeviBot*. Special thanks to Bob Pritchard of University of British Columbia School of Music for his intensive editing and philosophical outlook to this document. A loving thanks to Manjinder Benning, Satnam Minhas, & Jess Brown for the musical brotherhood formed while writing this dissertation which helped spawn and test drive many of the concepts and ideas through rehearsal and live performance. Thanks to my supportive roommates Yushna Saddul and Stephanie Kellett who dealt with my insanity while writing my thesis. A final thanks to my sister Asha Kapur and my parents Arun and Meera Kapur for their life guidance and unconditional love and support.

The following is a list collaborators for each chapter:

The Electronic Tabla: Perry R. Cook, Georg Essl, Philip Davidson, Manjul Bhargava

The Electronic Dholak: Ge Wang, Philip Davidson, Perry R. Cook, Dan Trueman, Tae Hong Park, Manjul Bhargava

The Electronic Sitar: Scott R. Wilson, Michael Gurevich, Ari Lazier, Philip Davidson, Bill Verplank, Eric Singer, Perry R. Cook

Wearable Sensors: Manjinder Benning, Eric Yang, Bernie Till, Ge Wang, Naznin Virji-Babul, George Tzanetakis, Peter Driessen, Perry R. Cook

The *MahaDeviBot*: Trimpin, Eric Singer, Afzal Suleman, Rodney Katz

Tempo Tracking Experiments: Manjinder Benning, George Tzanetakis

Rhythm Accompaniment Experiments: George Tzanetakis, Richard McWalter, Ge Wang

Pitch and Transcription Experiments: Graham Percival, Mathieu Lagrange, George Tzanetakis

Virtual Sensor Experiments: George Tzanetakis, Manjinder Benning

Affective Computing Experiments: Asha Kapur, Naznin Virji-Babul, George Tzanetakis

Integration and Music Performance: Andrew Schloss, Philip Davidson, Audrey Wright, David Hittson, Philip Blodgett, Richard Bruno, Peter Lee, Jason Park, Christoph Geisler, Perry R. Cook, Dan Trueman, Tae Hong Park, Manjul Bhargava, Ge Wang, Ari Lazier, Manjinder Benning, Satnam Minhas, Jesse Brown, Eric Singer, Trimpin

Chapter 1

1 Introduction

Motivation & Overview

When the world is at peace, when all things are tranquil and all men obey their superiors in all their courses, then music can be perfected. When desires and passions do not turn into wrongful paths, music can be perfected. Perfect music has its cause. It arises from equilibrium. Equilibrium arises from righteousness, and righteousness arises from the meaning of the cosmos. Therefore one can speak about music only with a man who has perceived the meaning of the cosmos." [71]

The idea of interdisciplinary study is a new theme re-emerging in academic research. It is a break from the norm of the past 200 years, where traditional scholars become experts in one area of study and know every microscopic detail about it. Interdisciplinary study involves a macroscopic view, allowing one to merge together a variety of fields in order to help *"perceive the meaning of the cosmos,"* and push academic research in new directions with the perspective of a scientist, philosopher and artist. The focus of this dissertation is to draw a deeper understanding of the complexity of music, drawing knowledge

from different disciplines including computer science, electrical engineering, mechanical engineering, and psychology.

The goal of this dissertation is to describe how North Indian classical music can be preserved and extended by building custom technology. The technology that is produced from this work serves as an infrastructure and set of tools for people to learn and teach Indian classical music and to better comprehend what it takes to perform *"perfect music"*. The technology will also serve as a means to push the traditional performance technique to new extremes helping forge multimedia art forms of the future.

1.1 Motivation

Historically, the majority of music traditions were preserved by oral transmission of rhythms and melodies from generation to generation. Indian culture in particular is well known for its musical oral traditions that are still prevalent today. In the Western Hemisphere, methods for transcribing music into written notation were introduced allowing more people to learn from the masters, not limiting it to those who had the ability to sit with them face to face. Then in the 1900's the age of audio recordings dawned using phonograms and vinyl, analog tapes, digital compact disks, and multi-channel tracking systems, with each step improving the quality and the accuracy of the frequency range of the recordings. The invention of visual recording where musical performances could be viewed on film, VHS, DVDs, or online QuickTime and You Tube clips, has given musicians a closer look at the masters' performances in order to aid emulation. However, both audio and visual records turn performed music into an artifact, ignoring what is truly important to learn and preserve tradition: *the process* of making music.

The work in this dissertation describes techniques and custom technology to further capture *the process* of becoming a performing artist. A key motivation for this work came in 2004, when Ustad Vilayat Khan, one of India's great

masters of sitar, passed away. He took with him a plethora of information on performance technique that is not preserved in the many legendary audio recordings he left behind.

The tools built and which will be described in detail in this dissertation can serve as pedagogical tools to help make Indian music theory more accessible to a wider audience. This work stands on the shoulders of those who have been in the computer music and audio technology field and have designed a number of different algorithms and techniques to extend 21st Century music. A majority of these researchers have based ideas upon Western music, whereas this work will bring music from India to the forefront to help test, modify and build upon traditional techniques.

1.2 Overview

Research on the process of a machine obtaining gestural data from a human and using it to form an intelligent response is essential in developing advanced human computer interaction systems of the future. Conducting these types of experiments in the realm of music is obviously challenging, but is particularly useful as music is a language with traditional rules that must be obeyed to constrain the machine's response. By using such constraints successful algorithms can be evaluated more easily by scientists and engineers. More importantly, it is possible to extend the number crunching into a cultural exhibition, building a system that contains a novel form of artistic expression that can be used on the stage.

One goal of this research, besides preserving traditional techniques, is to make progress towards a system for a musical robot to perform on stage, reacting, and improvising with a human musician in real-time. There are three main areas of research that need to be integrated to accomplish this goal: Musical Gesture Extraction, Robotic Design, and Machine Musicianship. A concluding section

will discuss integration of the research and how it is used live for performance on stage.

1.2.1 Related Work

Part I of this dissertation provides an overview of related work in order to inform the reader of what has been previously done and how it has influenced this work. Chapter 2 presents a history of musical gesture capturing systems, setting a foundation for what has been done in the past, and giving the reader a sense of the high shoulders that this work stands upon. Chapter 3 presents an in depth history of musical robotics, describing the work of masters in the field who have paved the way in the past century. Chapter 4 presents a history of machine musicianship algorithms, techniques and experiments.

1.2.2 Musical Gesture Extraction

Part II of this dissertation describes research on machine perception. This is accomplished by experimenting with different methods of sensor systems for capturing gestures of a performer. In a musical context, the machine can perceive human communication in three general categories. The first is directly through a microphone, amplifying the audio signal of the human's musical instrument. This serves as the machine's ears. The second is through sensors on the human's musical instrument. This is an extra sense that does not generally arise in human-to-human musical interaction. The third is through sensors placed on the human's body, deducing gestural movements during performance using camera arrays or other systems for sensing. These are analogous to the machine's eyes.

Chapter 5 to 7 describes systems for obtaining data via sensors placed on the traditional instruments. Chapter 5 discusses the first interface known as the Electronic Tabla (*ETabla*), which will lay the initial framework for interface

design based on a traditional instrument. Chapter 6 will describe the next drum interface, the Electronic Dholak (*EDholak*), a multiplayer Indian drum that explores the possibilities of network performance using the internet. Chapter 7 describes the Electronic Sitar (*ESitar*), a digitized version of Saraswati's (Hindu Goddess of Music) 19 stringed, gourd shelled instrument. Chapter 8 will discuss wearable sensors, including methods and experiments for capturing data from the human's body during musical performance.

1.2.3 Robotic Design

Part III of this dissertation describes musical robotics. This section involves developing a system for the computer to actuate a machine-based physical response. This machine must have the ability to make precise timing movements in order to stay in tempo with the human performer. A robotic instrument serves as a visual element for the audience, helping to convince them of the functionality of the interaction algorithms that would be lost by synthesizing through loudspeakers.

The acoustics of the robotic instrument is an interesting research topic, helping to determine what material to create the robot with and with what dimensions. Initial design schemes are to make robotic versions of traditional Indian instruments. Basing the machine on traditional form produces similar challenges to the school of robotics that tries to model the mechanics of the human body in the machine. However, in both cases, the robot should acquire skills which a human could not imagine performing. Chapter 9 describes work on designing The *MahaDeviBot*, a robotic system designed to perform Indian musical instruments.

1.2.4 Machine Musicianship

Part IV describes the final stage of this dissertation, which involves how a machine can deduce meaningful information from all of its sensor data to generate an appropriate response. The first challenge is to deal with the large volume of unstructured data. A feature extraction phase is implemented to reduce the data to a manageable and meaningful set of numbers. Feature selection criteria must be set and prioritized. In a musical context, the machine needs to have a perception of rhythm, which notes are being performed by the human and in what order and time durations, and even emotional content of the performer. Then the machine needs to be able to respond in real-time, and generate meaningful lines. Experiments for this part of the dissertation focus on sitar performance and interaction with the *MahaDeviBot*.

Chapter 10 describes experiments on automatically tracking tempo from a sitar performer. Chapter 11 describes experiments on robotic rhythm accompaniment software based on a real-time retrieval paradigm. Chapter 12 describes custom built software for automatic transcription of sitar performance. Chapter 13 describes methods for using machine learning to generate audio-based "virtual sensors" to extend our process to a larger community. Chapter 14 describes affective computing experiments, using wearable sensors for machine-based human emotion recognition.

1.2.5 Integration

Part V describes how all the research on stage has been integrated to achieve our final goal of preserving and extending North Indian performance. Chapter 15 is a chronological journal describing how technology invented is used in live performance. Chapter 16 discusses conclusions made from the dissertation and research with a detailed outline of key contributions made from this body of work.

Because of the interdisciplinary nature of this work, several Appendicies are included in this document to help give background knowledge to the reader about important musical and engineering theory needed to fully understand the details of this project. Appendix A presents an introduction to North Indian classical music theory. Appendix B provides a background on physical computing that includes sensor and microcontroller technology. Appendix C presents a background on machine learning. Appendix D presents a background on feature extraction methods. Appendix E introduces the computer music languages used for this research. Appendix F is a list of publications that came out of this body of work.

1.3 Key Contributions

This section briefly outlines the key contributions of this dissertation. Specific details are included throughout the dissertation.

Musical Gesture Extraction:

- The *ETabla* is the first hardware device to capture finger position and timing information from a traditional tabla performer, for use in modern multimedia concerts.
- The *EDholak* is the first multiplayer Indian drum performance system that took part in the first Indian Music-based networked performance.
- The *ESitar* is the first modified Indian classical string instrument to have sensors that capture performance gestures for archival of performance technique and for use in modern multimedia concerts.
- Research using the VICON motion capture system, the *KiOm* and the *WISP* is the first work of using wearable sensors on the human performer to learn and preserve more intricacies about North Indian Classical music.

Musical Robotics:

- The *MahaDeviBot* is the first mechanically driven drum machine to perform Indian Classical Rhythms for human-to-robot performances in conjunction with multimodal machine based perception.
- The *MahaDeviBot* also served as means for detailed comparison and evaluation of the use of solenoids in a variety of techniques for striking percussion instruments for musical performance.

Machine Musicianship:

- This research presents the first system for multimodal acquisition from a sitar performer to obtain tempo-tracking information using a Kalman Filter.
- This research presents the first system to use retrieval techniques for generating robotic drumming accompaniment in real-time.
- This research presents the first software to automatically transcribe a performance of a sitar performer using multimodal acquisition methods.
- This research presents the first method to create an audio-based “virtual sensor” for a sitar using machine learning techniques.
- This research presents the first experiments on using motion capture data for machine-based emotion detection.

Section I

Related Work

Chapter

2

2 A History of Musical Gesture Extraction

Interfaces for Musical Expression

Gesture is defined as a *"form of non-verbal communication made with a part of the body, to express a variety of feelings and thoughts"*[1]. This section presents research in techniques to extract musical gestures from a performing artist. Information is generally collected using sensor technology (See Appendix B for more information) that is affixed to musical instruments or even the human body.

A digital controller is a device that utilizes a variety of different sensors that measure human interaction and converts the collected information into the digital realm. For example, a mouse is a controller that uses an optical sensor system to convert hand movement into x and *y* coordinates on a computer screen. The goal of the work presented in this chapter is to invent musical controllers that can help a performer express rhythm, melody, harmony, intention and emotion.

This chapter presents a history of musical gesture extraction, describing systems built by various engineers, musicians and artists. *"Musical interfaces that we*

[1] Available at www.wikipedia.org (October 2006)

construct are influenced greatly by the type of music we like, the music we set out to make, the instruments we already know how to play, and the artists we choose to work with, as well as the available sensors, computers, and networks" [30]. Thus, this chapter presents newly made instruments that leverage traditional playing techniques. This background chapter sets the context of this work and is split into five sections: keyboard controllers, drum controllers, string controllers, wind controllers, and body controllers. The interfaces described are representative of the research in each area and are not an exhaustive list.

2.1 Keyboard Controllers

Electronic piano keyboards are the most well established electronic instruments and have had wide commercial success. Many homes across the world have Casio, Yamaha, Korg, or Roland keyboards with onboard MIDI sound banks for amateur and professional performance. Early interfaces included flashing lights, multiple buttons, and automatic accompaniment in many styles to help beginners play pieces. The latest upgrades to this technology are utilizing USB or firewire interfaces that enable modern players to connect their controllers to their laptops. That way any commercial music synthesis software can be used for maximum flexibility in sound production.

Most of these commercial interfaces do not have the full-size weighted keys that are necessary to approximate true traditional piano performance. They also do not have the ability to reproduce the sound of the "real" acoustic grand piano. This influenced innovators to create systems that can capture gesture data from a real piano. In the 1980's, Trimpin designed a system to captured which fingers were pressing which key on a grand piano. Currently one of the most robust systems to capture this information is commercially available. It is the Piano Bar[2], designed by Don Buchla in 2002 and now sold by Moog Music. It

[2] Available at: http://www.moogmusic.com/detail.php?main_product_id=71 (November 2006)

captures the full range of expressive piano performance by using a scanner bar that lies above any 88-key piano, gathering note velocity as well as a pedal sensor which gathers a performer's foot movement.

The SqueezeVox [34] is a controller based around an accordion that attempts to control breathing, pitch and articulation of human voice synthesis models. Pitch is controlled by the right hand using piano keys, vibrato, after touch and a linear pitch bend strip, while the left hand controls breathing using the bellows. Consonants and vowels are also controlled by the left hand via buttons or continuous controls. The Accordiatron [66] is a controller which also models a squeeze box paradigm, sensing both distance between two end panels and rotation of each of the hands.

Researchers at Osaka University in Japan designed a system for real-time fingering detection using a camera-based image detection technique, by coloring the finger nails of the performer [168].

2.2 Drum Controllers

Electronic percussion instruments are also commercially available in many sizes, shapes and forms. However, commercial interfaces are generally crude devices that capture the velocity of the striking implement and the moment of impact. Research laboratories have dissected the problem even further in order to try and capture the myriad of data needed to accurately describe a percussive gesture, including: angle of incidence of the strike, polar position of strike on the surface, and number of points of contact (when using multiple fingers). One of the main challenges is capturing both quick response times as well as more "intelligent" data about expressive information. This section describes the variety of techniques explored to solve this problem.

The Radio Drum [107] or Baton is one of the oldest digital music controllers. Built by Bob Boie at Bell Labs and improved by Max Mathews (the

father of computer music), the interface uses radio tracking techniques depending on electrical capacitance between two mallets and an array of receiving antennae mounted on a surface. The drum triangulates three separate analog signals that represent the x, y, z coordinates of each stick at any point in time.

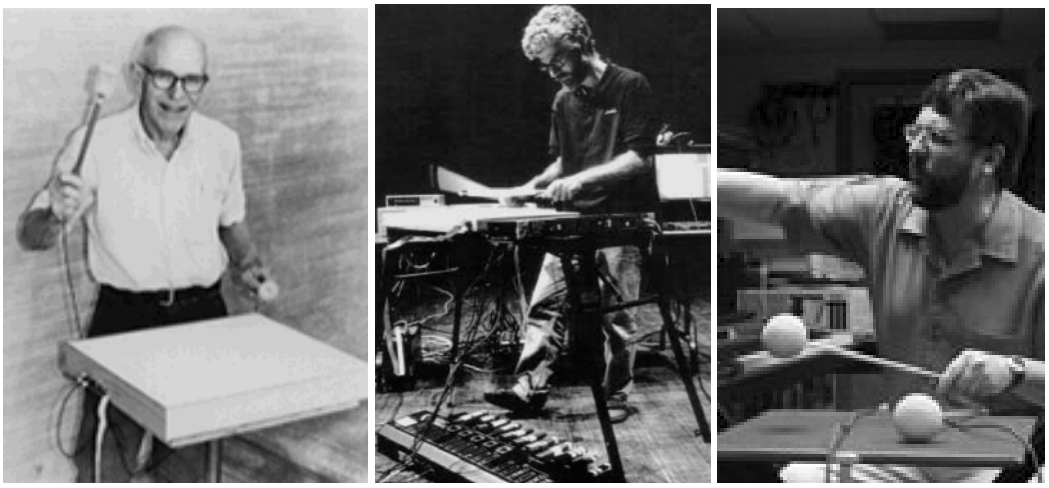


Figure 1 - Radio Baton/Drum used by Max Mathews, Andrew Schloss, and Richard Boulanger.

There have also been a number of methods that modify and augment sticks to gather detailed information about the performer's gestures. The Vodhran [105] uses electromagnetic sensors inside a stick to track six dimensions of position to drive a physical model of a drum. Diana Young built the AoBachi interface which uses accelerometers and gyroscopes mounted inside a set of Japanese bachi sticks to send acceleration and angular velocity data via Bluetooth, so a performer does not have any wires impeding performance [206]. The Buchla Lightening[3] uses infrared light tracking to track the position of wireless sticks in two dimensions.

D'CucKOO [12] is a very successful project combining MIDI marimbas, MIDI drum controllers and six-foot MIDI bamboo "trigger sticks", each based on piezo-electric technology. This evolved into the Jam-O-Drum, which uses an array of commercial based drum pads mounted into a surface to provide a collaborative

[3] Availabe at http://www.buchla.com/lightning (October 2006)

installation for novice performers [14]. The Rhythm Tree [124], one of the largest percussion interfaces in the world, uses 300 sensors to detect direct or indirect strikes, with LED enhanced piezo pads which light up with visual feedback for the users. Sofia Dahl's Selspot system [41] uses video and motion capture to analyze gestures of percussionists.

Figure 2 - D'CuCKOO 6 piece drum interfaces on stage (left). BeatBugs being performed in Toy Symphony in Glasgow, UK (right).

The BeatBugs [1, 194] are durable drum interfaces that can be networked together to control many rhythmic based compositions. These toy-like instruments have been used in symphonies involving children introducing them to music performance and technology. The PhISEM Shaker Percussion [27] controllers used accelerometers to map gestures to shaker physical models as well as control parameters to algorithmic interactive music involving pre-recorded bass, drums and piano.

Currently there are a number of commercially available products that allow anyone to take advantage of the power of electronic drums. Roland has successfully launched their electronic drum set line know as the V-Drums[4], as well as the HandSonic (HPD-15)[5] which uses force sensing resistors to create a hand drum interface. The DrumKat[6] is another powerful drum interface which uses force sensing technology. The Korg Wavedrum uses three contact

[4] Available at http://www.roland.com/V-Drums/ (October 2006)
[5] Available at http://www.roland.com/products/en/HPD-15/ (October 2006)
[6] Available at http://www.alternatemode.com/ (October 2006)

microphones underneath a drumhead in conjunction with synthesis algorithms to give a unique electronic drum sound to every strike. The Buchla Thunder[7] uses more than a dozen pads that sense both pressure and position, while the Marimba Lumina [8] brings mallet percussion to a new level with advanced control parameters including position along the length of bars, dampening, and note density. The Tactex Multi-Touch-Controller (MTC)[9] uses a grid of seventy two fiber optic sensing pads to distinguish multiple sources of pressure. The STC-1000[10] by the Mercurial Innovations Group is a newer, less expensive device that captures pressure from one source.

2.3 String Controllers

Tod Machover and the Hyperinstrument Group at MIT Media Lab have created a multitude of interfaces that combine the acoustic sound of the instrument with real-time synthesis controlled by sensors embedded in the interface, dubbed hyperinstruments. Their work is one of the few examples of serious study of combining synthesis and the acoustic sound of instruments that has been performed in public, most notably the hypercello performance by Yo-Yo Ma [103].

Figure 3 - Hypercello (Machover), SBass (Bahn), and Rbow (Trueman).

[7] Available at http://www.buchla.com/historical/thunder (October 2006)
[8] Available at http://www.buchla.com/mlumina (October 2006)
[9] Availabe at http://www.tactex.com/ (October 2006)
[10] Available at http://www.thinkmig.com/ (October 2006)

Dan Trueman designed the Rbow which is a violin bow with position and pressure sensors [176]. This evolved into the bowed-sensor-speaker-array (BOSSA) which models the violin's performance technique as well its spatial directivity pattern with a 12-channel speaker array [37]. This influenced Charles Nichols who designed the vBow, the first violin interface with haptic feedback using servo motors to simulate friction, vibration and elasticity of traditional performance [119]. Diana Young at the MIT Media Lab designed the hyperbow [205] that measures articulation, changes in position, acceleration, and changes in downward and lateral movements using electromagnetic field measurement techniques, accelerometers, and foil strain gauges. Dan Overholt at University of California Santa Barbara invented the Overtone Violin[121], which incorporated an optical pickup, 18 buttons, 3 rotary poteniometers, a joystick, an accelerometer, 2 sonar detectors, and a video camera, extending traditional violin performance.

The Sensor Bass (SBass) adds a series of slide sensors, force sensing resistors, potentiometers, buttons, an array of pickups, a mouse touch pad, and an biaxial accelerometer to a five string upright electric bass [9]. The Nukelele designed at Interval Research used two linear force sensing resistors to gather pluck and strike parameters to directly drive a plucked string physical model [30].

2.4 Wind Controllers

There are, surprisingly, a number of commercially available wind controllers. Akai have several types of wind controllers called EWIs[11]. The Yahama WX5[12] is modeled after the saxophone fingering system with sensors for obtaining breath and lip pressure. The Morrison Digital Trumpet[13] is an Australian made controller for trumpet performance. All these devices are acoustically quiet instruments that

[11] Available at: http://www.akaipro.com/prodEWI4000s.php (June 2007)
[12] Available at: http://www.yamaha.com/ (June 2007)
[13] Available at: http://www.digitaltrumpet.com.au/ (June 2007)

need a computer or synthesizer to make sound. They are also relatively expensive. A professional wind player might be hesitant to buy such an expensive gear.

The academic approach has been to modify traditional wind instruments with sensor technology, converting them to hyperinstruments. In the 1980's Trimpin built a sensor system for the saxophone to capture data including when the keys were pressed down and information about the wind pressure blown from the mouth of the performer.

The Cook/Morrill trumpet controller was designed to enable fast, accurate pitch detection using sensors on the valves, mouthpiece and bell [116]. Switches and sliders were also added to the interface to allow performers to trigger pre-composed motifs and navigate algorithmic composition parameters [35].

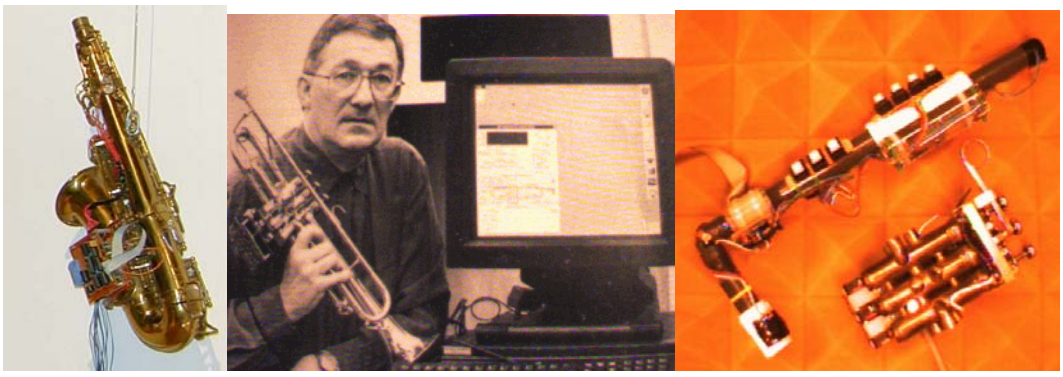


Figure 4 - Wind Hyperinstruments: (left) Trimpin's Saxophone, (middle) Cook/Morrill trumpet, (right) HIRN wind controller.

The HIRN wind controller [26], sensed rotation and translation in both hands, arm orientation, independent control with each finger, breath pressure, and muscle tension of lips was first used to control the WhirlWind physical model now available in *STK Toolkit* [36].

2.5 Body Controllers

Placing sensors on musical instruments is certainly one way to obtain data from a performing artist. However, placing sensors on the human body is another method

that can extend the possibilities of traditional performer. One of the pioneers is Joe Paradiso and his work with wearable sensors for interactive media [125]. One of the earlier contributions was designing wireless sensor shoes that a dancer could use to perform by triggering a number of musical events [124]. Another foot controller is the TapShoe [30] designed at Interval Research that used accelerometers and force sensing resistors to accent beats in algorithmicly composed rhythmic patterns.

Figure 5 - Body Controllers: (left to right) Tomie Hahn as PikaPika, Cook's Pico Glove, Cook's TapShoe, Paradiso's wireless sensor shoes.

There is also early work using a host of sensor systems such as the BioMuse and bend sensors [94, 197]. Head tracking devices using a camera-based approach for musical signal processing are described in [113]. Lamascope [106] tracks body movement using a digital video camera and markings on a performers clothes. A user can control melodic sequences and key with specific gestures. Marcelo Wanderly and Camurri use motion capture systems to gather data from performing musicians gather data about ancillary movements [18, 154]. Experiments using accelerometers in musical performance are presented in [30, 67, 85, 147, 150], placing them on various parts of the body including the head, feet and hands. The Pico Glove [28] was used to control the parameter space for fractal note-generation algorithms while blowing seashells. The MAGIC team at University of British Columbia wrote custom software [55] to use a cyber glove to control gesturally realized speech synthesis for performance [134].

2.6 Summary

Designing systems for capturing gestures from a musical performer has helped expand the capabilities of the modern musician. A myriad of new compositions have been performed educating global audiences of the capabilities of machines in artistic contexts. Using sensors to analyze traditional performance technique is a fairly new direction; however, building the interfaces is the first step. The scientist's and musicians' work presented in this chapter generally focus on Western music. The work presented in this dissertation describes the first instruments modifying North Indian classical instruments. *"Musical interface construction proceeds as more art than science, and possibly this is the only way that it can be done"[30].*

Chapter

3

3 A History of Musical Robotics

Solenoids, Motors and Gears playing music!

A robotic musical instrument is a sound-making device that automatically creates music with the use of mechanical parts, such as motors, solenoids and gears. Innovators in academic, entertainment and art circles have been designing musical robots for decades using algorithms and design schemes that are useful to the computer music society. In this chapter the history and evolution of robotic musical instruments are charted. In addition, future directions of the growing community's collective research are discussed.

To get underway, the author interviewed a number of artists and scientists who have built robotic instruments. These "Renaissance Men" include Trimpin, Eric Singer, Sergi Jorda, Gordon Monahan, Nik A. Baginsky, Miles van Dorssen, JBot from *Captured by Robots*, Chico MacMurtie, and Roger Dannenberg. Interview questions included specifics about the robots each one built, crucial skills needed in order to be a musical robotic engineer, together with personal motivations and future directions for the field.

Why build a robot that can play music? Each artist/engineer had their own reasons. All were musicians who had a background in electrical engineering and computer science and wanted to make new vehicles for interesting performance.

Some had experience in building interfaces for musical expression using sensors and microcontrollers for MIDI input devices and wanted to see what would happen if they "reversed the equation to create MIDI output devices," says Eric Singer. JBot from *Captured by Robots* explains his motivations, "I couldn't play with humans anymore, humans have too many problems, like drugs, egos, girlfriends, jobs....I figured I could make a band that I could play with until I die, and not worry about if anyone in the band was going to quit, and kill the band."

Trimpin told a story about when he was five years old and began to play the flugelhorn. After years of practicing, he developed an allergy of the lips that disabled him from playing the flugelhorn anymore. Thus he took up the clarinet. However, again after years of practicing, he developed an allergy of the tongue that stopped his playing of any reed instrument. Thus, Trimpin was motivated to create instruments that automatically performed in order to express the musical ideas that were present in his innovative mind.

Designing and building a musical robot is an interdisciplinary art form that involves a large number of crucial skills including knowledge of acoustics, electrical engineering, computer science, mechanical engineering, and machining (how to use a mill, lathe and welding equipment). Miles Van Dorssen comments, "I had to learn the mathematics of musical ratios relating to various scales and how waveforms propagate and behave in different shaped media." Eric Singer adds one of the most daunting skills is "learning how to parse a 5000 page industrial supply catalogue." From programming microcontrollers, to programming real-time system code, to using motors, gears and solenoids in conjunction with sensor technology while still having an artistic mind about the look, feel, transportability of the devices being designed, and most importantly, the acoustics and agility for sound making in order to create expressive music; These innovators deserve the title of "Renaissance Men".

In this chapter, musical robots of every type, shape and form will be presented. Section 3.1 discusses piano robots. Section 3.2 discusses robots used

for playback of audio. Section 3.3 discusses percussion robots while section 3.4 and 3.5 discuss string and wind robots respectively. Section 3.6 presents discussions on future directions of the field and postulates the importance of these devices in many research and entertainment areas.

3.1 Piano Robots

The Player Piano is one the first examples of an automatic mechanically played musical instrument, powered by foot pedals or a hand-crank. Compositions are punched into paper and read by the piano, automatically operating the hammers to create chords, melodies and harmonies.

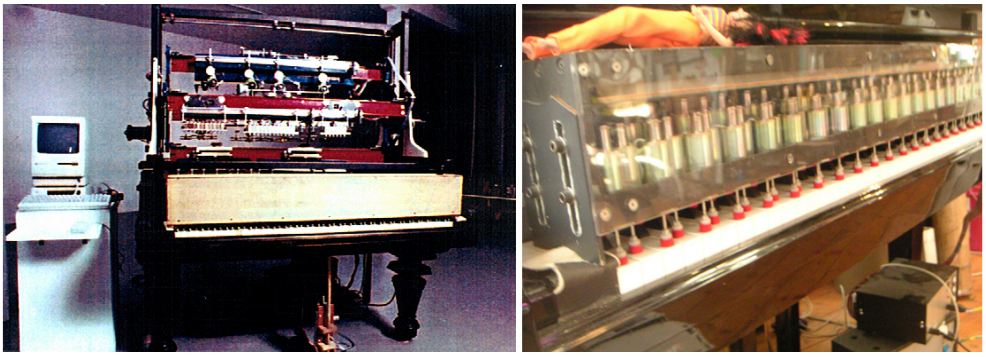
A French innovator, Fourneaux, invented the first player piano, which he called “Pianista” in 1863. In 1876, his invention was premiered at the Philadelphia Centennial Exhibition. In 1896, a man from Detroit named Edwin Scott Votey invented the “Pianola” which was a device that lay adjacent to the piano and performed pressing keys using wooden fingers. Pre-composed music was arranged on punched rolls of paper and powered by foot pedals. In 1897, a German innovator named Edwin Welte introduced a Player Piano which used loom technology invented by Jacquard Mills, where punched cards controlled weaving patterns in fabric [109].

Up until 1905, the piano rolls were created by hand from the music score directly, and hence, when played lacked expressiveness. In 1905, Ludwig Hupfeld of Leipzig built a “reproducing piano” he named “Dea”. It recorded an artist’s performance capturing the expressivity, tempo changes, and shading. In 1904, Welte improved upon his earlier designs and created his own reproducing system that was powered using an electric pump. This allowed the entire apparatus to fit inside the piano, the foot pedal, and keys were removed, turning the player piano into a cabinet-like musical box [109].

In 1886, the German Richard Eisenmann of the Electorphonisches Klavier firm positioned electromagnets close to a piano string to induce an infinite sustain. This method was not perfected until 1913 [109]. This led the way to electronic systems for control of mechanical pianos. Piano rolls were replaced by floppy disks, then compact disks, then MIDI, then software on laptops and software programs like *MAX/MSP* [135] and *ChucK* [190].

Today, automated pianos controlled by MIDI data can be purchased from companies such as QRS Music[14] and Yamaha[15]. QRS Music made a piano called "Pianomation" which can be retrofitted to any piano, while Yamaha makes the factory installed "Disklavier" system.

In the 1980's Trimpin designed the "Contraption Instant Prepared Piano 71512" [175] (Figure 6(a)) which "dramatically extends the whole harmonic spectrum by means of mechanically bowing, plucking, and other manipulations of the strings – simultaneously from above and below – through a remote controlled MIDI device." A combination of mechanized motors can tune the instrument, alter the frequency ratio and expand the timbre of the instrument. It can be played by a human performer or a piano adaptor (Figure 6(b)) which strikes the keys automatically (similar idea to Votey's first "Pianola").



(a) **(b)**

Figure 6 - Trimpin's automatic piano instruments (a) contraption Instant Prepared Piano 71512[175] (b) piano adaptop that strikes keys automatically

[14] http://www.qrsmusic.com/
[15] http://www.yamaha.com/

Another approach is the humanoid technique in which the engineers model the entire human body performing an instrument. A team at Waseda University in Tokyo created the famous musical humanoid WABOT-2 which performed the piano with two hands and feet while sight-reading music with its own vision system [141].

3.2 Turntable Robots

In the 1970s, musicians did not have the luxury of technology that could play back a specific sound on cue with a variety of interfaces, such as samplers do today. Seeing into the future, Trimpin began creating the world's first automatic turntable robot [174]. This device could be controlled to start or stop, speed up or slow down, go forward or go reverse, all with the signals from a Trimpin music protocol (before MIDI). Further extending the concept, eight turntables were built, networked together, and controlled like octaves on a piano. Later, in the 1980's once the MIDI standard emerged, the eight robotic turntables were retrofitted so any MIDI Device could control them. Figure 7 shows images of the retrofitted robotic turntables.

Figure 7 - Trimpin's eight robotic turntables displayed in his studio in Seattle Washington.

3.3 Percussion Robots

Percussion robots are presented in three categories: membranophones, idiophones, and extensions.

3.3.1 Membranophones

Traditionally, membranophones are drums with membranes [144]. Drums are struck with the hands or with sticks and other objects.

One approach to creating a robotic percussive drum is to make a motor/solenoid system that strikes the membrane with a stick. Researchers at Harvard University designed a system to accomplish robotic drum rolls with pneumatic actuators with variable passive impedance. “The robot can execute drum rolls across a frequency comparable to human drumming (bounce interval = 40-160 ms). The results demonstrate that modulation of passive impedance can permit a low bandwidth robot to execute certain types of fast manipulation tasks” [68].

Researchers at MIT had a different approach, using oscillators to drive either wrist or elbow of their robot (named “Cog”) to hit a drum with a stick. As shown in Figure 8, “…the stick is pivoted so it can swing freely, its motion damped by two felt or rubber pads. By using a piece of tape to modulate the free motion of the stick, the number of bounces of the stick on the drum could be controlled” [195].

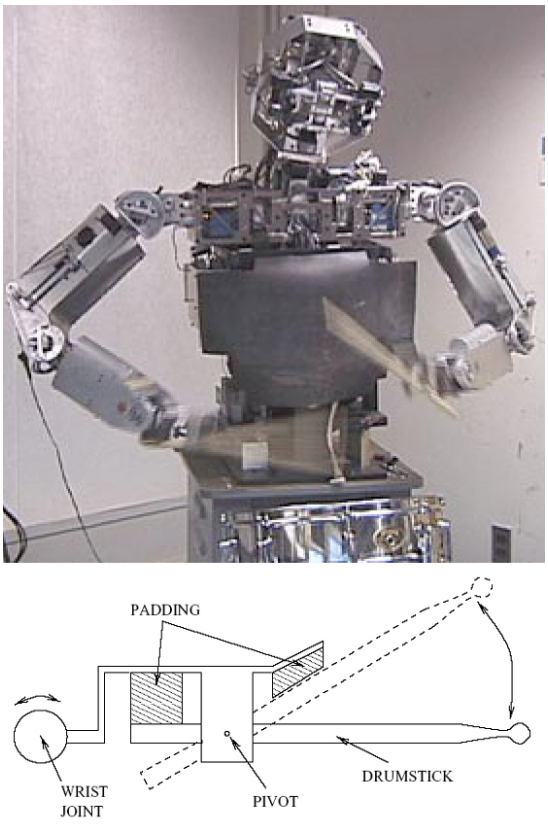


Figure 8 - Williamson's "Cog" robot playing drums. [195]

The team of Dr. Mitsuo Kawato developed a humanoid drumming robot which could imitate human drumming using hydraulics for smooth motion [4]. Trimpin in the 1970's took a different approach modifying drums so they can be played in a new way. He built *"…a revolving snare drum which creates a 'Leslie' effect as it turns rapidly in different directions"* [174]. Chico MacMurtie with Amophic Robot Works has made a variety of robotic humanoids which perform drums with silicon hands as shown in Figure 9(a). [104].

N.A Baginsky built two robotic drummers. The first was "Thelxiepeia" (Figure 9(b)), which performed a rototom with a simple striking mechanism and used a rotorary motor to control the pitch. The second was "LynxArm" which could play five drums at the same time [8].

Captured by Robots has two sets of robotic drummers as well, "DrmBot0110" and "Automaton" (Figure 9(c)) which perform live with other robotic members [75].

(a) **(b)** **(c)**

Figure 9 - (a) Chico MacMurtie Amorphic Drummer[104], (b) N.A Baginsky's robotic rototom "Thelxiepeia"[8], (c) JBot's *Captured by Robots*' "Automation" [75]

3.3.2 Idiophones

Traditional examples of idiophones include xylophone, marimba, chimes, cymbals, and gongs [144]. Trimpin, designed some of the first automatic mechanical percussion instruments as far back as the 1970s. Using solenoids, modification makes it possible to control the sensitivity of how hard or soft a mallet strikes an object [174]. Figure 10 shows example instruments, including cymbals, cowbells, woodblocks, and even a frying pan! Godfried Willem Raes and the Logos Foundation designed many percussion mechanical devices. One of the most popular is the automatic castanet performer showcased in New York City at the International Conference on New Interfaces for Musical Expression in June 2007 [139].

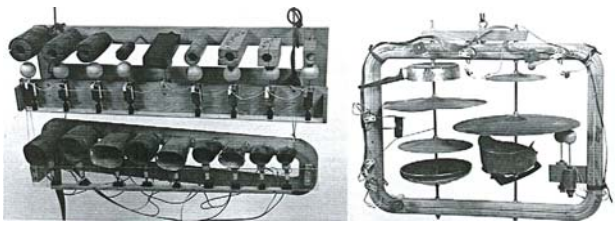


Figure 10 - Trimpin's robotic Idiophones.[174]

Figure 11 - LEMUR's TibetBot [159]

Eric Singer with LEMUR designed the "TibetBot" [159] which performs on three Tibetan singing bowls, using six arms to strike and aid in generating tone. The arms are triggered by MIDI controlled solenoids, each pair producing a high tone with an aluminum arm and a low tone with a rubber-protected arm. This device is shown in Figure 11.

Miles van Dorssen, in "The Cell" project created a number of robotic percussion instruments including an eight-octave Xylophone, Bamboo Rattle, high hat, gong, jingle bells, and tubular bells. [50]

Trimpin's "Conloninpurple" installation also fits under this category as a xylophone type instrument. It is a seven-octave instrument with wooden bars and metal resonators using a "dual resonator system". "The closed resonator amplifies the fundamental tone, the open extended 'horn' resonator amplifies a certain overtone which depends on the length of the horn extension" [174]. Each bar uses an electo-magnetic plunger which shoots up and strikes the bar when an appropriate MIDI message is recieved. This instrument is shown in Figure 12.

Figure 12 - Trimpin's "Conloninpurple" [174]

3.3.3 Extensions

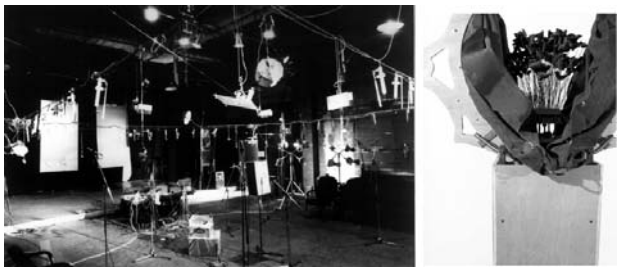
Extensions are percussion robots that do not fall into the two previous categories, transcending tradition to create new identities and art forms of musical sound.

One approach is to combine many instruments in one device as seen in Trimpin's "Ringo" which uses a solenoid-plunger system to strike 120 different instruments including xylophone bars, cylinders, bass drum, wooden cylinders, and many more [174]. Gordon Monahan had similar ideas, making an orchestra out of electronic surplus and trash that he named "Multiple Machine Matrix" (Figure 13 (a)). He later made a scaled down version known as "Silicon Lagoon" [115].

LEMUR has similar motivations in the design of ModBots, which are modular robots that can be attached virtually anywhere. "A microcontroller administers the appropriate voltage to hit, shake, scrape, bow, spin, or pluck sound from any sonorous object with the precision one would expect from digital control." ModBots are an armada of devices including HammerBots (beaters), SpinnerBots (wine-glass effect resonators), RecoBots (scrapers), SistrumBots (pullers), VibroBots (shakers), BowBot (bowers), PluckBot (pluckers)[159]. One example of how these were used was LEMUR's ShivaBot that was multi-armed percussion Indian god-like robot [160].

Another LEMUR robot is the !rBot shown in Figure 13(b). This instrument contains rattling shakers embedded within a shell. "Inspired by the human mouth, the goal of !rBot was to develop a percussive instrument in which

the release of sound could be shaped and modified by a malleable cavity. As the cavity opens and closes, it effectively acts like an analog filter, shaping the sound of the enclosed percussive instrument" [159].



(a) **(b)**

Figure 13 - (a) Gordon Monahan's "Multiple Machine Matrix" [115] (b) LEMUR's !rBot [159]

"Liquid Percussion" is another music sculpture installation by Trimpin, which is triggered by rainfall with the use of one hundred computer-controlled water valves. Water falls twenty feet into custom made vessels that are tuned to certain timbres. "I am demonstrating natural acoustic sounds … water is released through magnetic fields, gravity causes it to fall at a certain velocity from a particular height, striking a natural medium (glass, metal) and finally results in the sound waves being perceived as pitches and timbres" [174].

Another installation by Trimpin was his "Floating Klompen" (which are Dutch wooden shoes) which were placed in a small pond and acted as 100 percussive sound-producing instruments with mallets inside which struck the shoes [174]. Another nature influenced instrument is the LEMUR ForrestBot [159], which has small egg-shaped rattles attached to aluminum rods whose length determine the frequency of harmonic vibration.

3.4 String Robots

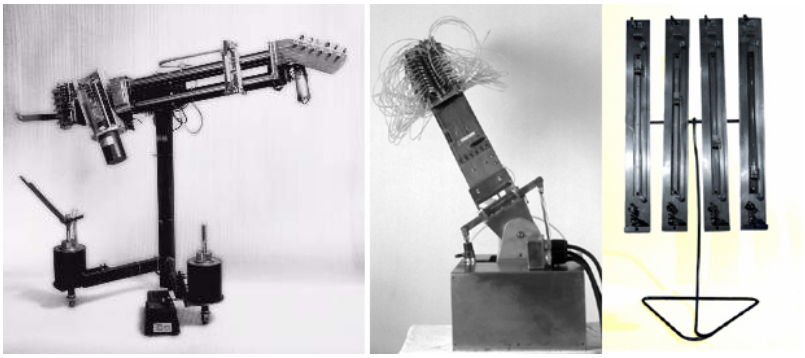
Mechanical devices that perform string instruments will be presented in two categories: plucked bots and bowed bots.

3.4.1 Plucked Bots

This category includes mechanical plucking devices that perform guitar-like instruments. Each one presented has its own technique and style.

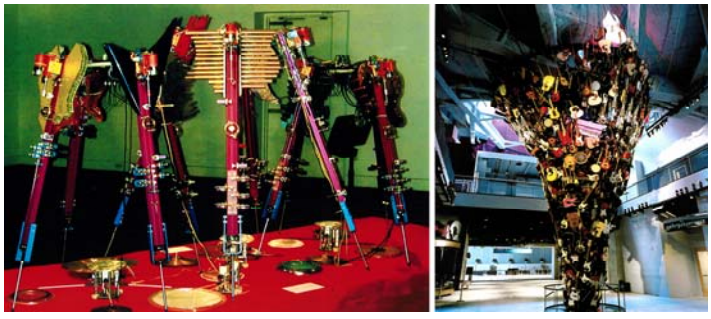
In the early 1990s, Trimpin created a series of twelve robotic guitar-like instruments (Figure 15(a)), an installation called Krautkontrol. Each guitar had a plucking mechanism (using a motor and H-bridge to change directions) four notes that could be fretted (using solenoids) as well as a damper (solenoid) [174].

N.A. Baginsky created a robotic slide guitar between 1992 and 2000 named “Aglaopheme” (Figure 14(a)). The six stringed instrument has a set of solenoids for plucking and damping each string, and a motor which positions the bridge for pitch manipulation [8].



(a) **(b)** **(c)**

Figure 14 - (a) N.A Baginsky’s “Aglaopheme” [8] (b) Sergi Jorda’s Afasia Electric Guitar Robot [77] (c) LEMUR’s Guitar Bot. [160]



(a) **(b)**

Figure 15 - (a) Krautkontrol [174] (b) “If VI was IX” [174] at the Experience Music Project, Seattle, USA.

1n 1997, in Sergi Jorda's *Afasia* [77] project, an electric guitar robot was designed that had a "seventy-two finger left hand", with twelve hammer-fingers for each of six strings. There is also "GTRBot" from *Captured By Robots* that performs guitar and bass at the same time [75].

In 2003, Eric Singer with LEMUR unveiled the GuitarBot [160] that is a series of four string devices. Each has its own plucking device, known as a "PickWheel", which is a series of three picks that rotate at a given speed. Each string also has a belt-driven movable bridge that travels along the length of the string similar to the bottleneck of a slide guitar, with a damper system at one end. The largest robotic guitar project to-date is a permanent installation at the Experience Music Project in Seattle. Trimpin's "If VI was IX"[174] (Figure 15(b)) is a collection of over 500 guitars, each with self-tuning mechanisms, plucking actuators, and pitch manipulation devices.

3.4.2 Bowed Bots

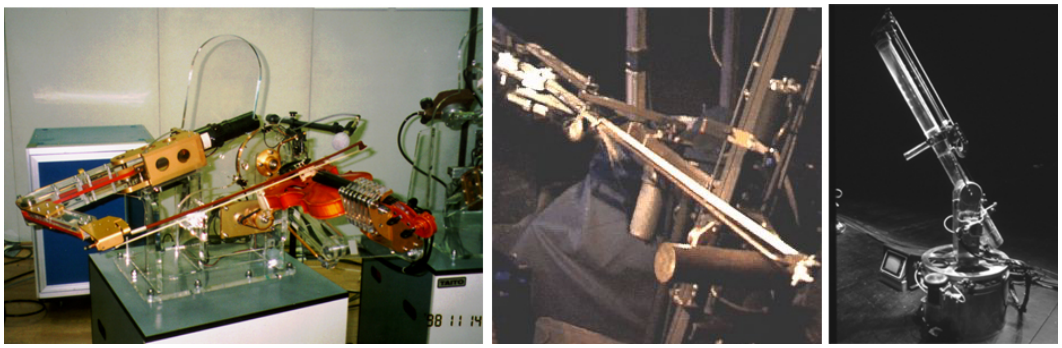
This category includes mechanical bowing devices that perform violin-like instruments. Note that Trimpin's and Eric Singer's guitar-like robots have modes in which they are bowed.

In 1920, C.V. Raman, designed an automatic mechanical playing violin [140] in order to conduct detailed studies of it acoustics and performance. This motivated Saunders to do similar work in 1937 [149].

Another project is the Mubot [79, 80], which was designed in by Makoto Kajitani in Japan in 1989. As one can see from Figure 16(a), this device performs using a real violin or cello with a system for bowing and pitch manipulation.

N.A. Baginsky also created a bowing system for his "Three Sirens" project to perform on bass. The device known as "Peisinoe" [8] has a motorized bow as well as an automatic plucking mechanism.

In Sergi Jorda's *Afasia*, a violin robot was designed using a similar design to their electric guitar robot described earlier, but with one string. "This string is fretted with an Ebow, while the {glissando} finger, controlled by a step motor, can slide up and down"[77].



(a) **(b)** **(c)**

Figure 16 - (a) Makoto Kajitani's Mubot [80], (b) N.A. Baginsky's "Peisinoe" bowing bass [8] (c) Sergi Jorda's Afasia Violin Robot [77].

3.5 Wind Robots

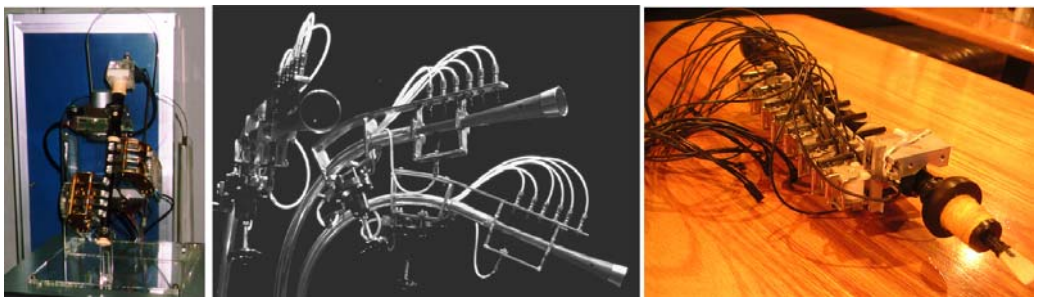
Mechanical devices that perform using wind instruments including brass, woodwinds and horn-type instruments will be presented in this section.

The Mubot [79, 80], introduced in the last section also performs using a clarinet as shown in Figure 17(a). For over ten years, a team at Waseda University has been developing an anthropomorphic robot [162, 167] that can play flute. In their approach, the robot is similar to human shape, size and form that holds a real flute and performs. Trimpin, Miles van Dorssen, and *Captured by Robots* all have included automatic horn shaped instruments on many of their different installations and devices [50, 75, 174]. Toyota designed a humanoid robot which cannot only walk, but can play a real trumpet with artificial lips[16]!

There are also many teams that have built robotic bagpipes. The first set was presented in 1993 ICMC in which the team designed a custom constructed

[16] Available at http://www.toyota.co.jp/en/special/robot/ (July 2007).

chamber fitting to traditional pipes [120]. The system used a belt-driven finger mechanism. *Afasia* also had a "Three-Bagpipe Robot" shown in Figure 17(b). "Each hole can be closed by a dedicated finger, controlled by an electro-valve. Three additional electro-valves and three pressure stabilizers are used for turning the blow on and off" [77]. "McBlare" [43] (Figure 17(c)) is a the latest version of a robotic bagpipe player, which made an appearance at ICMC 2004 in Miami by Roger Dannenberg and his team at Carnegie Mellon. This device actually performs using a traditional set of bagpipes. A custom air compressor was designed to control the chanter and automatic fingers.



(a) **(b)** **(c)**

Figure 17 - (a) Makoto Kajitani's Mubot [80], (b) Sergi Jorda's Afasia Pipes Robot [77] (c) Roger Dannenberg's "McBlare" robotic bagpipes.

3.6 Summary

This chapter described a history of robotic musical instruments. Each artist or scientist had their own way of expressing a traditional musical performance technique through an automated mechanical process. These instruments aided in creating new musical compositions which a human performer could not achieve alone. The work presented generally focused on Western music. This dissertation describes robotic systems which metaphor North Indian Classical music. It also presents a paradigm for testing the performance ability of a variety of mechanical parts for use in performance.

There are certainly many directions for the future of musical robots. Roger Dannenberg sees a future for robotic music in the computer music field saying "we've seen how much audience appeal there is right now, which has always been a problem for computer musicians." Miles Van Dorssen comments, "Eccentric, robotic artists are emerging from all corners of the globe. The future is in their imaginations." Eric Singer adds, "soon, robots will rule the world, and now is the time to get on their good side."

As microcontrollers, sensors, motors, and other computer/mechanical parts get cheaper, simple musical robots are becoming commercially available in toy stores. One favourite toy is the friendly monkey that crashes two cymbals together, shown in Figure 18(a). A series of automatic instruments by Maywa Denki, known as the "Tsukuba Series"[46] is available commercially in Japan. Also, entertainment theme parks such as Walt Disney World[17] have been using mechanical devices to portray musical ideas for decades. A famous attraction is the "Enchanted Tiki Room" (Figure 18(c)) where an armada of mechanical birds sing and dance, providing endless entertainment for children, performing acts that are not possible by humans.

Commercially available professional automatic instruments for the stage are still rare. However, Yamaha's Disklaviers are found in many studios, conservatories, and computer music facilities across the world. Roger Dannenberg says "Yamaha is building a robot marching band, so I expect to see a lot of robot music spectacles in the future."

[17] http://www.disney.com

(a) **(b)** **(c)**

Figure 18 - (a) Example of robotic percussive toy: friendly monkey playing cymbals. (b) Maywa Denki's "Tsukuba Series" [46], (c) "Enchanted Tiki Room" at Walt Disney World, Orlando, Florida, USA.

In education, courses where students build musical robots in order to learn concepts of the interdisciplinary art form are beginning to appear, especially in Japan. An example is a program at the Department of Systems and Control Engineering in Osaka Prefectural College of Technology described in [81].

Also at Waseda University in Japan, the anthropomorphic robot is used *"…as a tool for helping a human professor to improve the sound quality of beginner flutist players. In such a case, the robot is not only used to reproduce human flute playing but to evaluate pupil's performance and to provide useful verbal and graphical feedback so that learners' performances are improved [162]."*

Chapter 4

4 A History of Machine Musicianship

Computer Deduced Musical Meaning

Designing computer programs that will recognize and reason about human musical concepts enables the creation of applications for performance, education, and production that resonate and reinforce the basic nature of human musicianship [145].” The emerging field of “Machine Musicianship” coined by Robert Rowe at New York University, involves building real-time performance systems for human/computer interaction.

This chapter provides an overview of research on machine musicianship. There has been a great deal of work in this area and therefore not every project is included in this chapter. Included projects are the ones that inspired and influenced the research of this dissertation.

The first section describes research in algorithmic analysis used to automatically obtain information from a musician. The second section describes projects that utilize retrieval-based algorithms in order to generate computer responses. The final section presents research of engineers and composers who

have completed the loop and created real-time systems for improvising with a machine on stage.

4.1 Algorithmic Analysis

A machine must have tools for obtaining music information from a live performer. As engineers, *"we must labor mightily to make a computer program perform the analysis required of a freshman music student. Once the work is done, however, the program can make analysis more reliably and certainly much more quickly than the freshman"* [145]. This section briefly describes research on automatic systems for obtaining information on rhythm, pitch, chordal structure, and phrase boundaries. Robert Rowe describes a computer system which can analyze, perform and compose music based on traditional music theory [145].

Automatic music transcription is a well-researched area [93, 101, 203], yet not solved by any means. Automatic pitch extraction is also well researched and the implementation of a method used later in this dissertation is described in [15]. Tempo is one of the most important elements of music performance and there has been extensive work in automatic tempo tracking on audio signals [63, 152].

Other systems which have influenced the community in this domain are Dannenberg's score following system [42], George Lewis's Voyager [98], and Pachet's Continuator [122]. Score flowing involves listening to live performance from a musician and tracking the position real-time on a score. A system created by Roger Dannenberg in 1984, used approximate string matching techniques to track MIDI pitch information in a score [42]. Voyager is a computer music composition system by George Lewis that analyzes a improvising performer in real-time and controls a virtual orchestra responding to the human performer [98]. Pachet's Continuator listens to the style in which a human performs a phrase and the machine engages in a dialogue designed to continue the performers input [122]. These three real-time systems have begun to solve many of the problems of

algorithmic analysis to obtain useful information from the performing artist to generate musically meaningful responses.

4.2 Retrieval-Based Algorithms

This section describes projects that use a retrieval-based algorithm for generating computer-based responses. Retrieval refers to the act of selecting one instance from a large collection of digitized patterns, riffs, loops, audio, or even gestures. This section is included because we later describe a retrieval method for robotic response to a human musician. Related work will be presented in three sections: (1) Interfaces for Information Retrieval, (2) Retrieval for Live Performance Systems, and (3) Rhythm Information Retrieval.

4.2.1 Interfaces for Information Retrieval

Using sensor-based user interfaces for information retrieval is a new and emerging field of study. Hiroshi Ishii describes a tangible transparent interface in which musicBottles [74] can be opened and closed to explore a music database of classical, jazz, and techno music [73]. Ishii elegantly describes in his paper his mother's expertise in "everyday interaction with her familiar physical environment - opening a bottle of soy sauce in the kitchen." His team thus built a system that took advantage of this expertise, so that his mother could open a bottle and hear birds singing to know that tomorrow would be a sunny, beautiful day, rather then having to use a mouse and keyboard to check the system online.

MusiCocktail [108] is a system influenced by the musicBottles project. In this system Force Sensing Resistors placed under coasters measure how much liquid is being added to a particular cocktail glass. By mixing drinks, pre-recorded pieces of music are retrieved and augmented, allowing group participation for the interactive live musical environment system.

4.2.2 Retrieval for Live Performance Systems

There has been some initial research on using music information retrieval for live performance on stage. AudioPad [127] is an interface designed at MIT Media Lab, that combines the expressive character of multidimensional tracking with the modularity of a knob-based interface. This is accomplished using embedded LC tags inside a puck-like interface that is tracked in two dimensions on a tabletop. It is used to control parameters of audio playback, acting as a new interface for the modern disk jockey. An initial implementation of this device was the Sensetable [126]. In these early experiments the system was used to bind and unbind pucks to digital information, using the pucks for manipulation of the data, and visualizing complex information structures.

Block Jam [118] is an interface designed by Sony Research that controls audio playback with the use of twenty five blocks. Each block has a visual display and a button-like input for event driven control of functionality. Sensors within the blocks allow for gesture-based manipulation of the audio files. Jean-Julien Aucouturier from Sony CSL Paris and his team proposed SongSampler [5], a system for music information retrieval with human interaction. The system samples a song and then uses a MIDI instrument to perform the samples from the original sound file.

One idea proposed by our team is to use BeatBoxing, the art of vocal percussion, as a query mechanism for music information retrieval, especially for the retrieval of drum loops [84]. A system that classified and automatically identified individual beat boxing sounds, mapping them to corresponding drum samples was developed. A similar concept was proposed by [117] in which the team created a system for voice percussion recognition for drum pattern retrieval. Their approach used onomatopoeia for the internal representation of drum sounds which allowed for a larger variation of vocal input with an impressive

identification rate. [61] explores the use of the voice as a query mechanism within the different context of Indian tabla music.

Another approach is to use a microphone to retrieve audio recordings from an instrument to retrieve gestural information using machine learning techniques. Initial experiments using audio recordings of a snare drum are presented in [171]. Using Matlab for feature extraction and a variety of machine learning algorithms implemented in Weka, the team was able to get recognition results of over 90 percent. Later in 2005, this system was re-written in *Marsyas* to work in real-time [172]. This research also achieved above eighty percent recognition results using the *Marsyas* system on individual tabla strokes.

Another area of music information retrieval (MIR) that has potential to be used as a live performance tool is the idea of audio mosaicing. One of the earliest frameworks was developed as a general concatenate synthesis system that combined notes taken from segmented recordings [155]. The system had the ability to create high quality synthesis of classical instruments using a MIDI score. Another framework [207] uses a system of constraints to match segments of audio to a target recording. This used high level features to create mosaics. *Mosievius* [97] is one framework that allows a user to create audio mosaics in real-time using an interactive target specification and source selection. The system is integrated for the use of interfaces, such as a keyboard, to specify a source.

4.2.3 Rhythm Information Retrieval

Since our project is rhythmic in nature, this section is a small overview of research in information retrieval with a focus on rhythm. A system for Query-by-Rhythm was introduced in [24]. Rhythm is stored as strings turning song retrieval into a string matching problem. They propose an L-tree data structure for efficient matching. Automatic rhythm analysis is an important area of research. [62, 152] describes initial research on automatic tempo extraction. Obtaining rhythmic features for classification of ballroom dance music is discussed in [49]. [128]

explores using a Dynamic Programming approach to extract similarity of rhythmic patterns independent from the actual sounds. Using zero crossing rate to classify different percussive sounds is described in [64].

4.3 Stage Ready Systems

There are few systems that have closed the loop to create a live human/robotic performance system. Audiences who experienced Mari Kimura's recital with the LEMUR GuitarBot [160] can testify to its effectiveness. Mari performs on a violin and has software that listens to her improvisations and generates robotic response messages to be performed by the GuitarBot. Gil Weinberg's robotic drummer Haile [192] continues to grow in capabilities to interact with a live human percussionist [193]. Haile has two solenoid-based robotic arms that strike a large darbuka drum. The machine listens to a human performer play a rhythm, analyzes perceptual aspects, and uses the information to play along in a collaborative and improvisatory manner.

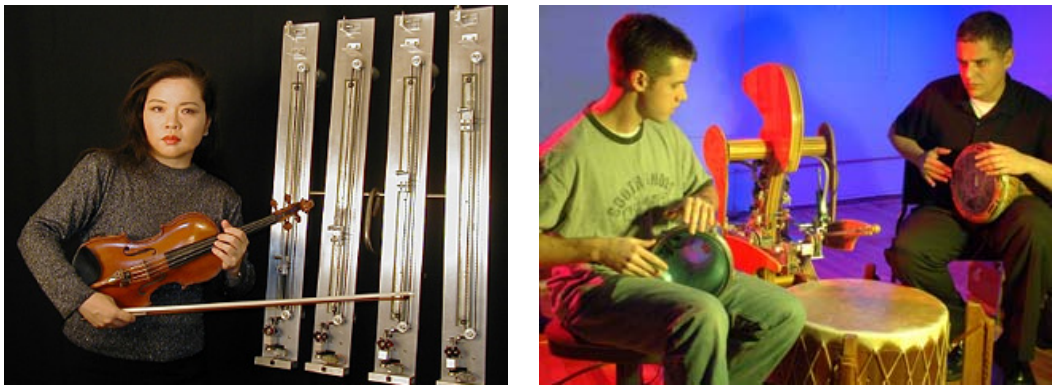


Figure 19 - (left) Mari Kamura and Eric Singer's GuitarBot; (right) Gil Wienberg's Haile

4.4 Summary

Robert Rowe writes, *"By delegating some of the creative responsibility to the performers and some to the computer program, the composer pushes composition up (to a meta-level captured in the process executed by the computer) and out (to*

the human performers improvising within the logic of the work). An interesting effect of this delegation is that the composer must give very detailed instructions to the computer at the same time that she gives up such precise direction of the human improviser. The resulting music requires a new kind of performance skill as much as it enables a new kind of composition.[145]"

The role of machine musicianship in live performance systems continues to grow in importance as machines get faster and algorithms become more accurate. The use of this research area in the preservation of traditional music is also apparent as scientists design advanced transcription systems and automatic classification software. The work in this dissertation applies these methods to the context of North Indian Classical music, with automatic transcription software and systems to enable a human sitar performer to interact in real-time with a robotic drummer.

Section II

Musical Gesture Extraction

Chapter 5

5 The Electronic Tabla

MIDI Indian Drumming

Tabla are a pair of hand drums traditionally used to accompany North Indian vocal and instrumental music. The silver, larger drum (shown in Figure 20) is known as the *Bayan*. The smaller wooden drum is known as the *Dahina*. [39] The pitch can be tuned by manipulating the tension on the *pudi* (drumhead). The *Bayan* is tuned by adjusting the tightness of the top rim. The *Dahina* can be tuned similarly, as well as by adjusting the position of the cylindrical wooden pieces on the body of the drum. Tabla are unique because the drumheads have weights at the center made of a paste of iron oxide, charcoal, starch, and gum (round, black spots shown in Figure 20) [143]. Also, the Tabla makes a myriad of different sounds by the many different ways it is stroked. These strokes follow an Indian tradition that has been passed on from generation to generation, from *guru* (teacher, master) to *shikshak* (student) in India. The combination of the "weighting" of the drum-head, and the variety of strokes by which the Tabla can be played, gives the drum a complexity that makes it a challenging controller to create, as well as a challenging sound to simulate.

Figure 20 - North Indian Tabla. The *Bayan* is the silver drum on the left. The *Dahina* is the wooden drum on the right.

This chapter discusses:

- The evolution of the technology of the Tabla from its origins to the present day.
- The traditional playing style of the Tabla, on which the controller is modeled.
- The creation of a real-time Tabla controller, using force-sensors.
- The physical modeling of the sound of the Tabla using banded waveguide synthesis.
- The creation of a real-time graphics feedback system that reacts to the Tabla controller.
- The experiments on measuring response time of the *ETabla* sensors

5.1 Evolution of the Tabla with Technology

There are a few accounts of the origin of the Tabla. One legend states that the Tabla was created in the 18th Century by Sidhar Khan Dhari, a famous *Pakhawaj* player. *Pakhawaj* is a genre of Indian drum defined by a barrel with drum-heads on either side. The *Mrindangam*, shown in Figure 21, is one drum in this family of drums. It was said that Sidhar Khan provoked an angry dispute after losing a music contest and his Pakhawaj was chopped in half by a sword. Thus, the first Tabla was created accidentally [47]. It is possible that the Tabla is related to drum pairs of antiquity, though references in old music texts completely disappeared after the 10th century [148]. Some Tablas were created out of clay, others out of

wood. As technology for producing metal alloys evolved, the *Bayan* started to be molded out of brass and steel [38].

Figure 21 - The *Mrindangam*, a drum of the *Pakhawaj* family of instruments.

As the popularity of the Tabla spread to the western hemisphere, nearly coincident with emergence of the personal computer, people began to combine the Tabla with computers. In 1992, James Kippen created software that allowed a user to input a traditional Tabla rhythmic pattern, which the computer would then use to synthesize an improvised pattern that followed traditional rules for variation [92]. In 1998, Mathew Wright and David Wessel of University of California Berkeley, aimed to achieve a similar goal, with a real time interface and unique data structure. They successfully created software that generated "free and unconstrained" music material, that could fit into a given traditional rhythmic structure [200]. Jae Hun Roh and Lynn Wilcox created two pressure sensitive pads to input rhythms. These patterns are then used to generate new phrases based on traditional Tabla patterns [142]. Additionally, Talvin Singh created a direct input from his Tabla to digital audio effects, achieving sound manipulations in an invention he calls "Tablatronics" [90].

There are a number of commercially available hand-drum controllers such as Buchla's Thunder [18], Korg's WaveDrum [19], and Roland's HandSonic [20] discussed in Chapter 2. The *ETabla* project, however, uses a new physical model of Tabla acoustics. Our main goal is to preserve the traditional appearance, feel,

[18] Available at: http://www.buchla.com/historical/thunder/ (January 2007)
[19] Available at: http://www.korg.com (January 2007)
[20] Available at: http://www.rolandus.com/pdf/roland/HPD-15.pdf (January 2007)

and performance characteristics of North Indian classical Tabla drumming, while electronically extending the variety of sounds available to the player.

5.2 Tabla Strokes

It is important to understand the traditional playing style of the Tabla to see how our controller takes advantage of the different strokes. Figure 22 is a picture showing the names of the different parts of the Tabla *pudi* (drum head).

Figure 22 - Tabla pudi (drumhead) with three parts: Chat, Maidan and Syahi.

5.2.1 *Bayan* Strokes

There are two strokes played on the *Bayan*. The *Ka* stroke is executed by slapping the flat left hand down on the *Bayan* as shown in Figure 23 (a). Notice the tips of the fingers extend from the *maidan* through to the *chat* and over the edge of the drum. The slapping hand remains on the drum after it is struck to kill all resonance, before it is removed. The *Ga* stroke, shown in Figure 23 (b), is executed by striking the *maidan* directly above the *syahi* with the middle and index fingers of the left hand. When the fingers strike, they immediately release away from the drum, to let the *Bayan* resonate. The heel of the left hand controls the pitch of the *Ga* stroke, as shown in Figure 23 (c). It controls the pitch at the attack of the stroke, and can also bend the pitch while the drum is resonating.

Pitch is controlled by two variables of the heel of the hand: force on to the *pudi*, and the position of the hand-heel on the *pudi* from the edge of the *maidan* and *syahi* to the center of the *syahi*. The greater the force on the *pudi*, the higher the pitch. The closer to the center of the *syahi,* the higher the pitch. [39]

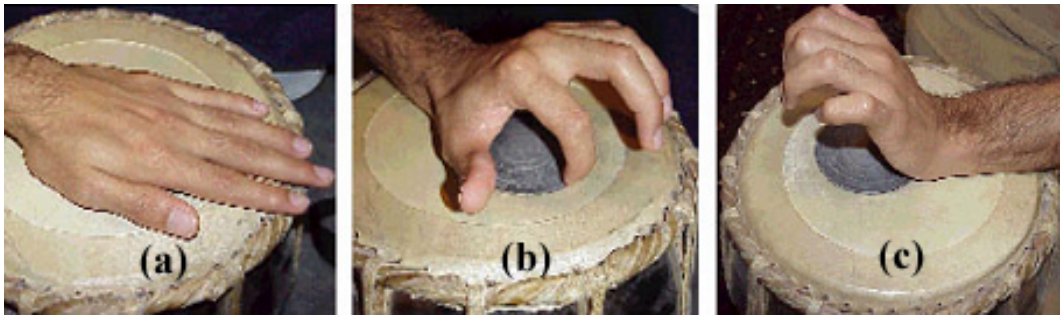


Figure 23 - Ga and Ka Strokes on the *Bayan*.

5.2.2 *Dahina* Strokes

There are six main strokes played on the *Dahina*. The *Na* stroke, shown in Figure 24 (a), is executed by lightly pressing the pinky finger of the right hand down between the *chat* and the *maidan*, and lightly pressing the ring finger down between the *syahi* and the *maidan* in order to mute the sound of the drum. Then one strikes the *chat* with the index finger and quickly releases it so the sound of the drum resonates. The *Ta* stroke is executed by striking the middle finger of the right hand at the center of the *syahi,* as shown in Figure 24 (b). The finger is held there before release so there is no resonance, creating a damped sound. The *Ti* stroke, shown in Figure 24 (c), is similar to *Ta* except the middle and ring finger of the right hand strike the center of the *syahi*. This stroke does not resonate and creates a damped sound. The *Tu* stroke is executed by striking the *maidan* with the index finger of the right hand and quickly releasing, as shown in Figure 25 (a). This stroke resonates the most because the pinky and ring fingers are not muting the *pudi.* [39] The *Tit* stroke, shown in Figure 25 (b), is executed similar to *Na*, by

lightly pressing the pinky finger of the right hand down between the *chat* and the *maidan*, and lightly pressing the ring finger down between the *syahi* and the *maidan*. The index finger then strikes the *chat*, quickly releasing to let it resonate. The index finger strike on the *chat* is further away from the pinky and ring finger, than it is on the *Na* stroke. *Tira* is a combination of two strokes on the *Dahina*, which explains the two syllables of the stroke. It is executed by shifting the entire right hand from one side of the drum to the other. It creates a damped sound at each strike. This stroke is shown in Figure 25 (c) and Figure 25 (d).

Figure 24 - Na, Ta and Ti strokes on the *Dahina*.

Figure 25 - Tu, Tit, and Tira strokes on the *Dahina*.

5.3 The MIDI Tabla Controller

We modeled our controller based on the hand positioning and movements of the strokes discussed. Note that our interest was not specifically to simply copy the traditional Tabla ("[just] copying an instrument is dumb, leveraging expert technique is smart" [31]), the goal of this project was to make an instrument that could be used to create an audio and visual experience that allows a performer expression, and enamors the audience. The *ETabla* is decoupled from the computer generated sound source and hence achieves a new versatility while

preserving the refined performance practices of traditional play. To leverage the existing technique of a skilled Tabla player and to actually test the instrument we decided it would be important to work with an expert traditional Tabla player. We decided that the *ETabla* needed to support traditional strokes accurately. We used square force sensing resistors (square FSR) to input force of different finger strikes, and long force sensing resistors (long FSR) to obtain the position of finger strikes as well as force.[21] All events are converted to MIDI signals and sent out via a MIDI output.

5.3.1 The *Bayan* Controller

The *Bayan* Controller was created using two square FSRs, and one long FSR. Figure 26 shows a layout of these FSRs. The top square FSR is used to capture *Ka* stroke events, when a player slaps down with their left hand. If it receives a signal, then the other two FSRs are ignored. The square FSR in the middle captures *Ga* stroke events, when struck by the middle and index finger of the left hand. The long FSR controls the pitch of the *Ga* stroke events, using two variables: force and position. The greater the force exerted by the heel of the left hand, the higher the pitch. The closer the heel of the hand gets to the *Ga* FSR, the higher the pitch. The pitch can be bent after a *Ga* stroke is triggered. The circuit diagram of the *Bayan* controller is shown in Figure 27.

[21] Available at: http://www-ccrma.stanford.edu/CCRMA/Courses/252/sensors/sensors.html (January 2007)

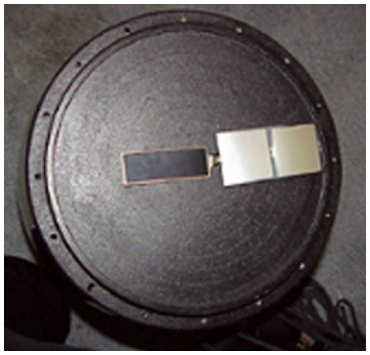


Figure 26 - Electronic *Bayan* Sensor Layout.

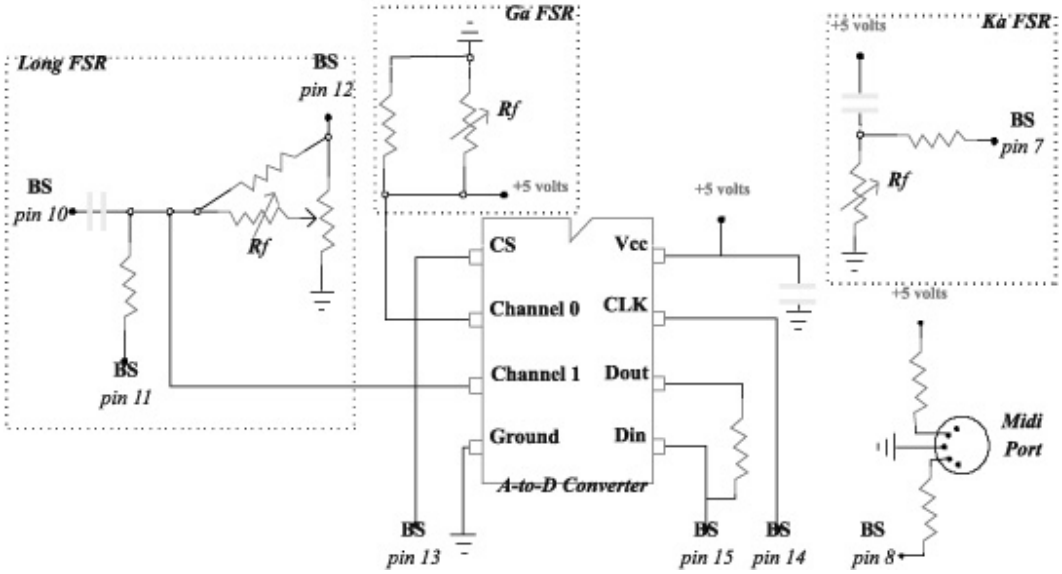


Figure 27 - Circuit diagram of *Bayan* Controller. The *Dahina* Controller uses similar logic.

5.3.2 The *Dahina* Controller

To implement the *Dahina* Controller, we used four FSRs: two long FSRs, one square FSR, and one small FSR. Figure 28 shows a layout of these FSRs. The small FSR triggers a *Tit* stroke event. It measures the velocity of the index finger's strike. The square FSR triggers a *Tira* stroke event. It measures the velocity of the hand slapping the top of the drum. If the *Tira* FSR is struck, all other FSRs are ignored. If the *Tit* FSR is struck, both long FSRs are ignored. The long FSR on the right in Figure 28 is the ring finger FSR, and the long FSR on the

left is the index finger FSR. If there is a little force on the ring finger FSR (modeling a mute), and the index finger FSR is struck at the edge of circle, a *Na* stroke is triggered. If the index finger FSR is struck near the center of the circle, a *Ta* stroke is triggered. If there is no force on the ring finger FSR, and the index finger FSR is stuck, then a *Tu* stroke is triggered. When the ring finger FSR is struck with enough force, and not held down, then a *Ti* stroke is triggered. The circuitry of this controller uses similar logic to that of the *Bayan*. Thus we have modeled every stroke that we discussed above. Figure 29 shows a picture of both controllers in their constructed Tabla encasements.[22] The force sensing resistors were placed on top of custom built wood pieces, and covered with neoprene as a protective layer, making the instrument acoustically quiet, and providing a flexible texture for *ETabla* performance.

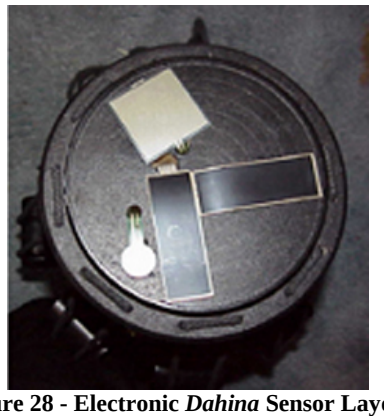


Figure 28 - Electronic *Dahina* Sensor Layout.

[22] The wood pieces were custom built by Brad Alexander at County Cabinet Shop in Princeton, NJ.

Figure 29 - The Electronic Tabla Controller.

5.4 Sound Simulation

The electronic Tabla controller signals can be used with any standard MIDI device to produce sound. However, the typical synthesis methods do not properly mimic the dynamics of the Tabla drums and hence the performance sound in relation to strokes is not well captured. Physical modeling is known to allow for direct physical interactions and hence the control values produced by the Tabla controller can be directly used as inputs rather than first finding a mapping that relates controller-output to synthesis-relevant parameters. We use the “banded waveguides” which were originally introduced for one-dimensional structures like bar percussion instruments [54] but has been generalized to higher-dimensional structures including membranes [53]. Here we will discuss only essential features of the ideas as they pertain to the *ETabla* controller and the reader is referred to [53] for a more detailed discussion of the synthesis method.

Banded waveguides are a generalization of digital waveguide filters [161] that accommodate complex material behavior and higher dimensions by modeling the traveling waves for each modal frequency separately as is depicted in Figure 30.

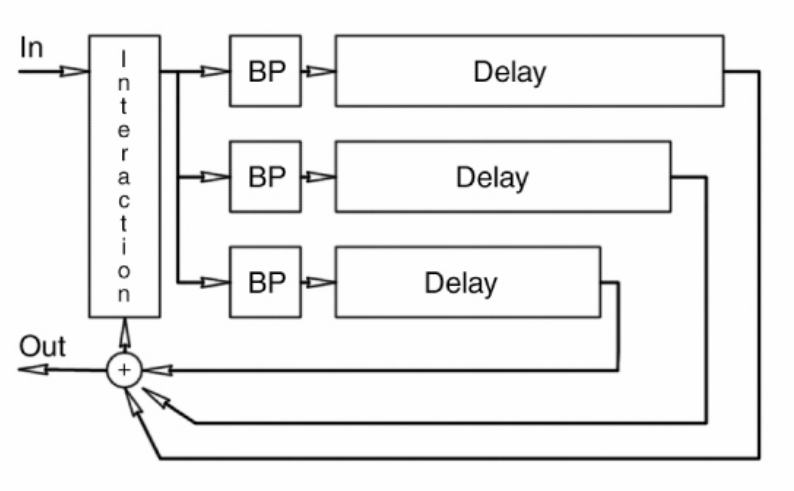


Figure 30 - Banded Waveguide Schematic.

Modes come about as standing waves, which is equivalent to the condition that traveling waves close onto themselves in phase. Hence the task of finding geometric positions from modes corresponds to finding paths that close onto themselves and finding the matching mode for that path. This problem has been studied by Keller and Rubinow [91] and the construction of finding these paths on a circular membrane is depicted in Figure 31. We have taken these paths as approximate representations as they were constructed for uniform membranes. Tabla membranes are non-uniform and heterogeneous in material. The effect of this non-uniformity is the tuning of the partials to harmonic ratios. The physical effect of the successive loading of the membrane lowers the frequencies of the partials [58]. This is equivalent to a slower propagation speed in the medium. Alternatively this can be viewed as a virtual lengthening of the closed path lengths of modes. The effect is modeled in this fashion in banded waveguides.

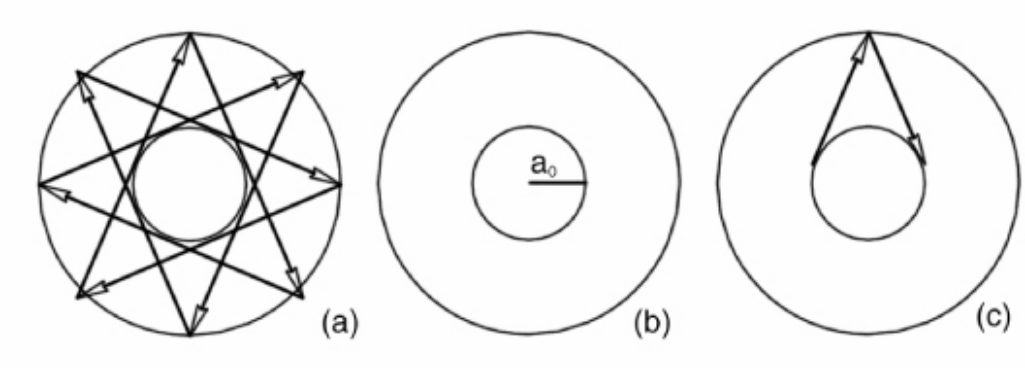


Figure 31 - Figures showing construction of paths that close onto themselves.

Tabla strokes correspond to feeding strike-velocities into the delay lines at the correct positions. A particularly interesting performance stroke is the *Ga* stroke performed on the *Bayan* depicted in Figure 23 (c). It includes a pitch bend that is achieved by modifying the vibrating area due to pushing forward. The exact dynamic behavior of this interaction is so far unknown. We make the simplified assumption that this can be viewed as a moving boundary, which in the case of banded waveguides corresponds to a shortening of the closed wavepaths, in turn corresponding to a shortening of the delay-lines of the model. The stroke starts at about 131 Hz (C-3) and ends at about 165 Hz (E-3) and hence corresponds to a 26% pitch increase and banded wavepath shortening. The simulation was implemented in C++ on a commodity PC and runs at interactive rates. A comparison of a recorded and a simulated *Ga* stroke can be seen in Figure 32. Both strokes are qualitatively similar and are judged by the listeners to be perceptually close. Since banded waveguides are based on standard linear waveguide filter theory, we can make the sonic model as accurate (or absurd) as we like.

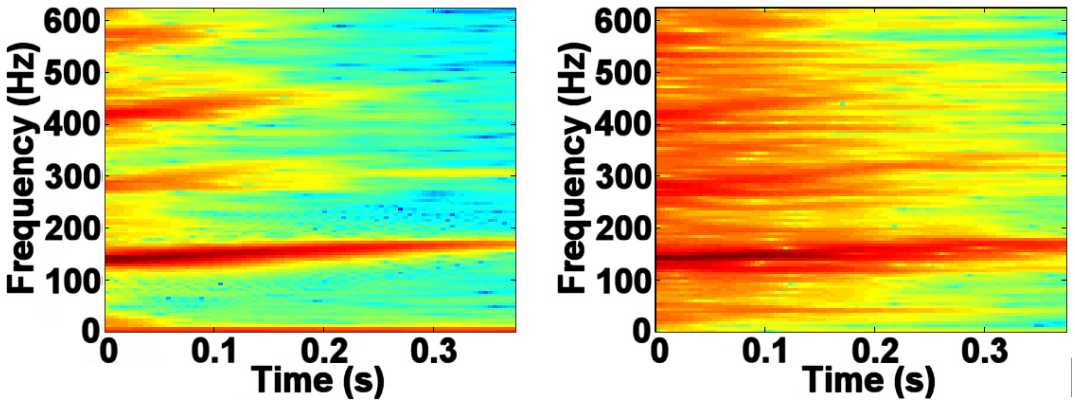


Figure 32 - Sonograms comparing recorded (left) and simulated (right) Ga strike.

5.5 Graphic Feedback

The visual system for the *ETabla* is designed to augment the experience of the *ETabla* for both the performer and audience through the generation of a visual display which responds in parallel with the aural elements of the system. Since audio synthesis requires most of the processing power on the audio machine, control messages from the tabla are routed to a second machine for graphics processing, using custom software built in C++/OpenGL. We will describe the response of the system to *Bayan* strikes.

For the concert performance, our concept for the graphics system was a combination of geometric forms and fluid motion. To respond to the percussive energy of *ETabla* music, the visualization we developed is based on a particle model in which strikes made by the player appear as patterns of small shapes, which form the basic visual elements of the display. As the player makes *Ka* and *Ga* strokes on the *Bayan* controller, particle bursts appear as lines, circles, cardioids, and other shapes depending on the type and quality of each strike as transmitted by the drum. Velocity and pitch are mapped to the size, color, complexity and physical characteristics of the patterns we create. Additional

control messages can be sent to the system by another performer to modify the mapping of *ETabla* signals to visual response as the performance progresses.

Once created, the motion of these particles is governed by a dynamically changing vector field that imposes forces on each particle to achieve a particular overall effect: Strike particles appear, break apart, and return into the background. The behavior of the field is governed by a distribution of ‘cells’ that determine how forces are exerted in their vicinity in response to the number, distribution, and motion of particles in their domain [163]. Through the feedback of cell-particle dynamics, we obtain a system with a short and long term visual response, as the energy introduced by tabla strikes excites secondary behavior in the physical system. By properly modifying the characteristics of particles and cell response behaviors, we can evoke an impression of real-world systems: fanning a flame, striking the surface of water, blowing leaves, or other more abstract behaviors (see Figure 33).

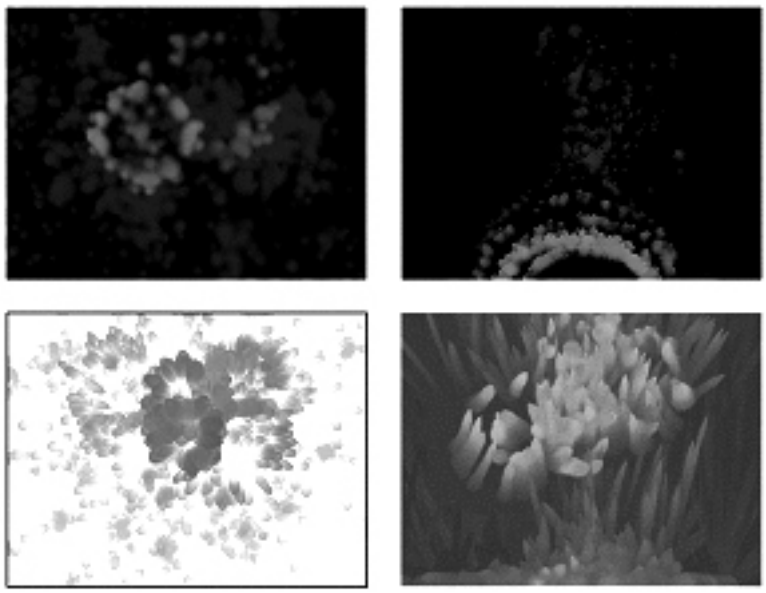


Figure 33 - Different modes of visual feedback.

Feedback from users of the *ETabla* commented that they would sometimes choose to ‘play’ the visuals using the controller, which is an interesting reversal of

our expectation - that perhaps the visual feedback should lead the performer towards playing certain rhythms. Following initial performances, we modified the software to provide a richer visual vocabulary. Beyond its use in performance, visual feedback is also helpful to display the state of our virtual drum. Though the *Bayan* is responsive to changes in tension on the head of the drum, our physical controller does not provide this degree of response. However, we may create the sense of an increase or decrease in tension on the drumhead through compressive or decompressive effects to the visual system. As a teaching tool, the system could display the names and hand positions for the various strikes being made, so that a novice user could reinforce their knowledge and correct their technique. This is an area for future research and implementation.

5.6 User Study of the *ETabla* Sensors

We administered three experiments on measuring the response time of the *ETabla* sensors. We recruited a musician who has been playing Tabla for ten years as our unbiased *ETabla* user who we tested throughout the development process. In the first test the player simply tried to trigger the eight basic traditional Tabla strokes discussed above. He was able to trigger all the strokes, but with a noticeable margin of error. We hypothesized that the errors occurred because at this point the *ETabla* was not mechanically reliable, as the sensors were taped to a slab of wood and a piece of cardboard which were sitting on top of a Tabla shell, without enough support to sustain reliable play. The results of this first battery of tests was sufficient, however, to verify that we had put the sensors in the right places, and a trained Tabla player could execute the strokes, even though he was not specifically trained on the *ETabla*.

After we had successfully created a secure system for encasing the circuit boards and fastening the sensors to the *ETabla* body, using custom built wood pieces, we performed a second round of user tests (A on Figure 34) We measured

the response time of each sensor on the E*Dahina* (the right hand drum of the *ETabla*). A metronome was used to measure the maximum rate at which one could strike a particular FSR before it became unreliable. We connected the MIDI Out messages produced by the *ETabla* to the Roland HandSonic. The tester was asked to perform one strike per metronome click. The metronome speed was increased as long as there was a sound response without immediate problems. Figure 34 is a chart showing the response times of the E*Dahina* by stroke in comparison with a later test. From this user test, it was clear that the two position-only FSRs were responding well. However, the long linear position FSRs were running too slow. This was because the force variable for the long FSR was captured through a slower data acquisition process. We also felt that the ring finger FSR was not calibrated correctly in the code and thus finger strike responses were difficult to pick up.

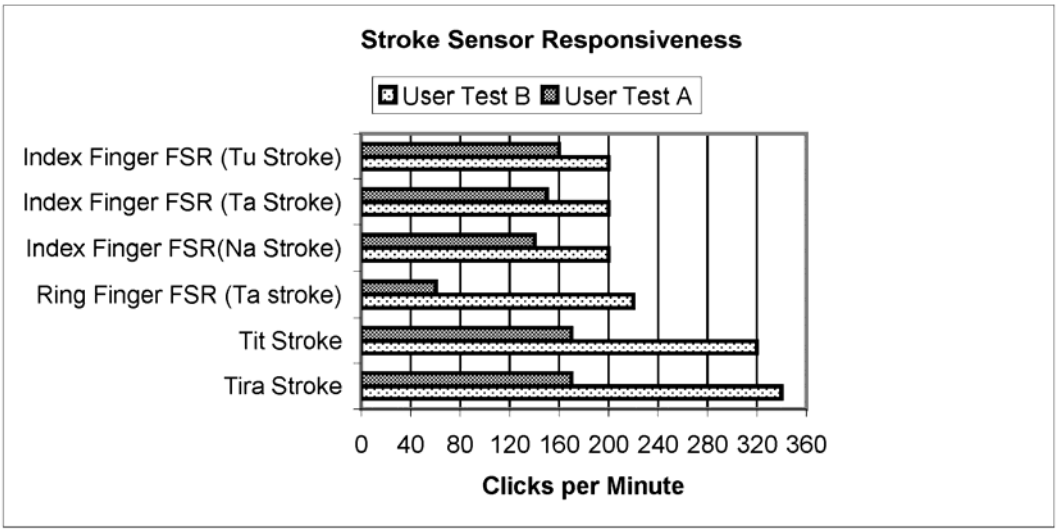


Figure 34 - User testing results of the *ETabla* of User Test A and B. The tests measured maximum strike rate for each sensor as evaluated by an expert performer.

To solve these timing problems, an upgrade of the microprocessor was carried out, yielding a system that could execute commands 2.5 times faster[23]. In user test B, our tester was successfully able to play a recognizable Tin Taal rhythm, a

[23] Upgrade from Basic StampII to Basic StampIIsx, BASIC Stamp Programming Manual. version 2.0. Parallax Inc. Available at : http://www.parallaxinc.com (January 2007).

traditional sixteen beat Tabla cycle, at a moderate tempo. He then tested the response time of the E*Dahina*. Figure 34 also shows the results of his tests. Once again, metronome speed was successively increased as long as the tester at the given speed achieved a sound response without immediate problems. The *Ta* stroke on the ring finger FSR was the slowest for this second user test, only being able to be hit at 60 beats per minute. With the new upgrade, the *ETabla* could now register the same stroke 3.5 times faster at 220 beats per minute. The improvement can be clearly seen in Figure 34. The *Tira* and *Tit* strokes were very fast and were acceptable for performance. The next goal was to raise every stroke close to this level. The tester complained that the two long FSRs were generally difficult to strike and get an immediate response. We knew we could fix this with recalibration. The tester also recommended that the edge of the Index Finger FSR should always play a *Na* stroke and the center should always play a *Tu* stroke. This improvement made the response time much faster.

The remaining improvements to the responsiveness were carried by carefully optimizing the microcontroller code. By some reordering of execution, it was ensured that no mathematical manipulation of variables occurred unless it was needed for a particular event. Wherever possible, divide and multiply operations were converted to shift right and left shift operations to save instruction time. After these improvements were tested, the behavior of the *ETabla* was up to the desired performance level, and we were ready to use the *ETabla* in a live concert setting.

5.7 Summary

We presented the *ETabla*, a real-time device for Tabla performance. Its design was motivated by the traditional instrument and the design takes classical stroke styles into account. We demonstrated how an implementation can be achieved that allows for the use of the *ETabla* in live performance, allowing for the

traditional performance style, but also augmenting the traditional interactions in various ways. Because the interaction and sound production mechanism have been decoupled, the performer can choose the method of sound production, independent of the physical interaction. Hence, non-standard sounds and alternative musical expressions can be presented while maintaining the performance expression of the traditional Tabla. In addition this decoupling allows for performance input to drive output in other modalities. We illustrate this ability by providing performance-dependent visual feedback. In concert, this visual feedback has been used as visual background for performed musical pieces. Similar visual feedback could, however, also be used for teaching purposes by lending additional cues to the student. This aspect remains to be explored in detail. Another interesting future application is the use of the *ETabla* to record performance styles of expert tabla players. This information could be used to facilitate teaching of novice players and the study of classical Indian drumming styles.

Chapter 6

6 The Electronic Dholak

Networked Performance: A Dream worth Dreaming?

The Dholak is a multiplayer Indian folk drum which inspired the idea of exploring multiplayer North Indian networked performance. Is the concept of musicians in multiple locations around the world performing together in real time using high speed Internet, with no latency, in front of live audiences a dream worth dreaming? Is there a valid point in researchers developing novel systems for networked performances, often spending large amounts of grant money to see this dream comes true? Is the music created using these systems worthy of being listened to, or should the performances be called 'live music'? Are the performers really interacting with each other over these long distances?

These are questions being asked by researchers who collaborate to create 'teleconcerts' or 'remote media events' as an application development project for their expensive Internet2 (USA) and CaNet (Canada) lines. Dreamers at Stanford University's CCRMA, McGill University, New York University, University of Southern California, Rensselaer Polytechnic Institute, the Electrotechnical Laboratory in Japan, and many other research facilities across the world have questioned and solved different pieces of the networked performance puzzle.

Standing on the shoulders of these innovators, our team has created a new system for live network performance, to help answer some of the questions for ourselves.

This chapter will present:

- The background to the development of networked media systems
- The design details of the GIGAPOPR networked media software
- The creation of the real-time *EDholak* multiplayer network controller
- The creation of veldt, a real-time networked visual feedback software that reacts to *EDholaks*, and other MIDI devices in multiple locations
- Concluding remarks and future applications of networked media systems.

We will show our full methodology and design, as well as demonstrate that the implementation of high-quality, low-latency, live networked media performance can be straightforward and relatively inexpensive. Further, we evaluate the validity of the aesthetic of the network performance and whether the dream is indeed worth dreaming.

6.1 Background

In the mid-1990s, a team at the Chukyo University of Toyota in Japan performed experiments using ISDN (128 kbps) to connect two or more concert venues with teleconferencing technology to allow musicians in remote locations to maintain ‘musical eye-to-eye contact’. Later, in 1997, the team developed a system to use a low-bandwidth Internet connection, consisting of an Internet server relay which redirected musical messages, and musical synthesis software clients at three different venues. Each unique site controlled frequency values of a single oscillator, and performers on stage transmitted controller data to change their frequency in response to another site’s change [169].

In 1997, at the USA/Japan Inter-College Computer Music Festival in Tokyo, Japan, a team at the Electrotechnical Laboratory in Tsukuba, Japan,

presented their work on a Remote Music Control Protocol (RMCP) that integrated the MIDI and UDP protocol to allow users at separate workstations to play as an ensemble. This system also had visualization feedback software, which reacted to what a user performed [62]. The University of California San Diego and the University of Southern California collaborated in the Global Visual Music Project with the pieces *Lemma 1* and *Lemma 2*, an improvisatory jam session between Greece and the United States [137] presented at the International Computer Music Conference in 1998. Presented all over the world in different versions, MIT's *Brain Opera* allowed observers to participate and contribute content to the performance via the Web [124].

In September 1999, at the AES conference in New York, the Society's Technical Committee on Network Audio Systems demonstrated a live swing band originating at McGill University in Montreal, streamed over a multi-channel audio and simultaneous video connection (combination of UDP and TCP protocol) to a theatre at New York University, where on stage a dancer reacted to the high quality 48 kHz, 16 bit, surround sound music [202]. This system was also used to network concerts between Montreal and the University of Southern California, in Los Angeles and later Montreal and Japan.

In spring 2000, a team at Stanford University's CCRMA presented a networked concert between two multi-channel venues on campus, both with live audiences using the campus intranet and TCP protocol, to test whether an accurate sound image of the remote space could be projected locally. That summer, using the same system at the Banff Center in Canada, ten-channel concert feeds from two concert halls were transported to a mixing room, and mixed down in real time [22]. These systems use the *SoundWIRE*, software that evaluates the reliability of a network by creating an 'acoustic ping' between the two host computers [21, 23]. Later in 2004 this system was used to network three geographically distinct locations (California, Montana and Victoria) in a project entitled 'Distributed MahaVishnu Orchestra'.

The Integrated Media Systems Center at the University of Southern California (USC) has developed YIMA, an end-to-end architecture for real-time storage and playback of high-quality multi-channel audio and video streams over IP, as part of their Remote Media Immersion project. In October 2002, the team successfully broadcast (16 channels of 24-bit 48 kHz samples per second audio and MPEG-2 720p formatted video at 45 Mb/s) a concert by the New World Symphony in Arlington, Virginia to an on-campus venue in Los Angeles, California. In September 2003, the system was tested internationally with a transmission to Inha University in South Korea [156].

Other projects in the last few years have confronted and exploited different aspects of networked music. *The Technophobe and the Madman* was an Internet2-distributed musical performance collaboration between New York University and Rensselear Polytechnic Institute [146]. *FMOL [76]* is a Virtual Music Instrument that was used between Dresden, Germany and Barcelona, Spain in 2001 [78]. *PeerSynth* is a framework developed for peer-to-peer networked performance that makes use of latency as a parameter of synthesis [165]. *SoundMesh* is an application designed to mix audio files in a live Internet2 improvisation [69]. The *Auricle* website is a kind of audio analysis/synthesis enhanced chat room [60]. Most of these do not attempt to mimic live performance over distance directly.

For more information, readers are directed to [10], [59] and [191], that survey network systems for music and sonic art. Also, another innovative article is a 1998 AES white paper [11] which forecast visions of network media performance that have influenced most of the research presented.

6.2 GIGAPOPR: Networked Media Performance Framework

GIGAPOPR is a framework for low-latency, bi-directional network media performance over a high-bandwidth connection used for the Gigapop Ritual performance at NIME 2003. It transmits multichannel uncompressed audio, uncompressed video, and MIDI data among an arbitrary number of nodes. GIGAPOPR served as the software framework for the *Gigapop Ritual*, discussed in detail below.

6.2.1 Challenges in Design

6.2.1.1 Latency

GIGAPOPR was designed to enable performers at geographically remote locations the ability to cooperate and interact with each other – recreating, as much as possible, the experience of playing together at the same place. Thus, one-way latency and round-trip latency both have critical effects on the quality of the interaction. Experiments conducted by CCRMA on quantifying the effects of latency in network performance show that humans perform best at roundtrip bi-directional audio latency between 20 and 30 milliseconds [65]. This was the toughest challenge our team had to face in building the framework.

6.2.1.2 Network Porridge

We define *network porridge* as any prolonged and perceptually significant audio artifact caused by some aspect of the network transmission. Network porridge, like the name suggests, is highly crackly and poppy audio resulting from one or more audio frames failing to reach the destination machine in time. One common cause of network porridge is inconsistent delay introduced by the network during transmission, as a result of dropped or delayed packets. Another cause may be contention between the network interface card (NIC), soundcard, and/or the CPU

on the sending or receiving machine. For example, if the sending machine tries to send a large single chunk of data over the network (such as an entire frame of uncompressed video), it may ‘tie up’ the NIC and delay the transmission of several frames of audio, even when there is ample bandwidth to support both.

6.2.1.3 Compensation for different sound card clock speeds

Most sound cards have an onboard clocking mechanism, which times the capture/playback of audio data. However, it is often the case that the sound card on one machine may have a clock that is slightly faster or slower than another, even if both sound cards are the same model. If the difference in clock speeds of the sending and receiving machines is great enough, then eventual clicks (or porridge) may be introduced.

6.2.1.4 Robustness

The Internet is inherently a best-effort transmission system. Packets can be lost, duplicated, and/or reordered between the end hosts. Transmission control protocols (such as TCP) alleviate this issue by tracking and acknowledging packet delivery, and re-transmitting potentially lost packets. However, since audio data in a live-networked performance must take place in a highly timely manner, packet re-transmission is impractical. Therefore, a system should respond robustly and reasonably to potential network problems.

6.2.2 Design and Implementation

6.2.2.1 Simplicity

The design and implementation of GIGAPOPR is straightforward, with only a few considerations and optimizations for low-latency, high-bandwidth throughput. The framework is divided into three subgroups of applications, one each for audio, MIDI and video. Each group of applications is designed to run in a separate,

autonomous process space. The challenge is finding a way to utilize the potential of the network in a real-time fashion.

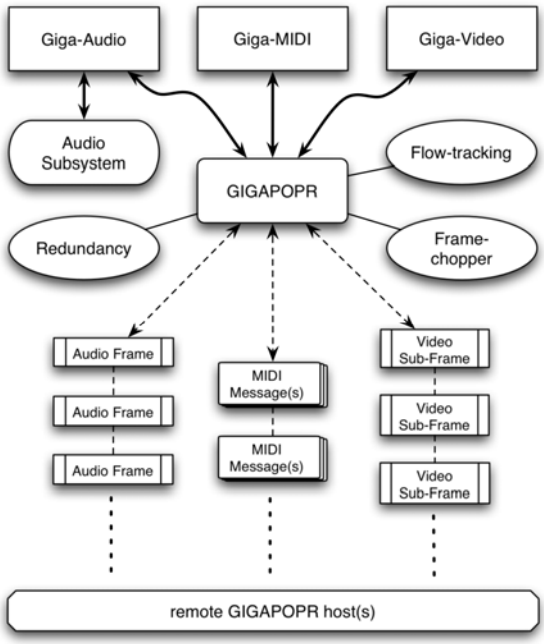


Figure 35 - Flow Control and Sequencing of GIGAPOPR.

6.2.2.2 Flow control and Sequencing

All data packets are transmitted using the GIGAPOPR protocol over UDP. UDP provides efficient and error-checked delivery of packets but is without flow control or congestion control. For our purposes, this is desirable since the system cannot afford to wait for re-transmission of lost audio packets (TCP-esque re-transmission is partly based on timeouts). If a packet is lost, then either the previous frame or silence is played. Furthermore, if the network is congested, there is little that an end-to-end connection can do. In this respect, we hope for the best from the bandwidth ceiling of a high-performance network. In our experience running over Internet2 and CA2Net, this was not a significant problem.

A sequence number is sent in the header of every GIGAPOPR audio packet. This simple sequence numbering scheme enforces ordering of incoming packets, allows the receiver to detect when packets were lost, and also makes possible redundant transmission of data. For example, it is possible for

GIGAPOPR to send copies of each frame of data to increase the chance of at least one of the packets reaching the destination. Sequence numbering for video is more involved since it sends sub-frames.

6.2.2.3 giga_audio

giga_audio is a client/server application for capturing audio at one host and sending it with low latency to a remote host for playback. The mechanism is very straightforward. The capturer/sender application reads in frames of audio from the A/D converter and performs some minimal transformations on the data (type-casting/endian-adjustment) and encloses the data in a packet and sends it out using the transmission module. The size of the audio frame is adjustable. As is to be expected, larger frames will contribute to overall latency, while smaller frames may incur extra network overhead that can lead to dropped packets. For our performance, we used 48,000 Hz, stereo, with buffer sizes of 512 sample frames. Additionally, redundant copies of each frame can be sent. The receiver/playback application receives the packets, performs simple sequence number checks (discarding out-of-date packets, and updating the next packet sequence number to expect) and also manages redundancy, if it is in use. It then pulls out the frames from each packet and sends them to the DAC for playback.

At the time of the performance, giga_audio was implemented using the Synthesis ToolKit (STK) and RtAudio for capture/playback and over a custom transmission module over UDP.

6.2.2.4 giga_midi

giga_midi is the MIDI counterpart of giga_audio. The ‘midi in’/sender host sends one or more MIDI messages in a single packet to the receiver/‘midi out’ host. The MIDI data receiver can be mapped to onboard or external MIDI devices. *giga_midi* was implemented over a custom module written with ALSA and also sent over UDP.

6.2.2.5 giga_video

The *giga_video* application follows the client/server model used by *giga_audio*. The video capture/sender application grabs video frames from any video source and sends it over UDP to the receiver/video playback application.

The design favours the timely transmission of audio and MIDI over that of video. Each video frame is actually sent in separate chunks, and sent with a small intentional delay between each one. This is to avoid tying up the NIC for a single large transmission, which might delay one or more audio packets from being sent on time. In GIGAPOPR, uncompressed 480 by 320 video frames are segmented into 30 by 40 equal-sized chunks and sent separately.

6.2.2.6 Configuration

We ran Linux (Redhat 9) with ALSA and the Planet-CCRMA[24] low-latency kernel. The audio/MIDI data were transmitted between two machines: a Pentium 4/ 2.8 GHz CPU/1 GB of RAM. The video transmission employed two additional machines: Pentium 3/ 400 MHz/128 MB of RAM. The real-time graphical feedback ran on a fifth machine: Pentium 3/800 MHz/ 512 MB of RAM, with GeForce 3 graphics card. Finally, a Pentium 3 laptop controlled additional devices on-stage.

6.2.2.7 Performance and Optimisations

Perhaps the most striking reflection from our implementation of GIGAPOPR is that on today's (and tomorrow's) high-performance networks, it really doesn't take much to get a high-quality bi-directional system up and running. For the most part, it suffices to have competence in implementing network and audio processing interfaces without introducing significant additional latency, and to know the right knobs to tweak. In this section, we discuss some factors that can

[24] Available at: http://ccrma.stanford.edu/planetccrma/software/ (February 2005).

greatly affect overall latency, as well as suggestions from our experience to reduce latency.

Several factors contribute to the overall audio latency of the system: (i) network latency between the source and destination hosts, (ii) end-host latency that involves buffering, context switching, processing, sending data to NIC, network stack processing, and the actual transmission time on the NIC's hardware, and (iii) hardware latency of sound cards and the host machine itself.

The network between the end hosts is the least controllable aspect of the system, in today's best-effort, end-to-end Internet. There is no direct way to even influence the routing of packets, or to avoid or respond to congestion. Until more programmable, dynamically routable networks become mainstream, we cross our fingers and leave these aspects to the underlying protocols and existing routing algorithms.

As for the end-host latency, we do have both direct and indirect control. Starting with the underlying operating system, it can be beneficial to install low latency kernel patches (if running Linux) such as the one packaged with Planet-CCRMA. On Mac OSX, setting the scheduling policies to round-robin for audio and network processing threads while keeping the rest as default first-in-first-out can significantly improve stability and latency for lower buffer sizes. Boosting process priority on both systems can also be helpful.

Finally, machine hardware and soundcard quality can have a big impact on latency and stability. For the machine itself, good bus performance is crucial, as audio I/O, network I/O, and often (depending on the architecture) memory operations may all contend for and share the bus. Yes, faster machines with more memory are good, too. Lastly, soundcard latency can vary vastly from model to model and across vendors. It is worthwhile to ensure all hosts have low-latency soundcards with appropriate configurations and settings.

At the time of the performance, we clocked between 120 and 160 ms round-trip latency between Princeton, NJ, and Montreal, Canada. We were able to

perform using 120 ms latency, and did not implement all of the ‘performance tips’ mentioned above – many of them came out of subsequent experiments. We are optimistic that we can do better on today’s improving networks and from experiences we have gained since.

6.3 The Electronic Dholak Controller

6.3.1 The Traditional Dholak of India

The Dholak (shown in Figure 36) is a barrel shaped hand drum originating in Northern India. It has one membrane on either side of the barrel, creating higher tones on the smaller end, and lower tones on the larger end [95]. The smaller side has a simple membrane, whereas the larger side has *Dholak masala* (a composition of tar, clay and sand) attached to the inside of the membrane, to lower the pitch and produce a well defined tone. The Dholak can be tuned in two ways depending on the type of drum. The traditional Dholak is laced with rope, so tuning is controlled by adjusting a series of metal rings that determine tightness of the rope. Modern Dholaks have metal turnbuckles which are easily adjusted for desired tone. The Dholak is widely used in folk music of villages of India. It is common for folk musicians to build Dholaks themselves from commonly available material. They then use the drums in musical rituals and special functions such as weddings, engagements and births. [157]

Figure 36 - Traditional Dholak.

Two musicians play the Dholak. The first musician strikes the two membranes with their left and right hands. There are two basic playing techniques; the open hand method is for louder playing, while the controlled finger method is for articulate playing. There are a few different positions to play the Dholak, but the most popular is squatting with the drum in front, the bass head on the left, and the treble head on the right. The second musician sits on the other side of the drum, facing the first musician. They strike the barrel with a hard object, such as a spoon or stick, giving rhythmic hits similar to a woodblock sound. [7]

6.3.2 The MIDI Dholak Controller

The design of the Electronic Dholak (shown in Figure 37(a)) is inspired by the collaborative nature of the traditional drum. Two musicians play the *EDholak* (shown in Figure 37(b)), the first striking both heads of the double-sided drum, and the second keeping time with a *Digital Spoon* and manipulating the sounds of the first player with custom built controls on the barrel of the drum and in software. We further explored multiplayer controllers by networking three drummers playing two *EDholaks* at two geographically diverse sites.

(a) **(b)** **(c)**

Figure 37 – Two-player *EDholak* with Piezo sensors, digital spoon, CBox and custom built MIDI Control Software.

Finger strikes are captured by five piezo sensors (three for the right hand and two for the left hand) which are stuck directly on the *EDholak*'s drum skins. Sensors are placed in positions that correlate to traditional Indian drumming (similar to tabla strokes described above). The left drum-skin head captures *Ga* and *Ka* strokes, while the right hand drum-skin captures *Na*, *Ta* and *Ti* strokes.

The *Digital Spoon* has a piezo sensor attached to the back of a flat wooden spoon. There is neoprene padding covering the piezo to keep the striking of the *Digital Spoon* acoustically quiet. The spoon player has the option of striking anywhere on the drum, or floor, triggering a audio/visual response, or striking on a linear force sensing resistor (FSR) on the *EDholak Controller Box*, which augments the audio/visual of the spoon strike and the audio/visual instances of all *EDholak* finger strikes. The *Controller Box* has a long FSR and a knob that the spoon player can use with his left hand to augment all sounds/graphic instances.

All piezo triggers are converted to MIDI by the *Alesis D4*[25] 8-channel Drum trigger box. The *Controller Box* is built using a Parallax Basic Stamp that converts all sensor data to MIDI. When two *EDholaks* are used in distinct locations, piezo generated MIDI signals are transferred using *GIGAPOPR* (custom built software created for *Gigapop Ritual* performance at NIME 2003) and then processed and merged together by an *Alesis D4*.

[25] Available at: http://www.alesis.com/downloads/manuals/D4_Manual.pdf (January 2007)

All MIDI messages are funneled to the *EDholak MIDI Control Software* written for Windows. This software is used by the spoon player to control many parameters of the performance. A user can toggle between a networked performance (two *EDholaks sending* MIDI messages) or just one. The software is custom built to communicate with the *Roland Handsonic*[26]. The user can pre-program patches that they wish to use in performance, in order of occurrence, and simply use the mouse to switch between them during a concert. The software also maps the *Control Box* sensors (Knob and FSR) to different MIDI Control Changes such as pitch, sweep, color, pan and volume, augmenting sounds of piezo MIDI signals going to the *HandSonic.* For example, the performers can start out by playing traditional Dholak sound samples while the spoon player selects a frequency sweep effect which morphs the samples to new expressive rhythmic sounds with the *Controller Box* and *Digital Spoon*. All MIDI messages from the software get transmitted to *veldt* to trigger visual events.

6.4 veldt: Networked Visual Feedback Software

The MIDI messages generated by *EDholak* drummers and spoon players are routed to a graphics computer running the *veldt* software, which synthesizes visuals in response to the patterns of drum triggers and other controller messages. *veldt* is an application that was designed from the ground up for the purpose of visual expression and performance. It receives MIDI messages from digital musical interfaces and maps them to a system of reactive events in order to generate live visuals, which are rendered in real time using the OpenGL2 graphics language. Mappings are flexible: sets of mappings may be arranged and modified during the design and rehearsal process, and triggered by control events during

[26] Available at http://www.roland.com/products/en/HPD-15/ (October 2006)

different movements of a performance, and arbitrary text, images, video, and geometric models may be used as source material.

We display a real-time composition of these media sources over geometric elements which are generated and modified according to the parameters of the current mapping. In addition to control events received from the performer, a physical simulation environment is incorporated to allow for a variety of secondary motion effects. This visually (and contextually) rich combination of source material over physically reactive structural elements allows for a response that is dynamically generated and artistically controlled. While the parameters that govern the overall response of the system to the drum controllers may be modified through cues such as MIDI program change messages, *veldt* allows an additional visual performer to control the finer aspects of the performance.

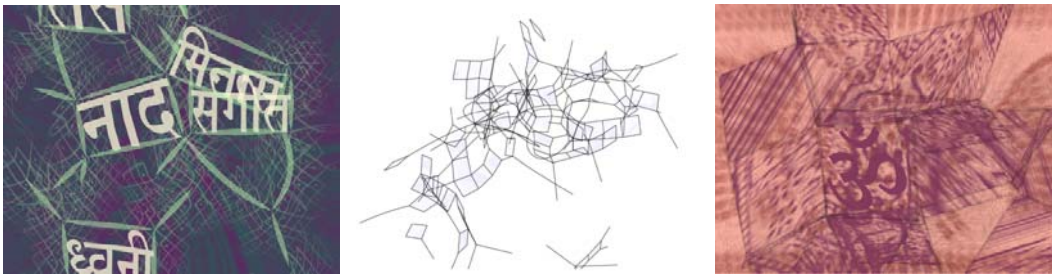


Figure 38 – (Left) Layering text elements (Hindi) over several sparse structures using *veldt*. (Middle) Example screenshot of structure evolved from a drumming sequence generated by veldt. (Right) A more cohesive structure generated by a variation on the rule set.

6.5 Summary

The promise of interactive, multi-performer, networked performances, including audience participation, has been with us for quite a long time now. New research agendas have been born to technically enable these types of performances. Some projects have begun to look at the social aspects of this area as well. This chapter served to report about specific systems, a composition, and a performance. Moreover, we asked questions as to the motivations, reasons, necessity and

validity, both artistic and aesthetic, of investing the significant time and money in order to perform in more than one place at once.

An interesting thing we discovered about networked audio/music is that it isn't as technically difficult as it has been. The recent availability of Internet2, Ca2Net and other optically based gigabit networks has made creating systems such as SoundWire and Gigapop rather simple. A good programmer with a standard networking textbook could implement our system. Performance system tweaking required quite a bit of experimentation, but when it came down to the performance itself, it worked fine. If it didn't work, the failure would have been because some astronomer decided to ftp a terabyte of data, or the dining hall closing at some university between Princeton and McGill prompting 200 students to suddenly rush back to their dorm rooms and start downloading movies using BitTorrent, or some similar reason. The promise of guaranteed quality standards on our networks went away with the demise of ATM (in the US and Canada), so it seems that we are 'stuck' with very high bandwidth, but no guarantees against porridge.

One aspect of future systems, especially those based on our existing infrastructures, might include components of handshaking, where the multiple sites each announce and probe each other as to the available capabilities. In this way, networked audio might behave much as instant messaging, where each site gives and receives what it can technically to the performance. Some sites might only send gestural/sensor data, minimal audio, and very low quality (or no) video, and synthesize a local audio performance based on minimal data from the other sites. Others might be able to provide and consume full-bandwidth uncompressed video, audio, and sensor data. The aesthetic issues surrounding these sorts of inhomogeneous, highly asymmetric systems are quite interesting for future research and study.

6.5.1 Good things about networked music performance

There are some good aspects to doing research in real-time networked sound, one of them being that sound is an excellent test-bed for testing network hardware and software. Latency and continuous quality of service is more important for sound than even for video. We can all tolerate a dropped or repeated video frame now and then, but not choppy audio. So in this way, sound and music are good for networking research and systems building, but this does not imply that networking is good for sound and music (except perhaps for funding opportunities).

Areas that will clearly become useful, once systems become commonplace and affordable, include applications in pedagogy, such as remote instruction, rehearsal, etc. The ability to rehearse remotely is also interesting for professionals in some cases. There are many cases of unique instruments that cannot be moved easily. Opting to do a rehearsal remotely from a studio (or one's home) rather than flying to the 'gig' and back, might be attractive, if the quality is good enough. Another aspect that Tanaka and others have mentioned is that the network and systems will breed new aesthetics, so new forms of art and interaction that don't fit the traditional molds of performance, improvisatory, audience, etc. might emerge, giving networked performance its own unique space in art.

6.5.2 Not so good things about networked music performance

Technically, as we have stated, existing networks do not provide guarantees of quality (delay or bandwidth), and we are fairly certain that for some time to come, any such guarantees would be very expensive to have if available. Internet2/Ca2Net are expensive themselves, and available only to academics with lots of serious research to do. To think that the physics department will buy the

music department a new gigabit router, and pay to rewire the concert halls with fibre, seems like pipe dreaming. Thus, expense is still a serious issue for all but a few.

One concern we have is the loss of the identity of the ‘band’ itself; that is, the loss of interaction of a finite number of players, each with their unique role, playing together on a single stage. Of course this is to be considered a ‘feature’ as well as a potential bug, but it is cause for concern, given the long history of musical performance in the more traditional molds.

This tradition provides important grounding for audiences, who should also be considered in the future of new music performance. Contemporary composers and musicians have historically inflicted quite a bit of grief on their audiences [6]. In this tradition, we suppose that having a robot playing an instrument on stage in a concert location, along with grainy video of human players in a remote location, could be amusing or even aesthetically pleasing. But once the novelty has worn off, the music and performance must stand on its own.

Related to this is the loss of society within the ‘band’ - that is, the interactions that go on between band members, both on and off stage. Waiting backstage to go on, and important aspects of socialization after a performance are not the same over a network. Being able to go out for a drink in Bombay after a performance can be more important and memorable than the actual performance itself. And, that drink and performance in Bombay can make the long airplane flight worth it as well.

6.5.3 A dream worth dreaming

Networked Media is a dream worth dreaming. The work completed by researchers so far comprises steps in the right direction on a path to a very uncertain destination. GIGAPOPR, the *EDholak*, and *veldt* are small pieces of a much

bigger puzzle. Applications must be constructed, and allowed to evolve naturally, that can take advantage of the ‘sound without space’.

Someday, musicians like Ustad Ashish Khan, Ustad Shahid Parvez or Ustad Zakir Hussain might be faced with a common decision of whether to sit at home in their fuzzy pajamas and play concerts with others, or to travel to the site of the performance. The author and his collaborators wonder if networked performances will be granted an artistic status as legitimate as more traditional musical endeavors.

Chapter 7

7 The Electronic Sitar

What can you do with sensors on a Sitar?

Sitar is *Saraswati's* (the Hindu Goddess of Music) 19-stringed, gourd shelled, traditional North Indian instrument. Its bulbous gourd (shown in Figure 39), cut flat on the top, is joined to a long-necked, hollowed, concave stem that stretches three feet long and three inches wide. The typical sitar contains seven strings on the upper bridge, and twelve sympathetic strings below, all tuned by tuning pegs. The upper strings include rhythm and drone strings, known as *chikari*. Melodies, which are primarily performed on one of the upper-most strings, induce sympathetic resonant vibrations in the corresponding sympathetic strings below. The sitar can have up to 22 moveable frets, tuned to the notes of a *Raga* (the melodic mode, scale, order, and rules of a particular piece of Indian classical music) [112, 184]. The sitar is a very sophisticated and subtle instrument, that can create vocal effects with incredible depths of feeling, making it a challenging digital controller to create.

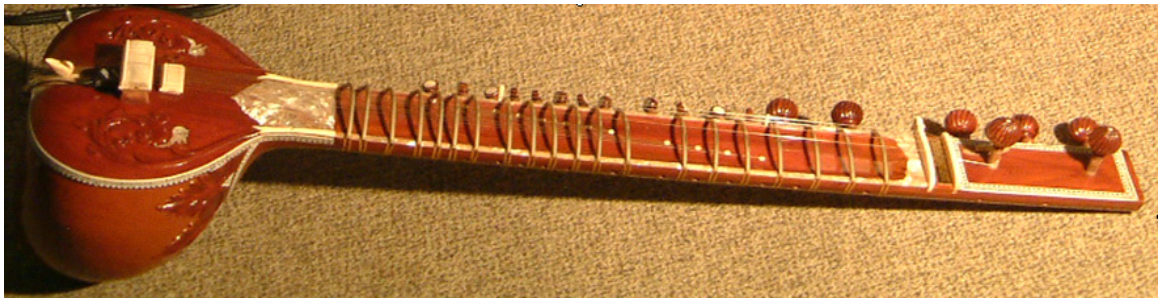


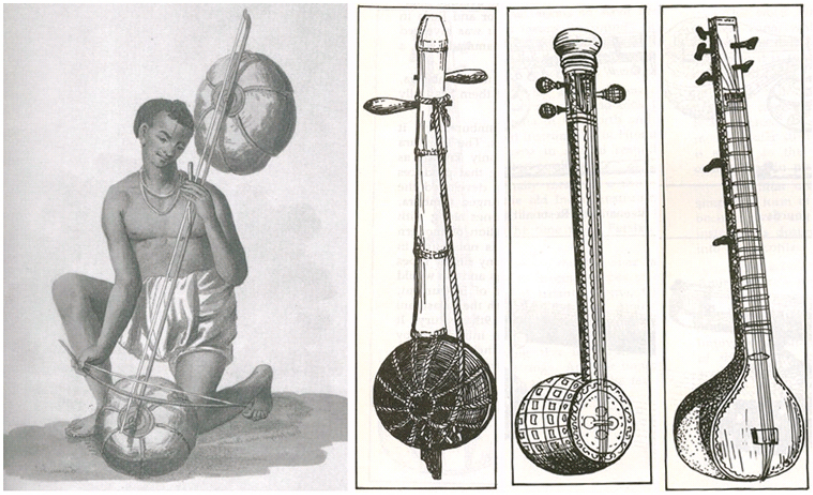
Figure 39 - A traditional Sitar.

The remainder of this chapter will present:

- The evolution of the technology of the sitar from its origins until the present day.
- The traditional playing style of the sitar, on which the controller is modeled.
- The creation of a real-time *ESitar* controller, using force sensors, accelerometers and resistor networks.
- The creation of a real-time graphical feedback system that reacts to the sitar controller.

7.1 Evolution of the Sitar

The precursor of the sitar is known as the *vina*, of the lute family of instruments, which is referenced in Vedic writings as early as the first millennium B.C. Figure 40 (a) shows an early version of the s*tick zither vina*, from the 6th and 7th century A.D. From this picture it is evident that the *stick zither* did not have frets, which ancient sculptures suggest evolved in the 10th and 11th Century A.D [7]. Figure 40 (b) shows a primitive type of *vina* instrument whose neck is made out of bamboo [157].



(a) (b) (c) (d)

Figure 40 - (a) A stick zither vina [7], (b) A vina made of bamboo [157], (c) A sehtar [157], (d) A 7-stringed sitar [157].

There exist several differing historical accounts of the sitar's evolution. Some sources claim the instrument descended directly from the *vina* as performers and builders made small modifications over time as technology and tools evolved. Others claim the similarity between the Middle-Eastern *tambur* and the Persian s*ehtar*, which traveled to

India during the Muslim occupation of India in 11th century. The name seems to have derived from the Persian *sehtar* (*she* - three, *tar* – strings) shown in Figure 40 (c). In the 18th century, instrumentalist Amir Khusro is credited with adapting the name, as well as reversing the order of the strings, placing the main melody string to the far outside, thus making it easier for the performer to play with the instrument upright [7]. He also improved the sitar by making the frets movable (for fine tuning), by using string to tie the frets down [157].

In the 18th century, after the innovation of creating a wider bridge, four more strings were added to the sitar, giving a total of seven strings (as seen in Figure 40 (d)). Other improvements include the introduction of metal frets and the *mizrab,* the pyramid-shaped, wire plectrum. In the 19th century, the *tarabdar* style of sitar emerged, which had nine to twelve sympathetic strings (known as *tarab)* positioned under the frets, as depicted in Figure 39 [7].

In 2003 I worked, with the help of an interdisciplinary team of collaborators, to bring the sitar into the modern era of computers, adding resistors, capacitors, force sensing resistors, microphones, and ethernet jacks to enhance the traditional technique with the use of a laptop.

7.2 Traditional Sitar Technique

It is important to understand the traditional playing style of the sitar to comprehend how our controller captures its hand gestures. In this section, we will define the different parts of the sitar, briefly explain how North Indians annotate melodic notes, and describe the basic technique of sitar playing.

7.2.1 Construction of a Sitar

The gourd section of the sitar is known as the *tumba* and plays the role of a resonating chamber. The flat piece of wood that lies on the front side of the *tumba* is known as the *tabli.* The long column that extends from the *tumba* is known as the *dand* (similar to the neck of a guitar), and is made out of the same material as the *tabli*. This part of the

instrument acts as a column resonator. Sometimes, a second *tumba* is put at the *dand* to increase resonance.

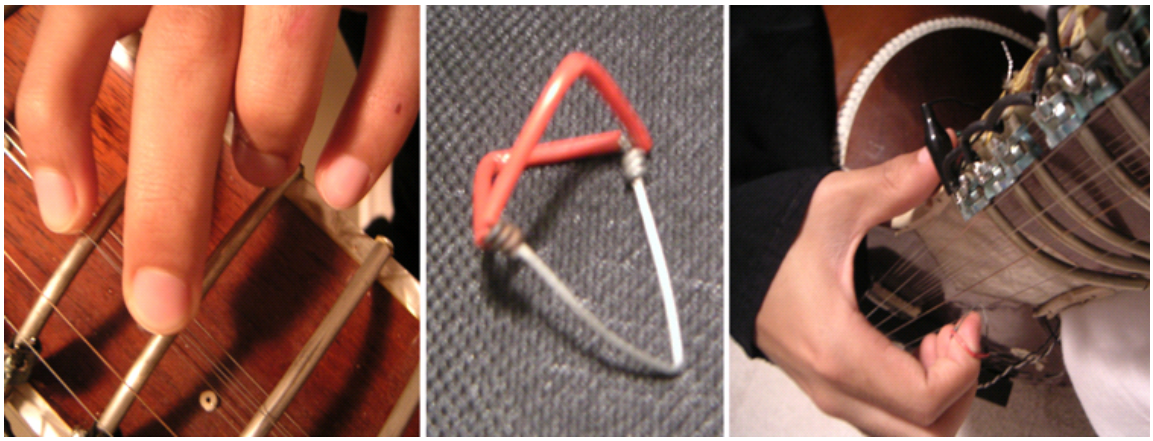
The seven main upper strings run along the *dand*, above moveable, curved metal frets, over a bridge (*jawari)* made of ivory or deer horn and are tied together at the *langot* at the very bottom of the sitar. The sympathetic strings, or *tarab* strings, run below the frets and have their own separate bridge *(ara),* but are also tied together at the *langot.* All strings are made of steel, except for the second upper string (right next to the main melody string), which is made of copper.

7.2.2 Sitar Playing Technique

It should be noted that there are two main styles of sitar technique: Ustad Vilayat Khan's system and Pandit Ravi Shankar's system. The main differences between the styles are that Ustad Vilayat Khan performs melodies on the higher octaves, eliminating the lowest string from the instrument, whereas Pandit Ravi Shankar's style has more range, and consequently melodies are performed in the lower octaves [7]. The *ESitar* is modeled on the Vilayat Khan system or *gharana*.

A performer generally sits on the floor in a cross-legged fashion. Melodies are performed primarily on the outer main string, and occasionally on the copper string. The sitar player uses his left index finger and middle finger, as shown in Figure 41(a), to press the string to the fret for the desired *swara.* In general, a pair of frets are spaced a half-step apart, with the exception of a few that are spaced by a whole step (typically around *Sa* and *Pa* – See Appendix A for a more detailed explination). The frets are elliptically curved so the string can be pulled downward, to bend to a higher note. This is how a performer incorporates the use of *shruti* (microtones).

On the right index finger, a sitar player wears a ring like plectrum, known as a *mizrab,* shown in Figure 41(b). The right hand thumb remains securely on the edge of the *dand* as shown on Figure 41(c), as the entire right hand gets pulled up and down over the main seven strings, letting the *mizrab* strum the desired melody. An upward stroke is known as *Dha* and a downward stroke is known as *Ra.[7, 184]*



(a) **(b)** **(c)**

Figure 41 - Traditional Sitar Playing Technique.

7.3 The MIDI Sitar Controllers

With the goal of capturing a wide variety of gestural input data, the *ESitar* controller combines several different families of sensing technology and signal processing methods. Two *ESitar*'s were constructed: *ESitar* 1.0 in summer of 2003 and *ESitar* 2.0 in summer of 2006. The methods used in both will be described including microcontroller platforms, different sensors systems and algorithms.

7.3.1 The Microcontroller

7.3.1.1 Atmel

The core of the *ESitar* 1.0's sensing and communication systems is an Atmel[27] AVR ATMega16 microcontroller. The microcontroller is exploited primarily for its several parallel on-board analog to digital converters [196]. As the various sensor inputs are digitized by the microcontroller we do some pre-processing of the resulting signals to clean them up and/or classify them before forwarding them on to the host computer via MIDI.

The Atmel is encased in a controller box as seen in Figure 42, with three switches, shaft encoders, and potentiometers used to trigger events, toggle between modes, and fine tune settings. The box also has an LCD to display controller data and settings to the

[27] http://www.atmel.com/ (January 2007)

performer, enabling him/her to be completely detached from the laptops running sound and graphic simulations. The sitar and headset are each connected to the main control box using ethernet-type patch cables. These cables provide a clean and robust interconnection over which the analog sensor signals are sent to the control hardware.

7.3.1.2 PIC

The new *ESitar* 2.0 made a platform change from the Atmel to the PIC[28] microcontroller, based on the mentoring of Eric Singer, Director of League of Electronic Music Urban Robots (LEMUR) in Brooklyn, New York. A major improvement was encasing the microchip, power regulation, sensor conditioning circuits, and MIDI out device in a box that fits behind the tuning pegs on the sitar itself. This reduces the number of wires, equipment, and complication needed for each performance. This box also has two potentiometers, six momentary buttons, and four push buttons for triggering and setting musical parameters.

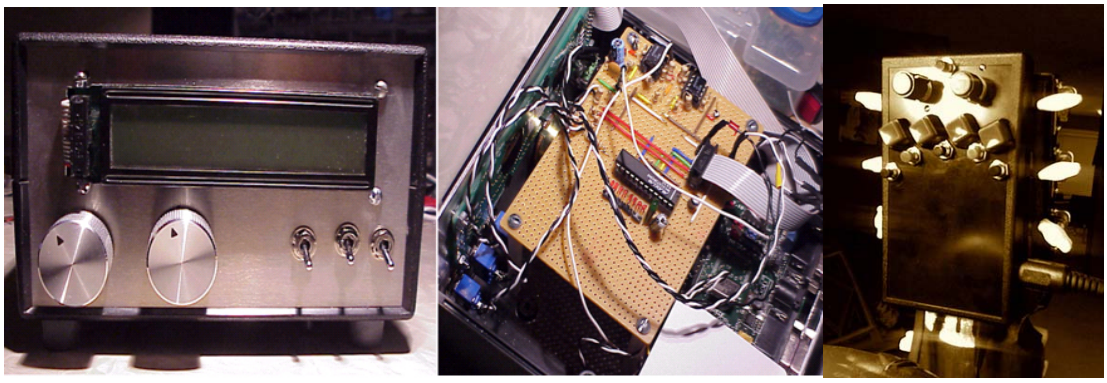


Figure 42 – Atmel Controller Box Encasement of *ESitar* 1.0 (left, middle). PIC Controller Box Encasement on *ESitar* 2.0 (right)

7.3.2 Sitar Construction Alterations

Initial experiments on the first *ESitar* were administered on a Vilayat Khan style sitar. The upgraded *ESitar* 2.0 was designed using new methods and theory obtained from three years of experience of touring and performing. The first step was to find a sitar maker in India to custom design an instrument with modifications to help encase the electronics. One major change to the traditional sitar was the move to worm-gear tuning

[28] http://www.microchip.com/ (January 2007)

pegs for the six main strings. This allows the sitar to remain in tune through all the intense bending during performance, and makes the instrument more accessible to Western music students. A second *tumba* (gourd) was also created to encase a speaker to allow for digital sound to resonate through the instrument as well as serve as a monitor for the performer. The bridge, traditionally made of ivory, and then deer bone was upgraded to black ebony wood from Africa, which generates an impressively clear sound and requires less maintenance. The frets themselves were pre-drilled to allow easy installation of the resistor network described in detail below.

7.3.3 Gesture Capturing

The controller captures gesture data including the depressed fret number, pluck time, thumb pressure, 3 axes of the performer's head tilt, and 3 axes of the sitar's tilt.

7.3.3.1 Fret Detection

The currently played fret is deduced using an exponentially distributed set of resistors which form a network interconnecting in series each of the frets on the *ESitar* (pictured in Figure 43). When the fingers of the left hand depress the string to touch a fret (as shown in Figure 41(a)), current flows through the string and the segment of the resistor network between the bottom and the played fret. The voltage drop across the in-circuit segment of the resistor network is digitized by the microcontroller. Using a lookup table it maps that value to a corresponding fret number and sends it out as a MIDI message. This design is inspired by Keith McMillan's Zeta Mirror 6 MIDI Guitar [185].

The *ESitar* 2.0 used a modified resistor network for fret detection based on more experimentation. Military grade resistors at 1% tolerance were used in this new version for more accurate results. Soldering the resistors to the pre-drilled holes in the frets provided for a more reliable connection that does not have to be re-soldered at every sound check!

As mentioned above, the performer may pull the string downward, bending a pitch to a higher note (for example play a *Pa* from the *Ga* fret). To capture this additional information that is independent of the played fret, we fitted the instrument with a piezo pick-up whose output was fed into a pitch detector. For initial experiments, the pitch

detector was implemented in a *pure data[136]* external object using an auto-correlation based method [208]. The pitch detection is bounded below by the pitch of the currently played fret and allows a range of eight semi-tones above. Further evolution of this concept using *Marsyas* is described in more detail in Chapter 12.

Figure 43 - The network of resistors on the frets of the *ESitar* 1.0 (left, middle). The *ESitar* 2.0 full body view (right).

7.3.3.2 Mizrab Pluck Direction

We are able to deduce the direction of a *mizrab* stroke using a force sensing resistor (FSR), which is placed directly under the right hand thumb, as shown in Figure 44. As mentioned before, the thumb never moves from this position while playing. However, the applied force varies based on *mizrab* stroke direction. A *Dha* stroke (upward stroke) produces more pressure on the thumb than a *Ra* stroke (downward stroke). We send a continuous stream of data from the FSR via MIDI, because this data is rhythmic in time and can be used compositionally for more then just deducing pluck direction. A force sensing resistor used to obtain thumb pressure proves to be useful in obtaining rhythmic data from the performer as will be explored in Chapter 10.

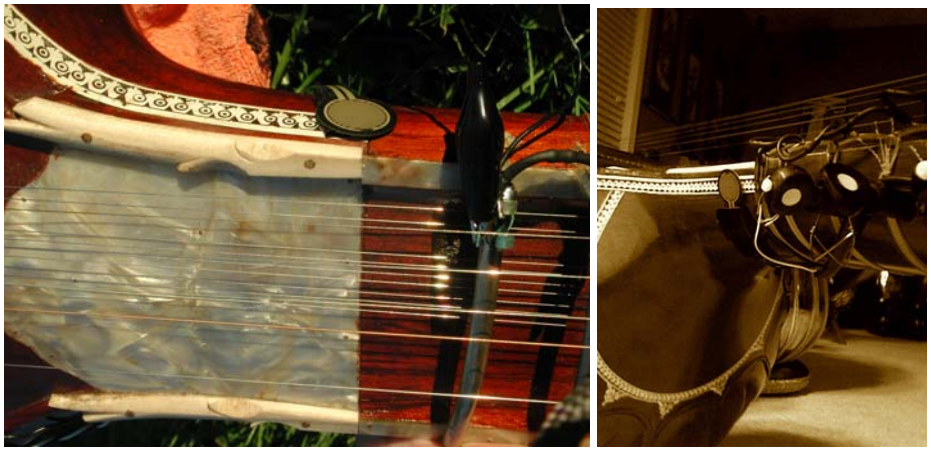


Figure 44 - FSR sensor used to measure thumb pressure on *ESitar* 1.0 (left) and *ESitar* 2.0 (right).

7.3.3.3 Mizrab Pluck Time

Pluck time is derived using two condenser microphones placed on a third bridge above the *ara* (shown in Figure 45). The microphones are located directly under the main melody string and the copper string. The signals from the microphones are passed through an analog envelope detector to extract the pluck time. We also use these microphones to determine on which string the melody is being played. If the melody is being played on the copper string (which is very rare), we can safely assume that the main melody string is not being played. The microcontroller sends a MIDI message when a pluck occurs, embedding the information for the string that was plucked.

Figure 45 - Gesture capturing sensors at base of *ESitar* 1.0.

7.3.3.4 3-axes Sitar Tilt

In the *ESitar* 2.0, there is a 3-axis accelerometer embedded in the controller box at the top of the neck, to capture ancillary sitar movement, as well as serve as yet another means to control synthesis and audio effect parameters. This sensor can be used to derive data for performer's posture with their instrument, as well as intricacies about playing technique such as jerk detection to help evaluate the beginning and end of melodic phrasing.

7.3.3.5 3-Axes Performers Head Tilt

An accelerometer is attached to a headset (as shown in Figure 46) in order to obtain 3-axes of head tilt information, on the *ESitar* 1.0. We see the head as an easy way to control

and trigger different events in the performance [113]. We send continuous head data out via MIDI messages. The headset would be a useful addition to almost any controller as a replacement for foot pedals, buttons, or knobs. It is particularly useful in this system as a sitar player's hands are always busy, and cannot use his/her feet due to the seated posture. This idea inspired Chapter 8 and the system evolved to attaching wireless acceleration sensors called *WISP*s to various parts of the performer's body.

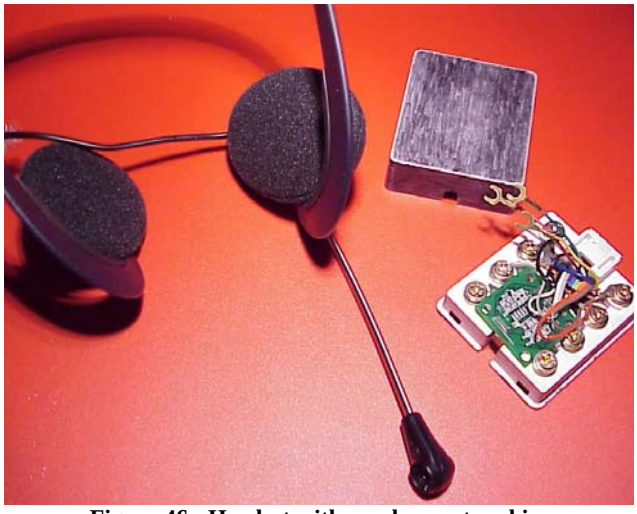


Figure 46 - Headset with accelerometer chip.

7.4 Graphic Feedback

Visualization of the *ESitar* performance were rendered again using *veldt*. Visualizations were modeled on the traditional form of melodic notation for sitar. As the player performs, the incoming note/velocity pairs were read from the MIDI signal to render a stream of *swara* (See Appendix A for more information), which are arranged in a helix as if they are printed on spinning drum of paper (shown in Figure 47 (left)). A discrete rhythm detection algorithm [48] was used over a recent history of notes played to estimate a rough beat-per-minute value, which modulates the speed at which the drum rotates so that one measure is displayed per rotation of the drum. Notes played with greater intensity are rendered in a larger, bolder style, emphasizing them within the overall display. Rhythmic patterns are reflected visually as symmetries around the drum.

Additional signals are measured in the performance to broaden the player's ability to change the scene. Signals received from two of the three axes from the tilt accelerometers on the headset are monitored, as well as the pressure measured from the thumb of the plucking hand, in order to pick up both continuous, low frequency measurements and additional rhythmic cues. The accelerometer values are used to parameterize the blending of several video streams over geometry in background of the video, in correlation with the movement through parameter spaces in our audio synthesis model. The thumb pressure provides a frequent, pulsing measurement in coordination with the plucks of the *ESitar*, and is used to control the scrubbing of short animation clips - in Figure 47 (right), those of a flower opening and closing.

There are certainly a wide range of mappings that could be conceived of with the new range of the measurements that are received from the *ESitar*. In a pedagogical setting, a pre-recorded reel of swara could be used as a score against which a student's accuracy could be measured visually, while also drawing their attention to more subtle aspects of their performance.

Figure 47 – (left) Roll of swara rendered over video stream. (right) Animation scrubbing from thumb pressure.

7.5 Summary

This chapter has presented a real-time device for sitar performance. The sitar controller captures gestural data from a performer, and uses it to manipulate sounds and visuals. A performer can now use a laptop with a sitar in order to create a multimedia experience for

a live audience, using traditional Indian classical sitar technique. Performance based visual feedback provides another means of expression for the performer, as well as a pedagogical tool. As will be shown in the later chapters of this dissertation, the *ESitar* serves as a powerful tool towards machine automated transcription of Indian Classical music.

Chapter 8

8 Wearable Sensors

Capturing data from sensors on the Human Body

The motion of the human body is a rich source of information, containing intricacies of musical performance which can aid in obtaining knowledge about intention and emotion through human interaction with an instrument. Proper posture is also important in music performance for musician sustainability and virtuosity. Building systems that could aid as pedagogical tools for training with correct posture is useful for beginners and even masters.

This chapter explores a variety of techniques for obtaining data from a performing artist by placing sensors on the human body. The sensor data is used for a variety of applications including sonification of body gestures for analysis, real-time control of synthesis and audio effect parameters, and posture feedback systems.

The first section describes experiments using a motion capture system. The second describes an evolution to using a wearable sensor package to obtain acceleration data. The third section describes yet another evolution to a wireless sensor package system that obtains orientation data. Experiments with Indian classical performers are included throughout the chapter.

8.1 Motion Capture for Musical Analysis

This section describes experiments using a motion capture system to help understand some of the intricacies of human body motion during musical performance. This research has been inspired by the work of Marcelo Wanderly at McGill University in Montreal, Canada, who uses motion capture systems to study three main factors that influence performance: (1) The instrument's constraints on the body, (2) the characteristics of the performance (e.g. rhythm, articulation, tempo, etc.) and (3) the interpretive momentary choices of the performer [187, 188]. The same team did further research to analyze the production and reproducibility of the performer's ancillary body movements [129].

The goal of this work is to build the necessary infrastructure to study the use of sonification for understanding human motion in a musical context. In order to achieve this, VICON[29], a commercial vision based motion capturing system was interfaced with various sound producing languages and frameworks. Sonification of human motion can yield results that are not observable by vision alone. Perception of periodicity, regularity, and speed of motion are a few of the attributes that are easier to observe with the aid of sound.

Although the proposed infrastructure has been applied to many areas of research, the goals relating to this dissertation include studying how a musician's posture and gestural movements during performance affect the sound produced as well as the emotional content (See Chapter 14) of the performer [87].

8.1.1 VICON Motion Capture System

The Vicon Motion Capture System is designed to track human or other movement in a room-size space. Spheres covered with reflective tape, known as markers, are placed as visual reference points on different parts of the human body. The VICON system makes use of six cameras and is designed to track and reconstruct these markers in three-dimensional space. When a marker is seen by one of the cameras, it will appear in the camera's view as a series of highly illuminated pixels in comparison to the background.

[29] http://www.vicon.com (Available January 2005)

During capture the coordinates of all the markers in each camera's view are stored in a data-station. The VICON system then links together the correct positions of each marker to form continuous trajectories, which represent the paths that each marker has taken throughout the capture and thus how the subject has moved over time. At least three of the cameras must view a marker for the point to be captured. Therefore, interpolation algorithms are applied in order to obtain continuous signals [198]. The VICON system measured the trajectories of each subject's movement in 3D space at a sampling rate of 120 Hz; However newer systems have much higher sample rates (1-2 kHz) for more precise data.

8.1.1.1 Data Collection

After motion capture trials are run using the VICON system, all data is labeled and interpolation algorithms are run to obtain continuous streams of marker positions. Next, each trial is exported to a text file. The first line of the text file contains the label names of the markers, separated by commas. Each line is time stamped and represents the x-, y-, z-, coordinates of all the markers for that particular time instance. All our experiments were captured at a 120 Hz sampling rate.

For this research, the set of data collected was on performers playing traditional instruments, namely the tabla and the violin. Tabla performance recordings were of traditional *Tin Taal Theka* excerpts of 16-beat cycles. As shown in Figure 48 (left), a full model of the right hand was captured using a custom built VICON plug-in to capture 28 marker points. The violin performances were of simple songs in moderate tempo in a major scale. As shown in Figure 48 (right), markers were placed to capture upper body movements including the head, arms, and upper clavicle.

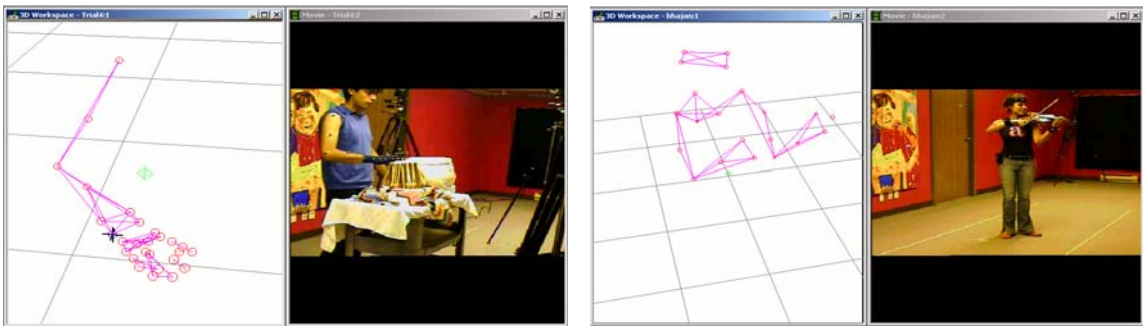


Figure 48 – (left) Screenshot of data capturing process for tabla performance. (right) Screenshot of data capturing process for violin performance.

8.1.2 Sonification Experiments

8.1.2.1 Framework

This section describes experiments on sonifying data obtained using the *VICON* motion capture system. The main goal is to build the necessary infrastructure in order to be able to map motion parameters of the human body to sound. For sonification the following three software frameworks were used: *Marsyas*[30] *[177]*, traditionally used for music information retrieval with audio analysis and synthesis, *ChucK*[31] *[189]*, an on-the-fly real-time synthesis language, and *Synthesis Toolkit*[32] *[36]*, a toolkit for sound synthesis that includes many physical models of instruments and sounds.

8.1.2.2 Importing Data

Once the data is collected, the files are imported into the desired synthesis language. For both *Marsyas* and *ChucK*, custom ViconFileSource classes were designed to read the marker sets. The motion capture data and derived features can be used to control parameters of different synthesis algorithms and digital audio effects for sonification purposes. Both languages were modular enough to allow for two streams of data, (in this case, Vicon and audio) to run at two different sampling rates.

8.1.2.3 Sonification Algorithms

Using *Marsyas* for audio analysis, feature extraction and classification, and STK for synthesis, and finally *ChucK*, a high-level language for rapid experimentation, it is possible to implement a breadth of sonification algorithms (see Figure 49).

---

[30] http://opihi.cs.uvic.ca/*Marsyas*/
[31] http://chuck.cs.princeton.edu/
[32] http://ccrma.stanford.edu/software/stk/

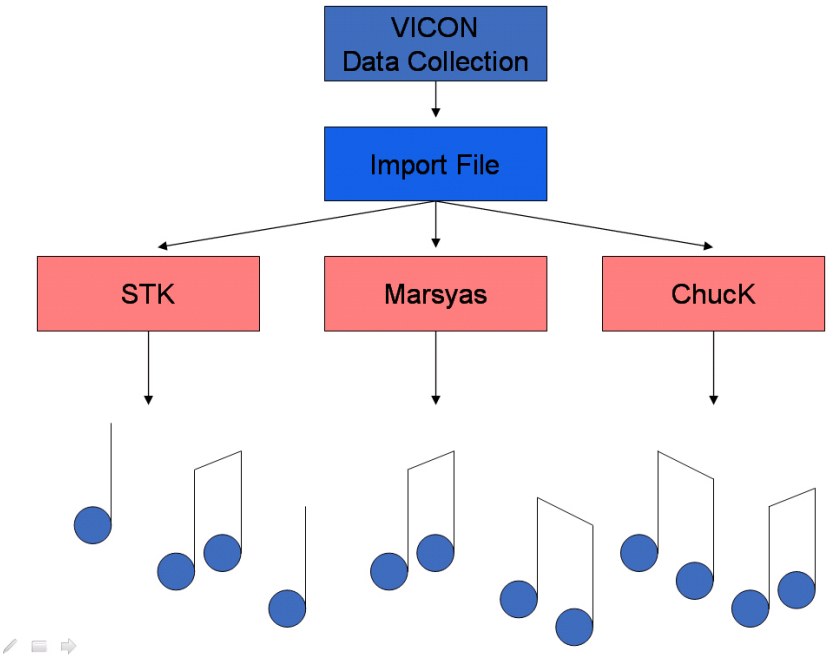


Figure 49 - Vicon Sonification Framework.

Using *Marsyas*, a simple gesture based additive synthesis module was designed, which took *n* different makers and used them to control the frequency of *n* sinusoids. The code sketch in Figure 50 shows how the 3 markers of the x,y,z wrist position can be used to control 3 sinusoidal oscillators in *Marsyas*. Another method was to use the gesture data to control the gain values of the *n* different wavetables. In order for this to work, each marker's data stream had to be normalized.

```
01# while (viconNet->getctrl("bool/notEmpty"))
02#    {
03#         // read marker data from file
04#         viconNet->process(in,out);
05#
06#         // control frequencies of sine oscillators
07#        pnet->updctrl("real/frequency1", out(1,0));
08#        pnet->updctrl("real/frequency2", out(2,0));
09#        pnet->updctrl("real/frequency3", out(3,0));
10#        // play the sound
11#         pnet->tick()
12#    }
```

Figure 50 - The following code sketch shows has the 3 markers of the x,y,z wrist Position can be used to control 3 sinusoidal oscillators in *Marsyas*.

Another technique that was easy to implement in *Marsyas* was gesture-based FM synthesis. FM synthesis is a method of creating musically interesting sounds by repetitively changing the basic frequency of a source. We set up a system to have the modulation index and source frequency change with data from the marker streams.

An example of motion controlled digital audio effect implemented in *Marsyas* is a real-time Phase Vocoder [57]. A Phase Vocoder is an algorithm for independent control of time stretching and pitch shifting. Thus the marker data streams can control the speed of the audio playback and the pitch independently of each other.

Using STK, we were able to control physical models of instruments [36]. This way we could use marker streams to control different parameters (such as tremolo rate, hardness, direction, vibrato, reed aperture, etc) on instruments including flute, clarinet, mandolin, shakers, and even sitar.

ChucK provides the ability to (1) precisely control the timing of a sonification algorithm and (2) easily factor many complex sonification algorithms and digital audio effects into concurrent modules that are clearer (and easier) to implement and reason about.

```
01#    // read single column of data from input
02#    ColumnReader r( input, column );
03#    float v;
04#
05#    // time-loop
06#    while( r.more() )
07#    {
08#        // read the next value
09#        r.nextValue() => v;
10#        // do stuff with v
11#        ...
12#        // advance time as desired
13#        T::ms => now;
14#    }
```

Figure 51 - Example template *ChucK* code for sonification of body motion data.

In this example (Figure 51), we show a simple template used for sonifying multi-valued streams of marker data by factoring into concurrent processes - one process for each value stream (column). The template first creates a reader for a specific column (line 2). In the loop (lines 6-14), the next value is read (line 9) and used for sonification (to control synthesis, etc., line 11). Finally, time is advanced by any user-definable amount (line 13).

This template can be instantiated one or more times as concurrent processes, each with a potentially different column number, time advancement pattern, and synthesis algorithm. Together, they sonify the entire dataset, or any subset thereof. For example, one process uses a granular model to sonify column 2, and another uses a plucked string model to sonify column 5. One of the properties of *ChucK* is that all such processes,

while independent, are guaranteed to be synchronized with sample-precision. Furthermore, it is possible to add/remove/replace a process on-the-fly, without restarting the system.

8.1.2.4 Experiments with Musical Instruments

The goal in this area of study is to sonify events of the gestures of performers playing different instruments. There are numerous areas of interest that can be explored using this framework.

First, we are interested in finding which markers contain musical information. This can be tested using STK's physical models of the instrument, in order to try and reproduce a performance, using the marker's data to control parameters of appropriate physical models. Our goal is to find how few markers can be used in order to reconstruct a musical phrase. Another interesting question is to observe interchanging traditional mappings (e.g. map plucking hand to bowing, and bowing hand to plucking) to obtain new types of sound.

Another area of interest is to observe ancillary gestures during performance (e.g. how the head moves during a violin performance). Specifically, the following questions are asked: When a performer plays the same composition, do the ancillary body gestures move in the same way? What is the minimum number of markers that need to be the same in order for the same performance to be played? What type of information do the ancillary markers obtain? Answering these questions using the proposed framework allows observations of subtle differences in movements that are difficult to see using only visual feedback.

As seen in Figure 48, initial experiments are based on a tabla and violin performance. The challenge with the tabla is the precise timing of fingers. Thus we use a detailed model of the hand, as described above, in order to preserve the performance. Challenges with the violin include timing like the tabla, but also the added dimension of melody and the associated emotions expressed as movement.

8.2 The *KiOm* Wearable Sensor

The VICON Motion Capture System provides an immense amount of data for analysis and research. However, the system used is not real-time (although real-time versions of hardware/software commercially exist). VICON system and other motion capture systems are very expensive, and cumbersome if not impossible to move on stage for performance. Also, the markers which are stuck on the musicians tend to fall off and lighting conditions must meet certain requirements for ideal capture. These drawbacks have influenced the invention of the *KiOm* [88] wearable sensor.

This section describes the use of wearable sensor technology to control parameters of audio effects for real-time musical signal processing. Traditional instrument performance techniques are preserved while the system modifies the resulting sound based upon the movements of the performer. Gesture data from a performing artist is captured using three-axis accelerometer packages that is converted to MIDI (musical instrument digital interface) messages using microcontroller technology. In the literature presented, sensors are used to drive synthesis algorithms directly, completely separating the sound source from the gesture. Our paradigm, and the key novelty of this work, is to keep traditional instrument performance technique, modifying the amplified acoustic signal with sensor data controlling a number of audio effect parameters.

A similar paradigm is that of the hyperinstrument [85, 103] where an acoustic instrument is augmented with sensors. In our approach, any performer can wear a low-cost sensor while keeping the acoustic instrument unmodified, allowing a more accessible and flexible system.

8.2.1 Wearable Sensor Design

The design of the *KiOm* (see Figure 52), is described in this section. A Kionix KXM52-1050[33] three-axis accelerometer is used. The three streams of analog gesture data from the sensor is read by the internal ADC of the Microchip PIC 18F2320[34]. These streams are converted to MIDI messages for use with most musical hardware/synthesizers.

---

[33] http://www.kionix.com/ (February 2005)
[34] http://www.microchip.com/ (February 2005)

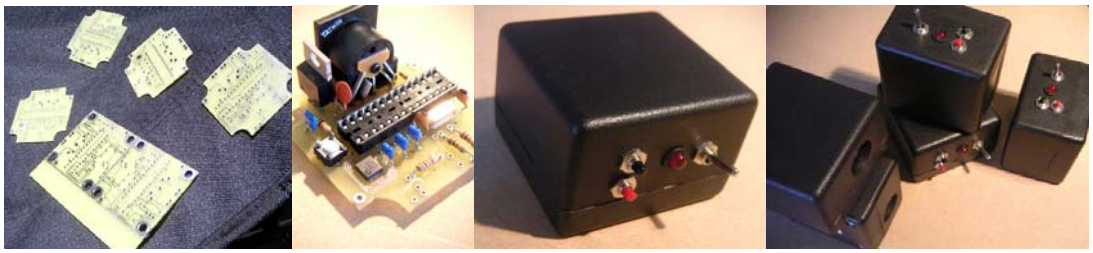


Figure 52 - The *KiOm* Circuit Boards and Encasement.

A microphone capturing the acoustic signal of the instrument is also part of the system. This way, synchronized gesture data and audio signals are captured in real-time by the system for signal processing. Figure 53 shows a diagram of our system.

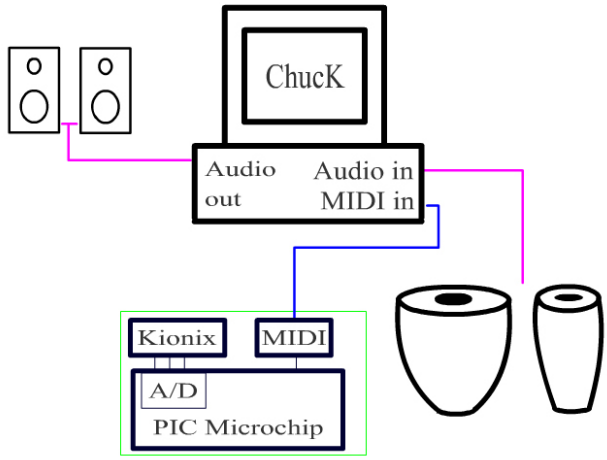


Figure 53 - Diagram of synchronized audio and gesture capture system.

8.2.2 Audio Signal Processing

This section will present how *ChucK* is used to for audio signal processing using both streams of data. The different synthesis algorithms that are used for experimentation are also described.

8.2.2.1 Synthesis Algorithms

For the initial experiments, a number of traditional synthesis algorithms and digital audio effect processors were implemented.

The first algorithm was a FIR comb filter. A FIR comb filter adds a delayed version of the input signal with its present input signal. There are two parameters to tune

the filter: *T* that is the amount of delay, and the *g* the amplitude of the delayed signal. The difference equation is given by [208]:

$$y(n) = x(n) + gx(n - M)$$
$$M = T / f_s$$
$$H(z) = 1 + gz^{-M}$$

The acceleration data from the wearable sensor can be used to control values of *T* and *g* on the acoustic signal *x(n)*.

Vibrato [208] is an algorithm which periodically varies the time delay at a low rate to produce a periodic pitch modulation. Typical values are 5 to 10 ms for average delay time, and 5 to 14 Hz for the low-frequency oscillator, parameters which two axes of acceleration from the *KiOm* control.

When a comb filter is combined with a modulating delayline, flanger, chorus, slapback and echo effects are produced. If an FIR comb filter and a delay between 10 and 25 ms are used, a doubling effect known as slapback occurs. If an FIR filter with a delay greater then 50 ms is used, an echo is heard. If the delay time is continuously varied between 0 and 15 ms, an effect known as flanging occurs. If the delay line is varied sporadically between 10 and 25 ms, a chorus effect occurs [208]. The *KiOm* is used to control parameters to all these different algorithms.

8.2.3 Case Studies

This section describes different experiments with a variety of instruments to show the versatility and evolution of our system. The *KiOm* is performed with a variety of instruments both Indian classical and Western.

Figure 54 shows our first experiments with a drum set performance. The wearable prototype sensor was placed on the hands of the drummer who was told to play with traditional technique. Because of the rhythmic nature and movement of the drummer's hands during the performance, using the gesture-captured data to affect the sounds of the drums was successful. Our favorite algorithms were controlling parameters of the comb filters and the flanger. Similar results were obtained by placing the sensors on the feet of the drummer while playing bass drum.

Our next experiment was with hand drumming on the traditional North Indian Tabla as seen on the left of Figure 55. Again, a traditional performance obtained rhythmic gesture capture data which musically combined as parameters to the various synthesis algorithms. Another method was to place the sensor on the head of a performer, as shown on the right of Figure 55. Here it is attached to a headset (headphones with boom microphone) so that the Indian vocalist or any other performer can sing and control the DSP parameters with head motions, thus leaving the hands and feet free to gesture to the audience. Another method is for the performer to play a traditional instrument wearing the headset, replacing the need for foot-pedals, knobs and buttons to control synthesis parameters. An example where this might be useful is during Sitar performance, in which the musician traditionally sits on the floor, and whose hands are occupied, leaving only the head to control parameterization. This was the initial experiment administered described in [85], which initiated this research.

Figure 54 – Wearable sensors used for a drum set performance.

Figure 55 - Wearable sensor used for a Tabla performance (left). Set up for using on head of performer (right).

Figure 56 - Wearable sensor used to capture scratching gesture of turntablist (left). Wearable sensor used in conjunction with live Computer Music Performance using *ChucK* (right)..

More experiments include performances with a turntablist who was scratching vinyl records with the *KiOm* placed on the hand (Figure 56 (left)), similar to the drum experiments. Experiments on a computer music performance were administered, in which a performer used a keyboard and mouse of a laptop, with a *KiOm* to capture gestures to control parameters of synthesis algorithms as shown on the right of Figure 56.

8.3 The *WISP* Wearable Sensors

The Wireless Inertial Sensor Package (*WISP*) [170], designed by Bernie Till and the Assistive Technology Team at University of Victoria, is a miniature Inertial Measurement Unit (IMU) specifically designed for the task of capturing human body movements. It can equally well be used to measure the spatial orientation of any kind of object to which it may be attached. Thus the data from the *WISP* provides an intuitive method to gather data from a musical performer. The *KiOm*'s have the disadvantage of being heavy and having wires that connect to the computer, certainly putting constraints on a musician. With the wireless *WISP*, the performer is free to move within a radius of about 50m with no other restrictions imposed by the technology such as weight or wiring.

8.3.1 Hardware Design

The *WISP* is a highly integrated IMU with on-board DSP and radio communication resources. It consists of a triaxial differential capacitance accelerometer, a triaxial magnetoresistive bridge magnetometer, a pair of biaxial vibrating mass coriolis-type rate gyros, and a NTC thermistor. This permits temperature-compensated measurements of

linear acceleration, orientation, and angular velocity. The first generation prototype of *WISP*, shown in Figure 57 next to a Canadian two-dollar coin, uses a 900 MHz transceiver with a 50Kb/s data rate. With a volume of less than 13cm^3 and a mass of less than 23g, including battery, the unit is about the size of a largish wrist watch. The *WISP* can operate for over 17 hours on a single 3.6V rechargeable Lithium cell, which accounts for over 50% of the volume and over 75% of the mass of the unit.

The fundamental difference between the *WISP* and comparable commercial products is that the *WISP* is completely untethered (the unit is wireless and rechargeable) in addition to being far less expensive. All comparable commercial products cost thousands of dollars per node, require an external power supply, and are wired. A wireless communication option is available in most cases, but as a separate box which the sensor nodes plug into. As can be seen in Figure 57, the small size and flat form-factor make it ideal for unobtrusive, live and on-stage, real-time motion capture.

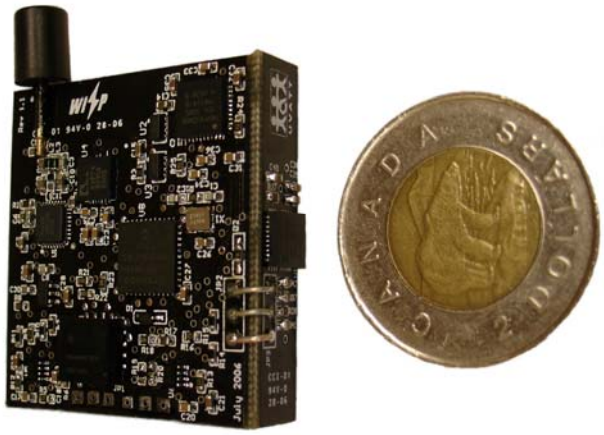


Figure 57 - Wireless Inertial Sensor Package (*WISP*)

8.3.2 The Data

The windows-based *WISP* application sends out the roll (rotation about x), pitch (rotation about y) and yaw (rotation about z) angles from the sensing unit over Open Sound Control (OSC) [199]. These angles are commonly used in aerospace literature to describe, for example, the orientation of an aircraft [110]. The data is read into *MAX/MSP* using the standard OSC reading objects. By subtracting successive samples of each orientation

angle, measures of angular velocity are obtained in addition to the raw orientation angles. With three angles and three angular velocities we have a total of six control parameters for each *WISP*. The data are then conditioned and transformed into MIDI control change messages which are sent to an audio synthesis engine.

The data from the *WISP* are received by *MAX/MSP* via the OSC protocol and converted into MIDI messages to communicate with a synthesis engine. The *WISP* is used to control audio and visual aspects of a live performance which, in turn, feed back to influence the emotional and physiological state of the performer allowing the performance to evolve in a natural self-organizing dynamic.

8.3.3 Real-Time Posture Feedback

Orientation data of the *WISP*, communicated via the OSC protocol [170], was used to give feedback to the tabla student to help them maintain correct right arm posture during playing. In the beginner stages of tabla, body posture, particularly orientation of the right arm is critical to the correct development of a student. Boundaries were imposed on the three axes of orientation given by the *WISP*. When any of the boundaries were exceeded by the student an axis specific sound would alert the student of incorrect posture.

Initial work also includes an animated human body model instrumented by seven *WISP*s attached to a student's upper body at key skeletal articulation points. Since each *WISP* is small and wireless this sort of whole-body gesture analysis is easy to implement and non-invasive for the student. A multi-*WISP* system will provide a much more detailed analysis of posture enabling the system to teach a student to maintain correct posture of the spine, neck, head, and left arm as well as the right arm. Of course this system can be extended beyond the tabla into any sort of posture critical applications.

8.4 Summary

This chapter showed comparisons between three different methods of capturing gestures by placing sensors on the human body of a North Indian classical performer. In each case the captured data were used in different applications including sonification, audio effect

control, real-time tempo tracking and performer/student posture training. Refer to Figure 58 for a summary of advantages and disadvantages of each method. Overall, our experiments show that gesture data from a musician as well as audio data is paramount in the evolution of musical signal processing.

Device	**Advantages**	**Disadvantages**
VICON Motion Capture	* Rich Amounts of Data * No Electronics on Body	* Expensive * Cumbersome for Stage * Markers fall off * Reflection Errors
KiOm	* Cheap * Accesible * MIDI Device Compatible	* Electronics on Body * Wired * Acceleration data only
WISP	* Orientation data * Wireless * OSC Device Compatible	* Expensive * Electronics on Body

Figure 58 - Comparison of Acquisition Methods

Section III

Musical Robotics

Chapter 9

9 The *MahaDeviBot*

A Comparison of Solenoid-Based Strategies for Robotic Drumming

Mechanical systems for musical expression have developed since the 19th Century. Before the phonogram, player pianos and other automated devices were the only means of listening to compositions, without the presence of live musicians. The invention of audio recording tools eliminated the necessity and progression of these types of instruments. In modern times, with the invention of the microcontroller and inexpensive electronic actuators, mechanical music is being revisited by many scholars and artists.

Musical robots have come at a time when tape pieces and laptop performances have left some in the computer music audiences wanting more interaction and physical movement from the performers [153]. The research in developing new interfaces for musical expression continues to bloom as the community is now beginning to focus on how actuators can be used to raise the bar even higher, creating new mediums for creative expression. Robotic systems can perform tasks not achievable by a human musician. The sound of a bell being struck on stage with its acoustic resonances with the concert hall can never be replaced by speakers, no matter how many directions they point. The use of

robotic systems as pedagogical implements is also proving to be significant. Indian classical students practice to a Tabla box with pre-recorded drum loops. The use of robotic strikers, performing real acoustic drums gives the students a more realistic paradigm for concentrated rehearsal.

A number of different drumming robots have been designed in the academic and artistic communities as described in Chapter 3. The drumming robots presented have all been one of a kind proof of concept systems and there hasn't been much work in qualitative comparative evaluation of different designs. Our goal in this chapter is to explore systems that can be used in the classroom to teach musical robotics. Therefore, we choose to focus on solenoid-based designs as hydraulic-based designs have prohibitive cost for classroom use. The designs presented are practical and can be replicated in a semester. The evaluation methods presented are important to inform composers and designers about strengths and limitations of different designs to guide composition decisions and performance constraints.

The development of the *MahaDeviBot* as a paradigm for various types of solenoid-based robotic drumming is described. The *MahaDeviBot* serves as a mechanical musical instrument which extends North Indian musical performance scenarios, while serving as a pedagogical tool to keep time and help portray complex rhythmic cycles to novice performers in a way that no audio speakers can ever emulate. The first section of this chapter describes the design strategies for the *MahaDeviBot*, including five different methods for using solenoids for rhythmic events. Next, the experimental evaluation of speed and dynamic testing of the different design methods is presented. A summary and discussion of results conclude the chapter.

9.1 Design

Different solenoid-based designs for robotic drumming are evaluated in the context of *MahaDeviBot,* a 12 armed robotic drummer that performs instruments from India including frame drums, bells, and shakers. Four different methods for solenoid-based drumming are described. A robotic head of *MahaDeviBot* is also described. Finally we present a piezo-based haptic feedback system for evaluation experiments and the machine's awareness of its own parts.

9.1.1 Arms

There are four different designs proposed, and appropriately named by the inventor: Kapur Fingers, Singer Hammer, Trimpin Hammer and Trimpin BellHop are described.

9.1.1.1 Kapur Fingers

The Kapur Fingers involve modifications of a push solenoid. One issue with the off-the-shelf versions of the solenoids is that during use they click against themselves making lots of mechanical sound. A key goal for a successful music robotic system is to reduce the noise of its parts so it does not interfere with the desired musical sound. Thus the push solenoids were reconfigured to reduce noise. The shaft and inner tubing were buffed with a wire spinning mesh using a Dremel. Then protective foam was placed toward the top of the shaft to stop downward bounce clicking. Rubber grommets were attached in order to prevent upward bounce-back clicking. The grommets were also used to simulate the softness of the human skin when striking the drum as well as to protect the drum skin.

Figure 59 - Kapur Finger using a grommet and padding.

9.1.1.2 Singer Hammer

The Singer Hammer is a modified version of the Eric Singer's ModBot [159]. The mechanism strikes a drum using a steel rod and ball. A pull solenoid is used to lift a block to which the rod is attached. A ball joint system was added to connect the solenoid to the bar for security and reliability of strokes. The trade-off was that it added some mechanical noise to the system. The *MahaDeviBot* has four Singer Hammers striking a variety of frame drums.

Figure 60 - Singer Hammer with added ball-joint striking mechanism.

9.1.1.3 Trimpin Hammer

The Trimpin Hammer is a modified version of Trimpin's variety of percussion instruments invented over the last 20 years [174]. Its key parts include female and male rod ends, and shaft collars. This is a very robust system which involves

using a lathe to tap the shaft of the solenoid so a male rod end can be secured. This is a mechanically quiet device, especially with the added plastic stopper to catch the hammer on the recoil. These devices are used to strike frame drums, gongs, and even bells as shown in Figure 62.

Figure 61 - Trimpin Hammer modified to fit the *MahaDeviBot* schematic.

Figure 62 - Trimpin Hammers use on *MahaDeviBot.*

9.1.1.4 Trimpin BellHop

The Trimpin BellHop is a modified version of technology designed for Trimpin's ColoninPurple, where thirty such devices were used to perform modified xylophones suspended from the ceiling of a gallery. These are made by modifying a pull solenoid by extending the inner tubing so that the shaft can be flipped upside down and triggered to hop out of the front edge and strike a xylophone or Indian bell (as shown in Figure 64). These, too, are mechanically quiet and robust.

Figure 63 - Trimpin BellHop outside shell tubing (left) and inside extended tubing (right).

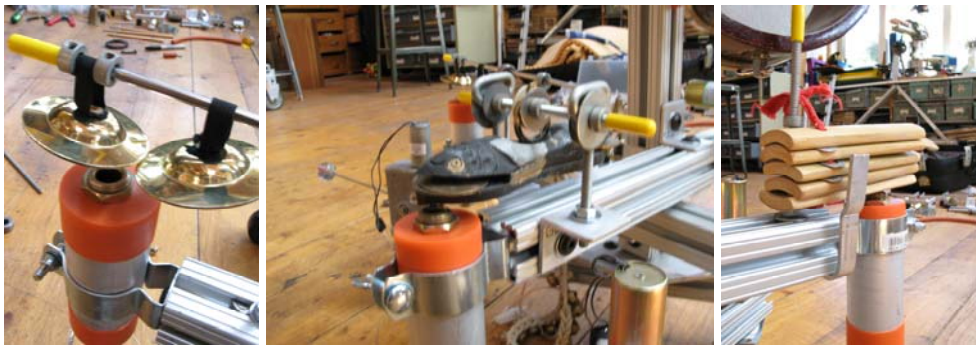


Figure 64 - Trimpin BellHops used on *MahaDeviBot.*

9.1.2 Head

The headpiece of the *MahaDeviBot* is a robotic head that can bounce up and down at a given tempo. This was made using a pull solenoid attached to a pipe. Two masks are attached to either side and the brain is visualized by recycled computer parts from ten-year old machines which have no use in our laboratories anymore. In performance with a human musician, the head serves as a visual feedback cue to inform the human of the machine-perceived tempo at a given instance in time.

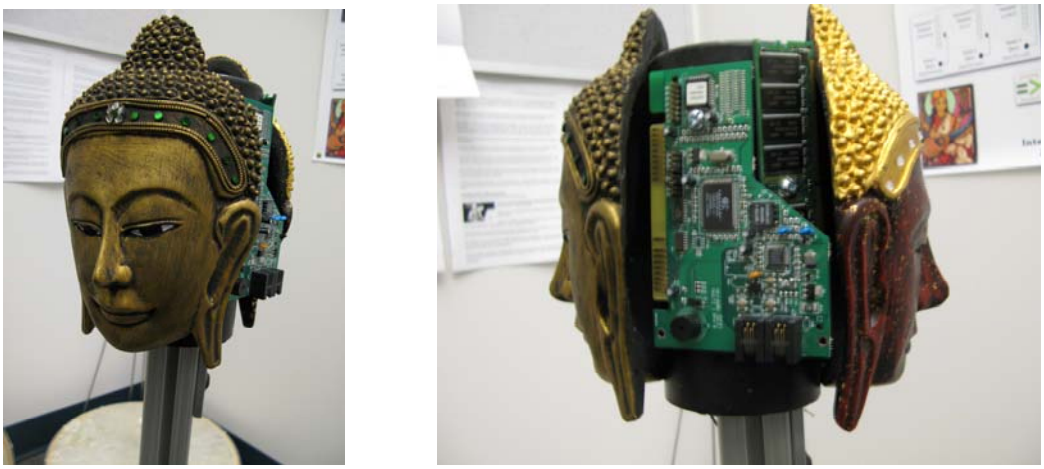


Figure 65 - The bouncing head of *MahaDeviBot*.

9.1.3 Haptic Feedback System

A haptic feedback system was implemented using piezo sensors attached to the frame drums and other instruments. This was to infuse the system with machine "self awareness", i.e. to know about the capabilities and limitations of its own implements. If the machine triggers a robotic drum strike and the piezo does not receive a signal, then it knows to increase the volume for the next strike. If it reaches its maximum volume ability and still no signal is received, then it knows that the mechanism is malfunctioning. This will trigger the machine to shut off and disable itself from further action during a performance to save power and reduce unnecessary mechanical noise. This feedback system is also used for the evaluation experiments described in the following section.

9.2 Experimental Evaluation

9.2.1 Speed Tests

Speed tests were administered to each type of solenoid-based system using the *ChucK* [189] strongly-timed programming language. The frequency of successive strikes began at 1 Hz and was incremented by .01 Hz until it was observed that

the mechanism was malfunctioning. The maximum speeds obtained by each device are portrayed in Figure 66.

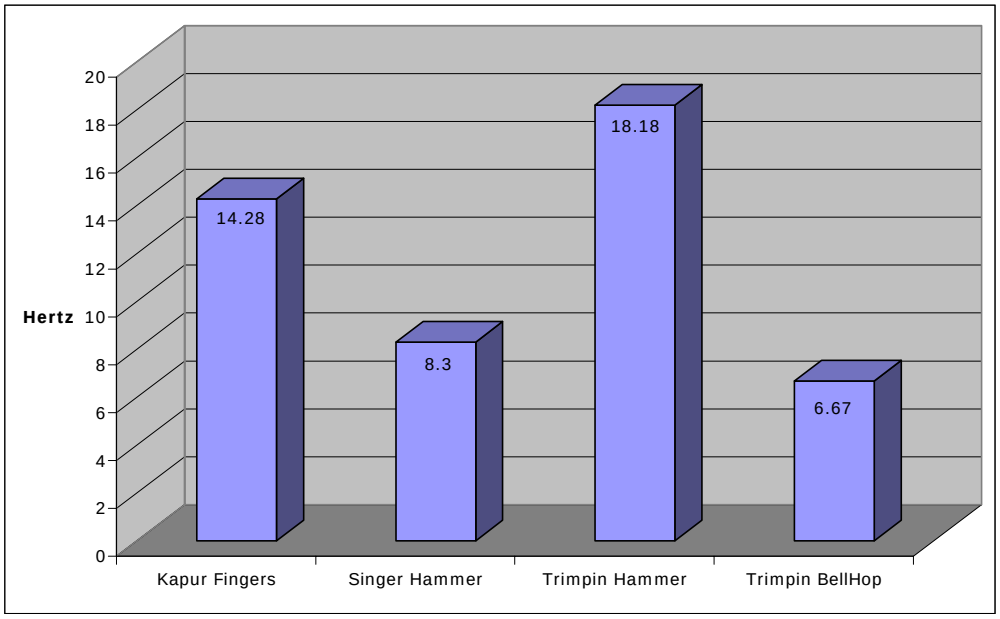


Figure 66 - Maximum speeds attainable by each robotic device.

9.2.2 Dynamic Range Tests

Dynamic range experimentation was administered by triggering robotic strikes with increasing strength using MIDI velocity messages ranging from 1 to 127. The piezo sensors placed on the drums measure the actual response for each dynamic level. Results are shown in Figure 67.

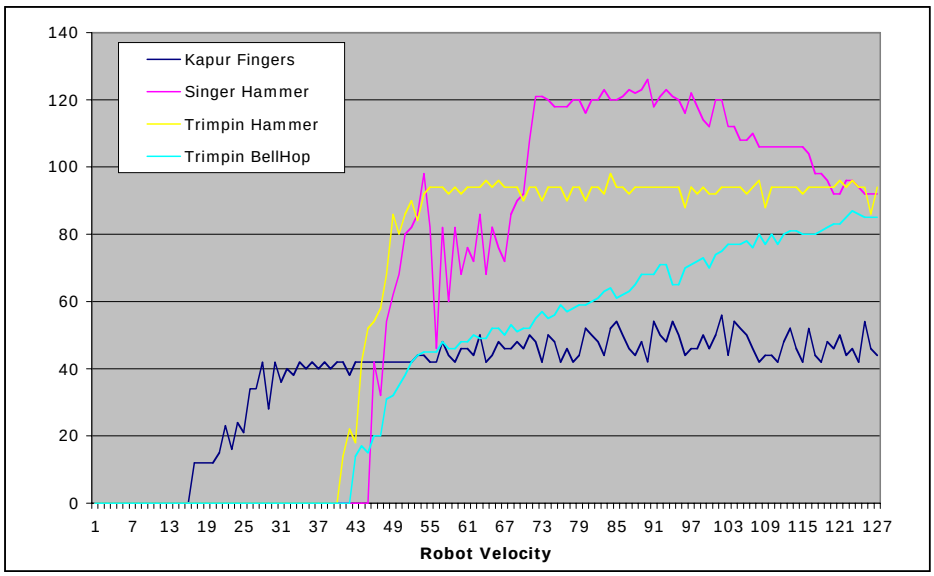


Figure 67 - Dynamic Range Testing Results.

9.2.3 Discussion

These experiments show that each design has different strengths and weaknesses. The Kapur Finger has moderately high speed capability reaching up to 14.28 Hz. However, it has limited dynamic range and cannot strike very loud. The Singer Hammer can strike very soft and very loud, but can only play as fast as 8.3 Hz. The Trimpin Hammer can roll at 18.18 Hz with only one finger, but does not have the dynamic capabilities seen in the Singer Hammer. The Trimpin Bellhop has the most linear dynamic response but is the slowest design.

9.3 Summary

This chapter described our methodology for designing a mechanical system to perform Indian Classical music using traditional folk instruments. Figure 68 shows an evolution of the design from wooden frameworks with Kapur Finger's striking a *Bayan*, to a 3-armed prototype system, to the complete 12-armed *MahaDeviBot* to-date. This chapter describes in detail the various design

strategies used to build the final version of the robotic Indian drummer. Even though some of the design tradeoffs were expected, the quantitative evaluation included provides more concrete and solid information. As an example of how these tradeoffs can influence robotic design for musical performance, the four designs are integrated into *MahaDeviBot* in the following ways: The Kapur Fingers are added to a drum with the Singer Hammer to allow large dynamic range and quick rolls from one frame drum. The Trimpin Hammer is used to perform drum rolls and is used for robotic Tabla performance. The Trimpin BellHop is used to strike bells and other instruments where volume is important and which will not be struck at high rates.

Figure 68 - Evolution of *MahaDeviBot* from wooden frames to the sleek 12-armed percussion playing device.

Future work includes making a completely automated framework in *ChucK* to evaluate robotic systems. We are also interested in designing mechanisms to allow the robot to strike at any x-, y- coordinate location. The next evolution of *MahaDeviBot* will include the use of other actuators including motors and gears.

Robotics combined with Indian Classical music presents a new paradigm for extending traditional ideas with the use of computers. The pedagogical applications upgrade the present audio speaker systems used to teach rhythmic cycles to the state-of-the-art mechanical systems which a student can practice with for a more realistic and concentrated rehearsal. A solo melodic artist can now

tour the world with a robotic drummer to accompany if software is “intelligent” enough to keep the interest of the audience. The next five chapters discuss our pursuits in this direction.

Section IV

Machine Musicianship

Chapter 10

10 Tempo Tracking Experiments

Where's the beat?

The "intelligence" of interactive multimedia systems of the future will rely on capturing data from humans using multimodal systems incorporating a variety of environmental sensors. Research on obtaining accurate perception about human action is crucial in building "intelligent" machine response. This chapter describes experiments testing the accuracy of machine perception in the context of music performance. The goal of this work is to develop an effective system for human-robot music interaction.

Conducting these types of experiments in the realm of music is obviously challenging, but fascinating at the same time. This is facilitated by the fact that music is a language with traditional rules, which must be obeyed to constrain a machine's response. Therefore the evaluation of successful algorithms by scientists and engineers is feasible. More importantly, it is possible to extend the number crunching into a cultural exhibition, building a system that contains a novel form of artistic expression, which can be used on stage.

More specifically, this chapter describes a multimodal sensor capturing system for traditional sitar performance. As described in Chapter 7, sensors for

extracting performance information are placed on the instrument. In addition wearable sensors are placed on the human performer, as described in Chapter 8. A robotic drummer has been built to accompany the sitar player, as described in Chapter 9. In this research, we ask the question: How does one make a robot perform in tempo with the human sitar player?

Analysis of accuracy of various methods of achieving this goal is presented. For each signal (sensors and audio) we extract onsets that are subsequently processed by Kalman filtering [17] for tempo tracking [89]. Late fusion of the tempo estimates is shown to be superior to using each signal individually. The final result is a real-time system with a robotic drummer changing tempo with the sitar performer in real-time.

The goal of this chapter is to improve tempo tracking in human-machine interaction. Tempo is one of the most important elements of music performance and there has been extensive work in automatic tempo tracking on audio signals [63]. We extend this work by incorporating information from sensors in addition to the audio signal. Without effective real-time tempo tracking, human-machine performance has to rely on a fixed beat, making it sound dry and artificial. The area of machine musicianship is the computer music communities' term for machine perception described in Chapter 4. Our system evolves the state-of-the-art different as it involves a multimodal sensor design to obtain improved accuracy for machine perception.

Section 10.1 presents the experimental procedure administered including details about the sensor capturing systems, wearable sensors and the robotic drummer. Section 10.2 describes the results of the experiments influencing design decisions for the real-time system. Section 10.3 contains a summary of concluding remarks.

10.1 Method

There are four major processing stages in our system. A block diagram of the system is shown in Figure 69. In the following subsection we describe each processing stage from left to right. In the acquisition stage performance information is collected using audio capture, two sensors on the instrument and a wearable sensor on the performer's body. Onsets for each seperate signal are detected after some initial signal conditioning. The onsets are used as input to four Kalman filters used for tempo tracking. The estimated beat periods for each signal are finally fused to provide a single estimate of the tempo.

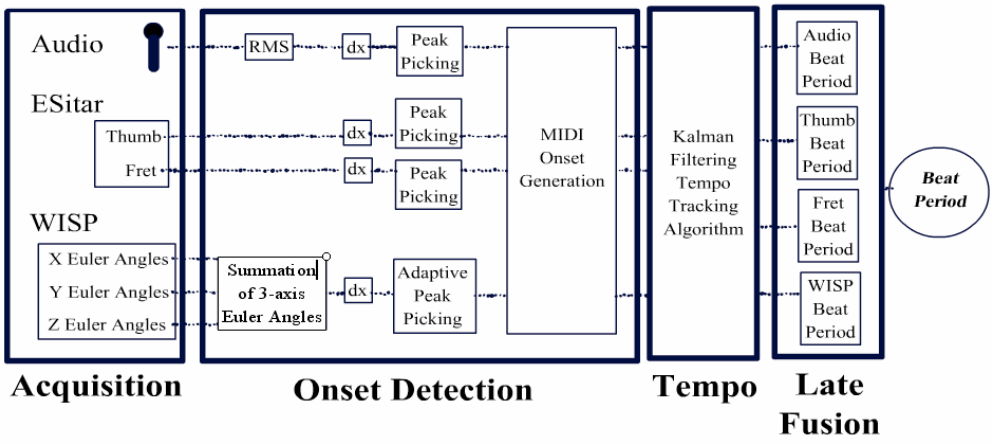


Figure 69 - Multimodal Sensors for Sitar Performance Perception.

10.1.1 Data Collection

For our experiments we recorded a data set of a performer playing the *ESitar* with a *WISP* on the right hand. Audio files were captured at a sampling rate of 44100 Hz. Thumb pressure and fret sensor data synchronized with audio analysis windows were recorded with *Marsyas* at a sampling rate of 44100/512 Hz using MIDI streams from the *ESitar*. Orientation data for the Open Sound Control (OSC) [199] streams of the *WISP* were also recorded.

While playing, the performer listened to a constant tempo metronome through headphones. 104 trials were recorded, with each trial lasting 30 seconds. Trials were evenly split into 80, 100, 120, and 140 BPM, using the metronome connected to the headphones. The performer would begin each trial by playing a scale at a quarter note tempo, and then a second time at double the tempo. The rest of the trial was an improvised session in tempo with the metronome.

10.1.2 Onset Detection

Onset detection algorithms were applied to the audio and sensor signals to gather periodicity information of the music performance. As seen in Figure 69, the RMS energy of the audio signal and the sum of the *WISP* 3-axes Euler angles are calculated while the values of the thumb and fret sensor are used directly. A peak-picking algorithm is applied to the derivatives of each signal to find onset locations. An adaptive peak-picking algorithm is applied on the *WISP* data to compensate for the large variability in wrist movement during performance.

Since each sensor captures different aspects of the underlying rhythmic structure, the onset streams are not identical in onset locations and phase. However we can expect that the distance between successive onsets will frequently coincide with the underlying tempo period and its multiples. In order to detect this underlying tempo period we utilize a real-time Kalman filtering algorithm for each sequence of onsets transmitted as MIDI signals.

10.1.3 Switching Kalman Filtering

Real-time tempo tracking is performed using a probabilistic Particle Filter. The algorithm tests various hypotheses of the output of a switching Kalman Filter against noisy onset measurements providing an optimal estimate of the beat period and beat [20]. Noisy onset measurements, extracted from the various sensor streams, are used as input to a real-time implementation of the tempo tracking algorithm [89]. In order to model the onset sequence we use a linear

dynamical system as proposed in [20]. The state vector x_k describing the system at a certain moment in time consists of the onset time τ_k and the period Δ_k defined as follows:

$$x_k = \begin{pmatrix} \tau_k \\ \Delta_k \end{pmatrix} = \begin{pmatrix} 1 & \gamma_k \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \tau_{k-1} \\ \Delta_{k-1} \end{pmatrix} + w_k$$

As can be seen the current state vector depends on the previous state vector x_{k-1} modified by a switching variable representing different rhythmic units (γ_k) and noise model (w_k) that takes into account deviations from the ideal sequence of onsets. Based on the above linear dynamical system the optimal sequence of tempo periods can be estimated using a Kalman filtering based approach. For more details please refer to [20].

In the following section the accuracy of the four estimated beat period streams is evaluated. In addition we show that late fusion of the streams can significantly improve tempo detection accuracy.

10.2 Experimental Results

Signal	**Tempo (BPM)**			
	80	***100***	***120***	***140***
Audio	46%	85%	86%	80%
Fret	27%	27%	57%	56%
Thumb	35%	62%	75%	65%
WISP	50%	91%	69%	53%
LATE FUSION:				
Audio/*WISP*/Thumb/Fret	45%	83%	89%	84%
Audio/*WISP*/Thumb	55%	88%	90%	82%
Audio/ *WISP*	58%	88%	89%	72%
Audio/Thumb	57%	88%	90%	80%
WISP/Thumb	47%	95%	78%	69%

Figure 70 - Comparison of Acquisition Methods.

Figure 70 shows the percentages of frames for which the tempo was correctly estimated. Tempo estimates are generated at 86Hz resulting in approximately 2600 estimates/30 second clip in the dataset. From the percentages of Figure 70, we can conclude that when using a single acquisition method, the *WISP* obtained the best results at slower tempos, and the audio signal was best for faster tempos. Overall, the audio signal performed the best as a single input, whereas the fret data provided the least accurate information.

When looking carefully through the detected onsets from the different types of acquisition methods, we observed that they exhibit outliers and discontinuities at times. To address this problem we utilize a late fusion approach where we consider each acquisition method in turn for discontinuities. If a discontinuity is found, we consider the next acquisition method, and repeat the process until either a smooth estimate is obtained or all acquisition methods have been exhausted. When performing late fusion the acquisition methods are considered in the order listed on bottom half of Figure 70.

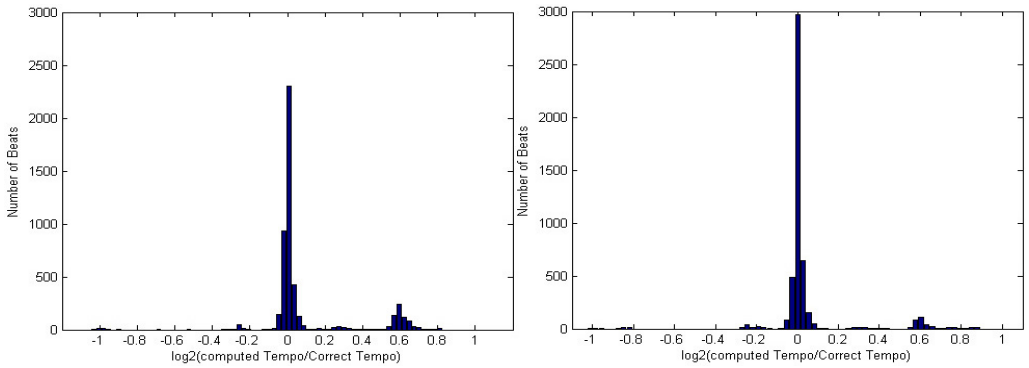


Figure 71 - Normalized Histogram of Tempo Estimation of Audio (left) and Fused Audio and Thumb (right)

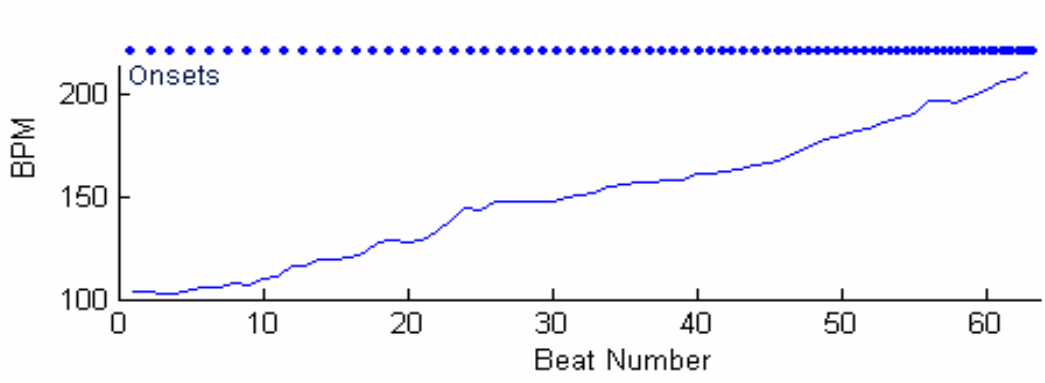


Figure 72 - Kalman Tempo Tracking with decreasing onset periods.

10.3 Summary

By fusing the acquisition methods together, we are able to get more accurate results. At 80 BPM, by fusing the information from *WISP* and the audio streams, the algorithm generates more accurate results then either signal on its own. When all the sensors are used together, the most accurate results are achieved at 140 BPM, proving that even the fret data can improve accuracy of tempo estimation. Overall, the information from the audio fused with the thumb sensor was the strongest.

Figure 71 shows histograms of the ratio between the estimated tempo and the correct tempo. The ratios are plotted on a $\log_2$ scale where the zero point indicates correct tempo while -1 and +1 indicate half and double errors respectively. Errors of 3/2 noticed at 0.6 on the $\log_2$ scale can be attributed to the tempo tracker falsely following triple meter onsets [63]. Figure 71 shows that a greater accuracy can be achieved by fusing the audio stream with the thumb sensor stream. Figure 72 shows how as the onset period calculated decreases, the output of the Kalman Filter increases in tempo. This is key to understand how this methodology can be used for a sitar performer to speed up and have the robotic drummer follow in real-time. This is a key attribute for “intelligent” machine-

based musical performance systems extending the performers and composers capabilities for creating new music.

Chapter 11

11 Rhythm Accompaniment Experiments

Beat Analysis for Automatic "Memory" Generation

Traditional Indian Classical music has been taught with the aid of a electronic Tabla box, where students can practice along with pre-recorded theka loops. This allows the performer to select any time cycle and rehearse at a variable tempo. The main problem with this system is that one beat repeats over and over again, which is boring and not realistic for a true performance situation. This motivated the work explored in this chapter of generating an interactive rhythm accompaniment system which would evolve based on human input. We will present a software framework to generate machine driven performance using a database structure for storing "memory" of "what to perform". This application introduces a new level of control and functionality to the modern North Indian musician with a variety of flexible capabilities.

This chapter begins by describing two applications for automatic rhythmic accompaniment. It then describes methods and experimentation on how a machine can automatically fill its own "memory" with rhythms by "listening" to audio files.

11.1 Applications

A series of performance scenarios using custom written software were designed to interface the *ESitar* with *MahaDeviBot*. This section will describe two frameworks towards interactive machine musicianship: (1) a symbolic MIR-based approach, and (2) an audio driven MIR-based approach.

11.1.1 A Symbolic MIR-based Approach

The Music Information Retrieval (MIR) community [35] inspired our initial framework and experimentation for this approach. The goal of this system is to generate a variety of rhythmic accompaniment that evolves over time based on human performance by using sensors to query databases of pre-composed beats. To achieve this, symbolic event databases (shown in Figure 73) for each robotic instrument were filled with rhythmic phrases and variations. During performance, at any given time, queries are generated by sensor data captured from the human performer. As this software is written in *ChucK* [189], it was easy for the databases to be time and tempo locked to each other to allow for multiple permutations and combinations of rhythm. Figure 73 shows an example of how the system can be mapped. In this case, thumb pressure from the *ESitar* queries what rhythm robotic instrument 1 (*Dha* strokes) will mechanically play on the low frame drum. It is possible to generate a large number of combinations and permutations of rhythms by accessing patterns in each database. This proved to be a successful technique for performances on stage[36].

[35] http://www.ismir.net/ (January 2007)
[36] Videos Available at: http://www.karmetik.com
(Technology → Robotics Department)

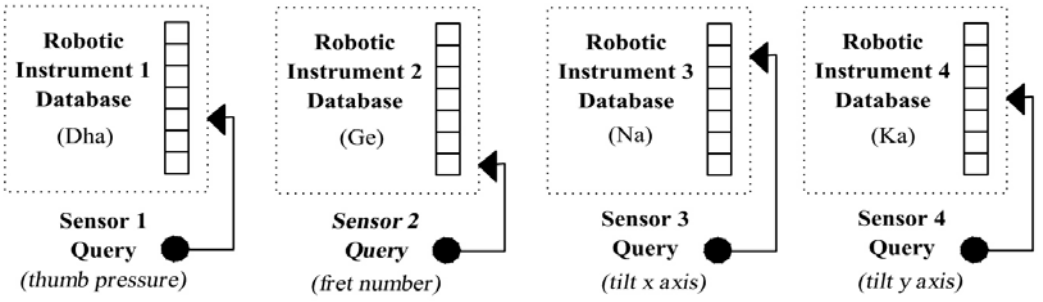


Figure 73 - Symbolic MIR-based approach showing how *ESitar* sensors are used as queries to multiple robotic drum rhythm databases.

One issue to address is how the queries are generated. In order to provide additional information for queries the derivatives and second derivatives of each sensor data stream are also utilized. Also there are more advanced feature extraction methods, for example obtaining interonset interval values between peaks of the thumb pressure data. There are many algorithms that can be explored; however, the main philosophical question is whether the human should have full control of the machine's performance.

11.1.2 An Audio Driven MIR-based Approach

A major drawback to the system described in the section above is that rhythms have to be pre-programmed symbolically "by hand". This motivated this second approach, to have the machine automatically fill its databases by listening to pre-recorded drum beats, music, or even take audio input from a live performer. We chose to write this software in *Marsyas* [177] because of its strong audio analysis capabilities.

Now that we have a system to convert audio signals into symbolic data (described in Section 11.2) that can be used for robotic performance, the next step is to create a meaningful retrieval method for real-time performance.

The first step is having a human play the *ESitar* along with each beat stored in the audio database. The sensor data is recorded and aligned with the corresponding symbolic rhythm. For our initial experiments, we chose to use the

thumb sensor as our query key, as seen in Figure 74. Before a performance, a database of symbolic rhythms for “Boom” and “Chick” are matched with the corresponding thumb sensor data.

During live performance, a window of thumb sensor data is collected forming a query. This query is then compared with all the pre-recorded thumb sensor data in the database, by doing a simple correlation calculation. The closest match is used to select the corresponding symbolic “boom” and “chick” rhythm for performance by *MahaDeviBot*.

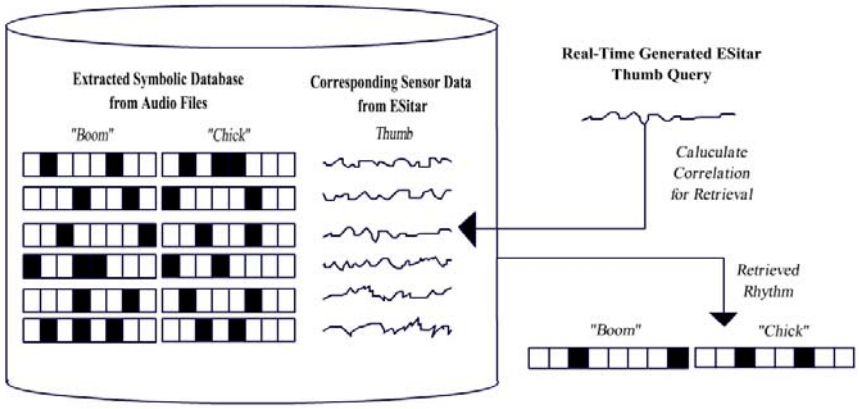


Figure 74 - Audio Driven Retrieval Approach.

11.2 Method

This section explores the use of signal processing techniques to provide rhythmic transcriptions of polyphonic music and drum patterns [180] to fill the databases and build computer “memory”. Transcription refers to the process of converting the audio recording to a symbolic representation similar to a musical score. The transcribed symbolic representation can then be used for retrieval tasks as described in the above applications.

In order to extract information for transcription and retrieval applications we perform what we term “boom-chick” analysis. The idea is to detect the onset of low frequency events typically corresponding to bass drum hits and high

frequency events typically corresponding to snare drum hits from the polyphonic audio sources. This is accomplished by onset detection using adaptive thresholding and peak picking on the amplitude envelope of two of the frequency bands. Unlike many of the proposed approaches in drum transcription we do not utilize a classification model because that would constrain the application of our method to specific types of sounds. Even when a classification approach is utilized a data-driven front end as the one described in this work can be used as a preprocessing step for training. Two front-ends are compared. The first one is based on using regular band pass filters designed for detecting drum sounds. The second one is based on the wavelet transform and is similar to the Beat Histogram calculation front-end described in [178]. Experiments in section IV compare results using the two front-ends.

11.2.1 Filter analysis

In order to detect drum sounds two sub bands (“boom” and “chick”) are utilized. The signal is analyzed in windows of approximately 3 seconds. The frequency range for the low sub band is 30Hz to 280Hz. This was determined empirically to give good results over a variety of drum loops and music recordings. A simple band pass filter implemented in the frequency domain was used to select the “boom” band. The frequency range for the high sub band is 2700 Hz- 5500 KHz. The “chick” sub band is implemented using a Butterworth filter.

11.2.2 Wavelet-based analysis

The second front-end is based on decomposing the signal into different frequency bands using a Discrete Wavelet Transform (DWT) similar to the method described in [178] for the calculation of *Beat Histograms*. Figure 75 shows how a window of an audio signal (approximately 3 seconds) is converted to the wavelet domain with a fourth order Daubechies wavelet. The “boom” band and the

"chick" band were determined through experimentation although unlike the filter-based approach the boundaries were constrained by the octave characteristics of the dyadic wavelet transform. The bands with the most information were approximately 300Hz-600Hz for the "boom" and 2.7KHz-5.5KHz for the "chick". To convert the signals back to the time domain, the wavelet coefficients for all the bands except the chosen one are set to zero and then the Inverse Wavelet Transform is calculated, converting the subband signal to the time domain.

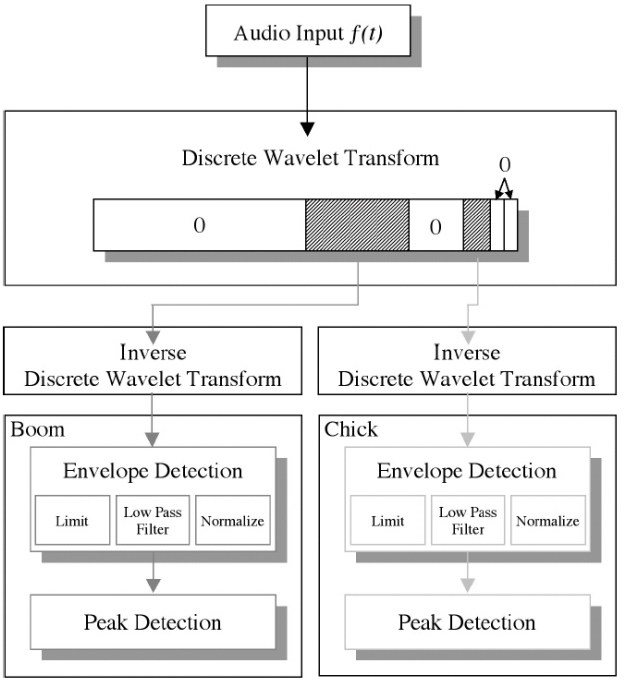


Figure 75 - Wavelet Front End for Drum Sound Detection.

11.2.3 Envelope extraction and Onset Detection

Once the sub bands are calculated using either front-end, they are processed to find the onset of the "boom" and "chick" sounds. First the envelope of each subband is calculated by using full wave rectification followed by low pass filtering and normalization. Once the envelope is extracted an adaptive peak detection algorithm based on thresholding is utized to find the onset times of the percussive sounds. If the spacing between adjacent peaks is small, then only the

highest peak will be selected. Figure 76 shows a drum loop decomposed into “boom” and “chick” bands and the corresponding detected onsets.

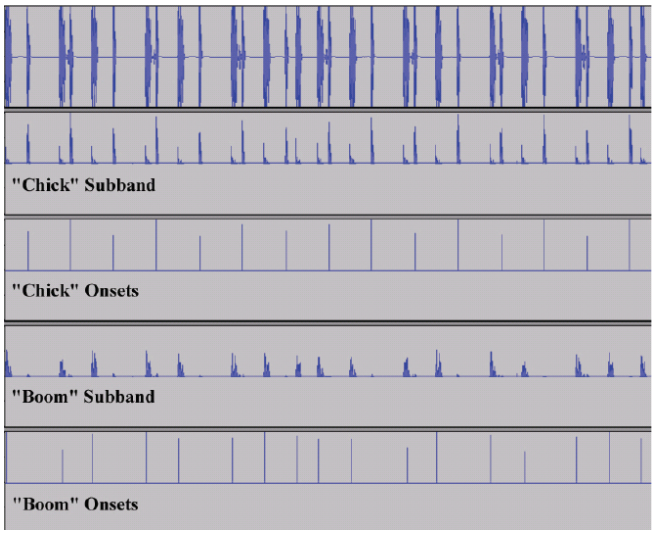


Figure 76 - Audio Signal “Boom-Chick” Decomposition.

11.2.4 Transcription

The goal of our transcription is to convert the two sequences of onset times for the “boom” and “chick” bands into symbolic form. Music notation is relative to the tempo of the piece which means that two pieces that have the same drum pattern played at different tempi will still have the same notation. The first step for transcription is to calculate the IOI (Interonset Intervals) which are the time differences in samples between onset positions. The IOIs are subsequently quantized in order to ignore small variations in tempo. A combination of heuristics based on music knowledge and clustering of the quantized IOIs is used to select the IOI that corresponds to a quarter note. Once this basic rhythmic unit is established all IOIs are expressed relative to it as integer ratios. The resulting ratios can then be directly rendered in music notation. Currently the output of the system is a textual representation of the durations. Figure 77 show a common music notation and an Indian tabla notation rendering of the output of the system.

Even though the graphic notation was rendered manually it corresponds directly to the output of the drum transcription system.

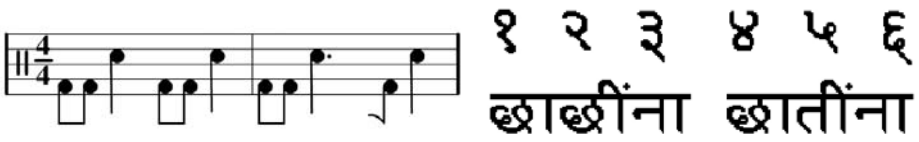


Figure 77 - (left) Transcribed Drum Loop (Bass and Snare). (right) Transcribed Tabla Loop in Hindi (Dadra – 6 Beat).

11.2.5 Data Collection

Three sample data sets were collected and utilized. They consist of techno beats, tabla *thekas* and music clips. The techno beats and tabla *thekas* were recorded using DigiDesign Digi 002 ProTools at a sampling rate of 44100 Hz. The techno beats were gathered from Dr. Rex in Propellerheads Reason. Four styles (Dub, House, Rhythm & Blues, Drum & Bass) were recorded (10 each) at a tempo of 120 BPM. The tabla beats were recorded with a pair of AKG C1000s to obtain stereo separation of the different drums. Ten of each of four “thekas” (meaning beats per cycle) were recorded (Tin Taal Theka (16), Jhaap Taal Theka (10), Rupak Theka (7), Dadra Theka (6)). The music clips consist of jazz, funk, pop/rock and dance music with strong rhythm.

11.3 Experimental Results

The evaluation of the system was performed by comparative testing between the actual and detected beats by two drummers. After listening to each track, false positive and false negative drum hits were detected separately for each type (“boom” and “chick”). False positives are the set of instances in which a drum hit was detected but did not actually occur in the original recording. False negatives are the set of instances where a drum hit occurs in the original recording but is not detected automatically by the system. In order to determine consistency in annotation, five random samples from each dataset were analyzed by both

drummers. The results were found to be consistent and therefore the ground-truth annotation task was split evenly among the two drummers. These two expert users also provided feedback for the fine tuning of the system.

The results are summarized using the standard precision and recall measures. Precision measures the effectiveness of the algorithm by dividing the number of correctly detected hits (true positives) by the total number of detected hits (true positives + false positives). Recall represents the accuracy of the algorithm by dividing the number of correctly detected hits (true positives) by the total number of actual hits in the original recording (false negatives + true positive). Recall can be improved by lowering precision and vice versa. A common way to combine these two measures is the so called F-measure defined as (P is precision, R is recall and higher values of the F-measure indicate better retrieval performance):

$$F = \frac{2 * P * R}{P + R}$$

Figure 78 summarizes the detection results using the two front-ends for the 3 different audio data collections. As can be seen from the figure overall the detection of the low frequency “boom” events is better than the “chick” events. This is expected as there is less variability in bass drum sounds and less interference from other instruments and percussive sounds. The results are better for the drum loops and tabla *thekas* where there are only percussive sounds. As expected the results are not as good for the polyphonic audio clips where the presence of other interfering sounds, such as singers, guitars, and other instruments makes detection harder. The difference between the feature front-ends is statistically insignificant given the size of the collections used. As the filter front-end is faster to compute, we provide more detailed results for that front-end in Figure 79. Drum transcription of a 30-second clip on a Pentium III 1.2 GHz takes approximately 8 seconds for the wavelet front-end and 2 seconds for the filter front-end. These performance measurements are relative as there was no

effort to optimize the run-time performance of the system. Also the analysis is causal (requires a single pass) and therefore can be performed on streaming audio.

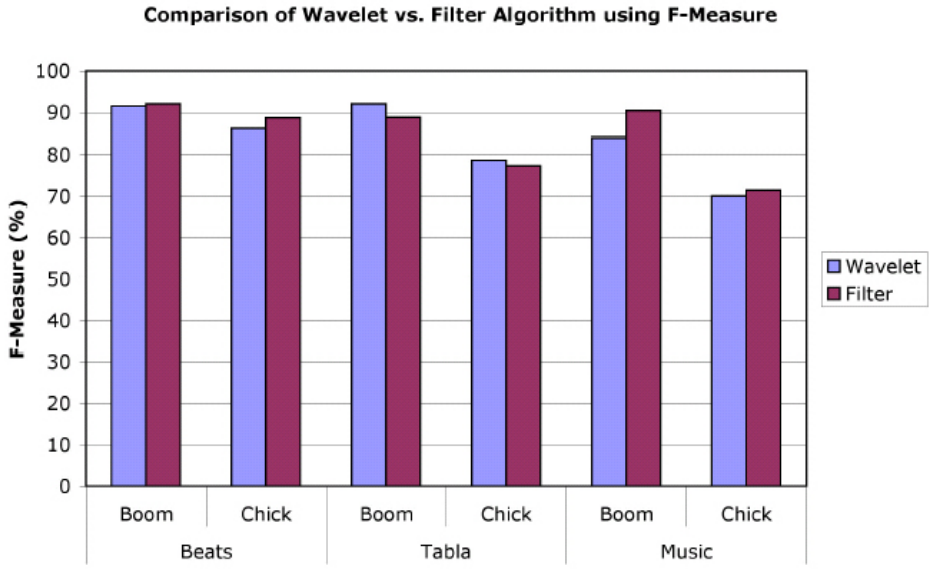


Figure 78 - Comparison of Filter and Wavelet Front-end.

Category	Recall	Precision	F-measure
Rnb	0.94	0.97	0.95
Dnb	0.83	0.86	0.84
Dub	0.92	0.84	0.88
Hse	0.93	0.83	0.88
Average	0.90	0.87	0.89
Dadra	0.53	1.00	0.69
Rupak	0.51	1.00	0.68
Jhaptaal	0.79	1.00	0.88
Tintaal	0.73	1.00	0.84
Average	0.64	1.00	0.77
Various	0.60	0.51	0.55
Dance	0.94	0.63	0.75
Funk	0.93	0.58	0.72
Average	0.82	0.57	0.67

Category	Recall	Precision	F-measure
Rnb	0.85	0.83	0.84
Dnb	0.93	0.88	0.90
Dub	0.92	0.97	0.94
Hse	0.99	1.00	0.99
Average	0.92	0.92	0.92
Dadra	0.98	0.90	0.94
Rupak	1.000	0.56	0.72
Jhaptaal	0.98	0.93	0.95
Tintaal	0.91	0.97	0.94
Average	0.97	0.84	0.89
Various	0.88	0.79	0.83
Dance	0.92	0.89	0.90
Funk	0.86	0.95	0.90
Average	0.89	0.88	0.88

Figure 79 - “Chick” hit detection results for Filter Front End (left). "Boom" hit detection results for Filter Front End (right).

11.4 Summary

This chapter describes a system for a robotic drummer to accompany a live sitar performer. The system uses a retrieval approach to access pre-recorded/processed data in real-time based on interaction with the human performer. In essence this is a “memory” system for musical information. We also show experimentation on methods of the machine filling its own “memory” using audio signal processing methods for onset detection. A comparison of a filter-based method and a wavelet-based method is described. The final goal of this line of research is to have the machine listen and create its own database during a performance generating new rhythmic passages at every concert, following the tradition of North Indian classical music.

Chapter 12

12 Pitch & Transcription Experiments

Pedagogical Tools for Preservation

Historically, most musical traditions were preserved via oral transmission. With the invention of music notation, audio recordings, and video, more information can be retained. However, most of the valuable performance data must still be passed by oral means. There will never be a technological replacement for face-to-face teaching, but new methods for archiving performance data will let us retain and disseminate more information. Automatic music transcription is a well-researched area [93, 101, 203]. The novelty of our work presented in the chapter is that we look beyond the audio data by using sensors to avoid octave errors and problems caused from polyphonic transcription. In addition, our work does not share the bias of most research that focuses only on Western music.

This chapter describes a music transcription system for sitar performance. Unlike many Western fretted stringed instruments (classical guitar, viola de gamba, etc) sitar performers pull (or "bend") their strings to produce higher pitches. In normal performance, the bending of a string will produce notes as much as a fifth higher than the same fret-position played without bending. In

addition to simply showing which notes were audible, our framework also provides information about how to produce such notes. A musician working from an audio recording (or transcription of an audio recording) alone will need to determine from which fret they should begin pulling. This can be challenging for a skilled performer, let alone a beginner. By representing the fret information on the sheet music, sitar musicians may overcome these problems.

Most automatic music transcription research is concerned with producing sheet music from the audio signal alone. However, the audio data does not include certain performance data that is vital for the preservation of instrument performance techniques and the creation of annotated guidelines for students. We propose the use of modified traditional instruments enhanced with sensors which can obtain such data; as a case study we examine the sitar. This chapter describes how the technology we built can be used to preserve intricacies of traditional North Indian performance in generating a paradigm for automatically generated sheet music.

12.1 Method

For the research described in this chapter, fret data from the *ESitar* was captured by a network of resistors connecting each fret as described in Chapter 7. The fret sensor is translated into MIDI pitch values based on equivalent resistance induced by left hand placement on the neck of the instrument. Each fret has a “bucket” of values, converting raw sensor data into discrete pitch values seen in Figure 81. Data was recorded at a sampling rate of (44100 ÷ 512) Hz. The synchronized audio signal was recorded with a Shure Beta-57 microphone at 44100 Hz. The entire system is displayed in Figure 80.

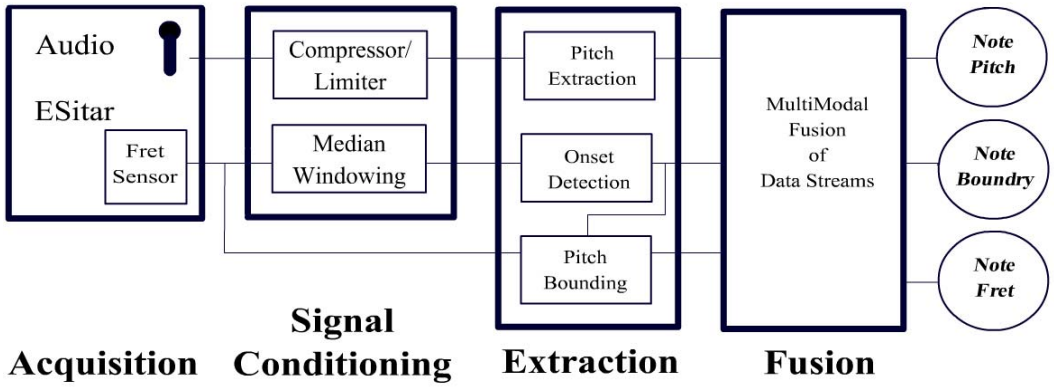


Figure 80 - Block Diagram of Transcription Method

12.1.1 Audio Signal Chain and Pitch Extraction

A compressor/limiter was used on the audio signal to generate balanced amplitude for all frequencies. Without this step, our experiments yielded poor results for notes played at lower frequencies.

To automatically determine the pitch an implementation of the method described in [15] was used. We utilize the autocorrelation function to efficiently estimate the fundamental frequency (f_0). For a time signal *s(n)* that is stationary, the autocorrelation $r_s(\tau)$ as a function of the lag is defined as:

$$r_s(\tau) = \frac{1}{n}\sum_{j=t}^{t+N} s(t)s(t+\tau)$$

This function has a global maximum for $r = 0$. If there are also significant additional maxima, the signal is called periodic and there exists a lag_0, the period, so that all these maxima are placed at the lags n_0, for every integer n, with $r_s(n_0) = r_s(0)$. The inverse of the lag_0 provides an estimation of the fundamental frequency f_0. The period is determined by scanning $r_s( )$, starting at zero, and stopping at the first global maximum with a non-zero abscissa. Quadratic interpolation is used to further improve the frequency estimation. In practical cases, the relative amplitude of those maxima may change and other maxima may appear due to small aperiodicities of the signal. The issue then is to relevantly select which

maximum corresponds to the f_0 by considering several candidates under a plausible range and pick the one with the highest confidence. See [15] for further references on the algorithm.

12.1.2 Fusion with Fret Signal Chain

Although military grade (1% tolerance) resistors were used, the fret data was still noisy due to environmental factors including the resistance of the performer's body. For each sample, we smoothed the data by comparing the median value of the previous ten samples with the median of the next ten samples (including the current sample). If the median values differed by more than a certain amount, we marked that sample as being a note boundary.

To get an accurate final result, pitch information from the audio signal chain was fused with onset and pitch boundaries calculated from the fret signal chain. The fret provided convenient lower and upper bounds on the pitch: a note cannot be lower than the fret, nor higher than a fifth (ie seven MIDI notes) above the fret. Using the note boundaries derived from the fret data, we found the median value of the pitches inside the boundaries supplied by the fret data. These are represented by the vertical lines in Figure 81, and are the note pitches in the final output.

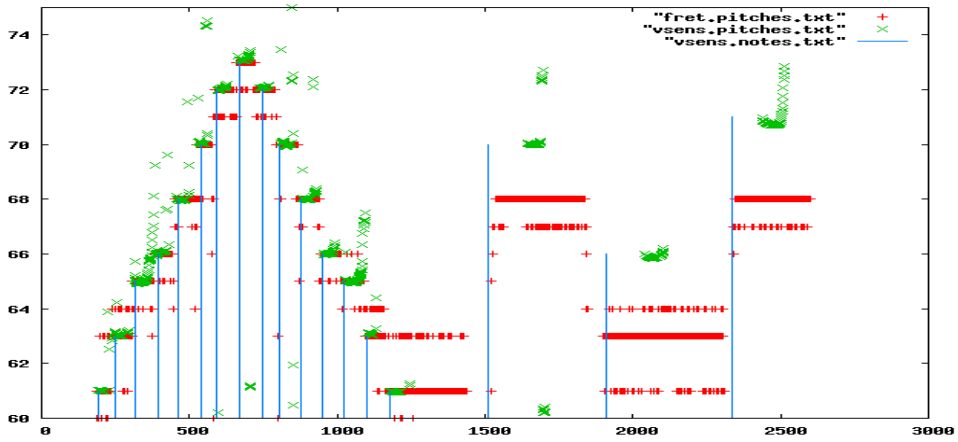


Figure 81 - Fret data (red), Audio pitches (green) , and the resulting detected notes (blue lines).

12.2 Sheet Music

Although there are many computer notation programs for Western music, there are no such counterparts for Indian music. Indian notation is not standardized and there is no way to notate both frets and audible notes, so we invented our own notation. In North Indian classical music, notes are described by seven *swaras*. They are known as *Shadja (Sa), Rishab (Re), Gandhar (Ga), Madhyam (Ma), Pancham (Pa), Dhaivat (Dha), and Nishad (Ni).* These are equivalent to the Western solfege scale (More details in Appendix A). We produce sheet music (Figure 82) showing sitar musicians the audible note played and which fret was used.

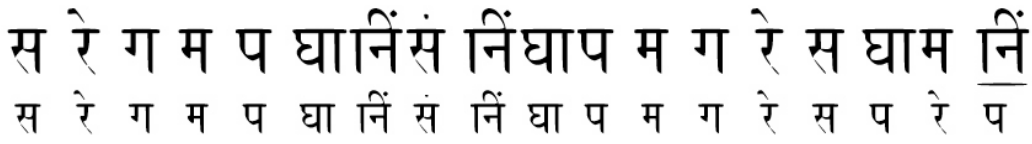


Figure 82 - Sheet music of Sitar Performance. The top notes are the audible notes, while the lower notes are the fret position. Notice the final three notes were pulled.

12.3 Summary

We have developed a system which can be used to produce sheet music for sitar musicians. In addition to simply showing which notes were audible, it also provides information about how to produce such notes. As well as aiding beginners, the sheet music may be used for archival purposes.

Although the system works well for simple sitar melodies, currently it does not detect glissandi. In addition, since we detect note boundaries by examining the fret data, we cannot detect repeated notes plucked on the same fret. Future work in this area is planned, as is inventing notation to display expressive marks.

Chapter 13

13 "Virtual-Sensor" Gesture Extraction

Capturing data from the Timbre Space

Throughout history musical instruments have been some of the best examples of artifacts designed for interaction. In recent years a combination of cheaper sensors, more powerful computers and rapid prototyping software have resulted in a plethora of interactive electroacoustic music performances and installations. In many of these performances, traditional acoustic instruments are blended with computer generated sounds and visuals. Automatically sensing gestures is frequently desired in such interactive multimedia performances.

There are two main approaches to sensing instrumental gestures. In direct acquisition, traditional acoustical instruments are extended/modified with a variety of sensors such as force sensing resistors, and accelerometers (as described in Chapters 5-8). The purpose of these sensors is to measure various aspects of the gestures of the performers interacting with their instruments. A variety of such "hyper" instruments have been proposed (as described in Chapter 2). However, there are many pitfalls in creating such sensor-based controller systems: Purchasing microcontrollers and certain sensors can be expensive; The massive tangle of wires interconnecting one unit to the next can become failure-

prone. Things that can go wrong include: analog circuitry break down, or sensors wearing out right before or during a performance forcing musicians to carry a soldering iron along with their tuning fork. The biggest problem with “hyper” instruments is that there is usually only one version, and the builder is the only one that can benefit from the data acquired and use the instrument in performance. These problems have motivated researchers to work on indirect acquisition in which the musical instrument is not modified in any way. The only input is provided by noninvasive sensors, typically with one or more microphones. The recorded audio is then analyzed to measure the various gestures. Probably the most common and familiar example of indirect acquisition is the use of automatic pitch detectors to turn monophonic acoustic instruments into MIDI instruments. In most cases indirect acquisition doesn’t directly capture the intended measurement and the signal needs to be analyzed to extract the information. Usually this analysis is achieved by using real-time signal processing techniques. More recently an additional stage of supervised machine learning has been utilized in order to “train” the information extraction algorithm. The disadvantage of indirect acquisition is the significant effort required to develop the signal processing algorithms. In addition, if machine learning is utilized the training of the system can be time consuming and labor intensive.

The main problem addressed in this chapter is the efficient and effective construction of indirect acquisition systems for musical instruments in the context of interactive media. Our proposed solution is based on the idea of using direct sensors to train machine learning models that predict the direct sensor outputs from acoustical data. Once these indirect models have been trained and evaluated, they can be used as “virtual” sensors in place of the direct sensors. This approach is motivated by ideas in multimodal data fusion with the slight twist that in our case the data fusion is only used during the learning phase. We believe that the idea of using direct sensors to learn indirect acquisition can be applied to other areas of multimodal interaction in addition to musical instruments.

This approach of using direct sensors to “learn” indirect acquisition models has some nice characteristics. Large amounts of training data can be collected with minimum effort simply by playing the enhanced instrument with the sensors. Once the system is trained and provided the accuracy and performance of the learned “virtual” sensor is satisfactory there is no need for direct sensors or modifications to the instrument.

The traditional use of machine learning in audio analysis has been in classification where the output of the system an ordinal value (for example the instrument name). We explore regression that refers to machine learning systems where the output is a continuous variable. One of the challenges in regression is obtaining large amounts of data for training; this is much easier using our proposed approach. In our experiments, we use audio-based feature extraction with synchronized continuous sensor data to train a “virtual” sensor using machine learning. More specifically we describe experiments using the *ESitar*, further extending the use of the traditional North Indian sitar. This chapter describes our method of experimentation, results and conclusions.

13.1 Method

13.1.1 Audio-Based Analysis

In this section we describe how the audio signal is analyzed. For each short time segment of audio data numerical features are calculated. At the same time, sensor data is also captured. These two steams of data potentially have different sampling rates. In addition, in some cases, the gestural data is not regularly sampled. We have developed tools to align the two streams of data for these cases. Once the features are aligned with the sensor data, we train a “pseudo” sensor using regression and explore its performance.

13.1.1.1 Audio-Based feature extraction

The feature set used for the experiments described in this paper is based on standard features used in isolated tone musical instrument classification, music and audio recognition. It consists of four features computed based on the Short Time Fourier Transform (STFT) magnitude of the incoming audio signal. It consists of the Spectral Centroid (defined as the first moment of the magnitude spectrum), Rolloff, Flux as well as RMS energy. The features are calculated using a short time analysis window with duration 10-40 milliseconds. In addition, the means and variances of the features over a larger texture window (0.2-1.0 seconds) are computed resulting in a feature set with eight dimensions. The large texture window captures the dynamic nature of spectral information over time and was a necessary addition to achieve any results in mapping features to gestures. Ideally the size of the analysis and texture windows should correspond as closely as possible to the natural time resolution of the gesture we want to map. In our experiments we have looked at how these parameters affect the desired output. In addition, the range of values we explored was determined empirically by inspecting the data acquired by the sensors.

13.1.1.2 Audio-Based Pitch Extraction

The pitch of the melody string (without the presence of drones) is extracted directly from the audio signals using the method described in [173]. This method is an engineering simplification of a perceptually-based pitch detector and works by splitting the signal into two frequency bands (above and below 1000Hz), applying envelope extraction on the high-frequency band followed by enhanced autocorrelation (a method for reducing the effect of harmonic peaks in pitch estimation). Figure 83 shows a graph of a simple ascending diatonic scale calculated directly from audio analysis.

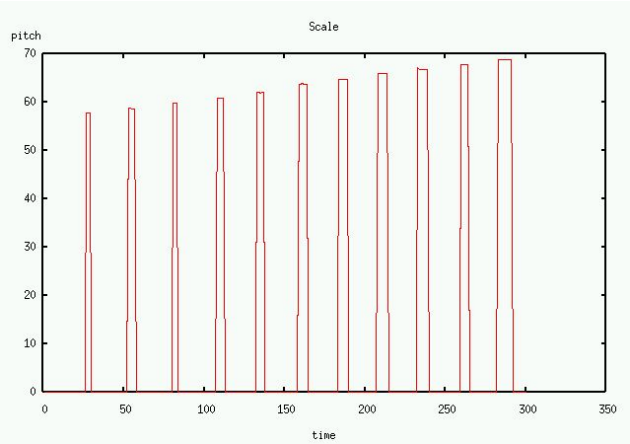


Figure 83 - Graph of Audio-Based Pitch extraction on an ascending diatonic scale without drone strings being played.

The audio-based pitch extraction is similar to many existing systems that do not use machine learning, therefore it will not be further discussed. Currently the audio-based pitch extraction works only if the drone strings are not audible. As a future project, we are planning to explore a machine learning approach to pitch extraction when the drone strings are sounding.

The interaction between sensors and audio-based analysis can go both ways. For example, we used the audio-based pitch extractor to debug and calibrate the fret-sensor. Then the fret sensor we used as ground truth for machine learning of the pitch in the presence of drone strings. We believe that this bootstrapping process can be very handy in the design and development of gestural music interfaces in general.

13.1.1.3 Regression Analysis

Regression refers to the prediction of real-valued outputs from real-valued inputs. Multivariate regression refers to predicting a single real-valued output from multiple real-valued inputs. A classic example is predicting the height of a person using their measured weight and age. There are a variety of methods proposed in the machine learning [114] literature for regression. For the experiments described in this chapter, we use linear regression where the output is formed as a

linear combination of the inputs with an additional constant factor. Linear regression is quick to compute and therefore useful for doing repetitive experiments for exploring the parameters. We also employ a more powerful back propagation neural network [96] that can deal with non-linear combinations of the input data. The neural network is slower to train but provides better regression performance. Finally, the M5 prime decision tree based regression algorithm was also used [138]. The performance of regression is measured by a correlation coefficient which ranges from 0.0 to 1.0 where 1.0 indicates a perfect fit. In the case of gestural control, there is significant amount of noise and the sensor data doesn't necessarily reflect directly the gesture to be captured. Therefore, the correlation coefficient can mainly be used as a relative performance measure between different algorithms rather than an absolute indication of audio-based gestural capturing.

13.1.1.4 Data Collection

In order to conduct the experiments the following tools were used to record audio and sensor data. Audio files were recorded with DigiDesign's ProTools Digi 002 Console using a piezo pickup (shown in Figure 45) placed directly on the sitar's *tabli*. MIDI data was piped through *pure data [136]* where it was filtered and sent to a custom built MIDI Logger program which recorded time stamps and all MIDI signals. Feature extraction of the audio signals was performed using *Marsyas*. The sampling rate of the audio files and the sensor data were not the same. The audio data was sampled at 44100 Hz and then down sampled for processing to 22050 Hz. Also the sensor data were not regularly sampled. Tools were developed to align the data for use with *Weka [72],* a tool for data mining with a collection of various machine learning algorithms.

For the experiments, excerpts of a *jhala* portion of a *raga* were performed on the *ESitar*. *Jhala* is a portion of sitar performance characterized by the constant repetition of pitches, including the drone, creating a driving rhythm [7]. Because of the rhythmic nature of this type of playing we chose to explore the signals of

the thumb sensor to get an indication of *mizrab* pluck direction using audio-based feature analysis and regressive machine learning algorithms.

13.2 Experimental Results

Our first experiment was to analyze the effect of the analysis window size used for audio based feature extraction. Table 1 shows the results from this experiment. Take note that the texture size remained constant at 0.5 seconds and linear regression was used. The correlation coefficient for random inputs is 0.14. It is apparent based on the table that an analysis window of length 256 (which corresponds to 10 milliseconds) achieves the best results. It can also be seen that the results are significantly better than chance. We used this window size for the next experiment.

Analysis Window Size (samples at 22.5 KHz)	128	256	512
Correlation Coefficient	0.2795	0.3226	0.2414

Table 1 - Effect of analysis window size.

The next experiment explored the effect of texture window size and choice of regression method. Table 2 shows the results from this experiment. The rows correspond to regression methods and the columns correspond to texture window sizes expressed in number of analysis windows. For example, 40 corresponds to 40 windows of 256 samples at 22050 Hz sampling rate which is approximately 0.5 seconds. To avoid overfitting we use a percentage split where the first 50% of the audio and gesture data recording is used to train the regression algorithm which is subsequently used to predict the second half of recorded data.

	10	*20*	*30*	*40*
Random Input	0.14	0.14	0.14	0.14
Linear Regression	0.28	0.33	0.28	0.27
Neural Network	0.27	0.45	0.37	0.43
M5' Regression Method	0.28	0.39	0.37	0.37

Table 2 - Effect of texture window size (columns) and regression method (rows).

It is evident from the table and Figure 84 that the best choice of texture window size is 20 which corresponds to 0.25 seconds. In addition, the best regression performance was obtained using the back propagation neural network. Another interesting observation is that the relation of inputs to outputs is non-linear as can be seen from the performance of the neural network and M5' regression algorithm compared to the linear regression.

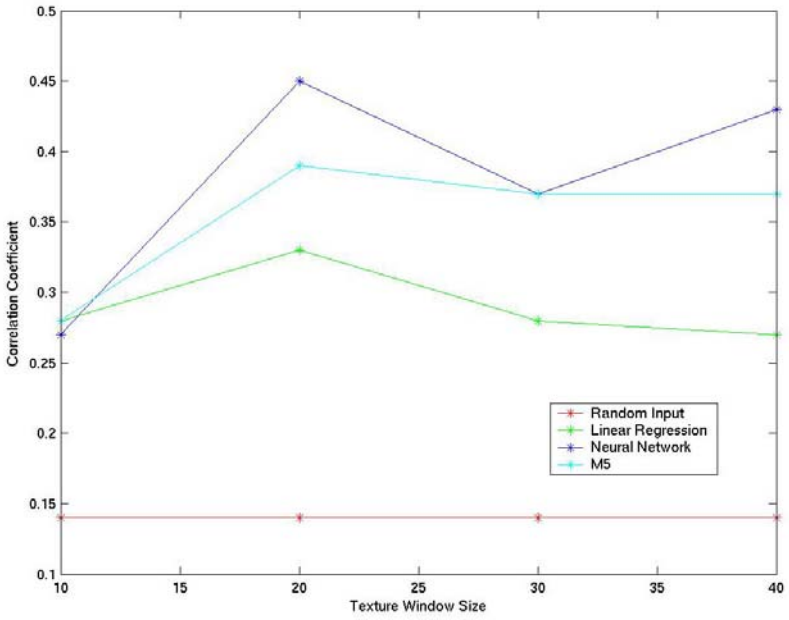


Figure 84- Graph showing the effect of texture window size and regression method.

13.3 Summary

In this chapter, we propose the use of direct sensors to “train” machine learning models based on audio feature extraction for indirect acquisition. Once the model is trained and its performance is satisfactory the direct sensors can be discarded. That way large amounts of training data for machine learning can be collected with minimum effort just by playing the instrument. In addition, the learned indirect acquisition method allows capturing of non-trivial gestures without modifications to the instrument. We believe that the idea of using direct sensors to train indirect acquisition methods can be applied to other area of interactive media and data fusion.

There are many directions for future work. We are exploring the use of additional audio-based features such as Linear Prediction Coefficients (LPC) and sinusoidal analysis. We are also planning more extensive experiments with more instruments, players and desired gestures. Creating tools for further processing the gesture data to reduce the noise and outliers is another direction for future research. Another eventual goal is to use these techniques for transcription of music performances.

The idea of a “virtual” sensor will aid in making the software and tools designed to work specifically with the *ESitar* accessible to those who do not have sensor enhanced instruments. This will help disseminate the information to a larger audience of performers in India and continue to extend and preserve traditional techniques.

Chapter 14

14 Affective Computing Experiments

Gesture-Based Human Emotion Detection.

Detecting and recognizing motion evolved to be an essential aspect of human survival. As part of this, the visual-perceptual system is extremely sensitive to implicitly coherent structures revealed through biological movement. Humans have the ability to extract emotional content from non-verbal human interaction, facial expressions and body gestures. Training a machine to recognize human emotion is far more challenging and is an active field of research generally referred to as “affective computing” [130]. Advances in this area will have a significant impact on human-computer interactive interfaces and applications.

Since a performer is conveying feeling, sentiment and mood, emotion is an essential part of any music. Particularly in Indian Classical music, *rasa* theory refers to the aesthetic experience and emotional response of the audience during a performance. There are nine *rasa* states in which to categorize all *raga-s: shringara* (romantic/erotic), *haysa* (comic), *karuna* (pathetic), *raudra* (wrathful), *vira* (heroic), *Bhayanaka* (terrifying), *bibhatsa* (odious), *adbhuta* (wonderous), *shanta* (peaceful, calm) [7]. Hence affective computing systems have a role in

advanced “intelligent” music systems of the future. The experiments portrayed in this chapter are our first attempts at exploring this problem. They are in no way complete or fully comprehensive. We are simply setting a paradigm for which experimentation in this area can be explored, and giving initial results using the framework and technology we have discussed through out this dissertation.

In this chapter we first describe background work in affective computing including motivation and applications to other fields other that music. Next, we describe our method for collecting data using the VICON motion capture system. Next, using collected data we show results of training automatic emotion classifiers using different machine learning algorithms. These results are compared with a user study of human perception of the same data.

14.1 Background

Imagine online learning systems which can sense if a student is confused and can re-explain a concept with further examples [83]. Imagine global positioning systems in cars re-routing drivers to less crowded, safer streets when they sense frustration or anger [56]. Imagine lawyers using laptops in the court room to analyze emotional behavior content from witnesses. Imagine audiovisual alarms activating when security guards, train conductors, surgeons or even nuclear power plant workers are bored or not paying attention [123]. These possible scenarios are indicative examples of what motivates some researchers in this emerging field of affective computing.

Currently there are two main approaches to affective computing: Audio-based techniques that determine emotion from spoken word (described in [40, 82, 182]) and video-based techniques that examine and classify facial expressions (described in [13, 52, 151]). More advanced systems are multimodal and use a variety of microphones, video cameras and other biological sensors to enlighten the machine with richer signals from the human [25, 158, 204]. The above list of

references is representative of existing work and not exhaustive. For more details on the evolution and future of affective computing as well as more complete lists of references readers are pointed to papers [123, 131].

In the review of the literature as briefly discussed above, almost all systems focus on emotion recognition based on audio or facial expression data. Most researchers do not analyze the full skeletal movements of the human body, with the exception of [132] who uses custom-built sensor systems such as a "Conductor's Jacket", glove, or a respiratory sports bra for data acquisition of selected human body movements. Others have used motion capture systems for affective computing experiments with methods different from our own [133, 183]. Following up on research by [45, 186] who present experiments which confirm that body movements and postures do contain emotional data, our team has designed a system that uses the VICON[37] motion capturing system to obtain gestural data from the entire body to identify different types of emotion.

14.2 Method

14.2.1 Data collection

Markers were placed at fourteen reference points on five different subjects (two of whom were professional dancers). The subjects were asked to enact four basic emotions using their body movements. No specific instructions for how these emotions should be enacted were given, resulting in a variety of different interpretations. The basic emotions used were sadness, joy, anger, and fear. The VICON system measured the trajectories of each subject's movement in 3D space at a sampling rate of 120 Hz. Each subject performed each emotion twenty five times; and each emotion was performed for a duration of ten seconds. We manually labeled the reference points of the body throughout the window of

[37] http://www.vicon.com (May 2005)

movement and filled missing data points by interpolation. A database of 500 raw data files with continuous x, y, and z-coordinates of each of the fourteen reference points was created. This database was used to extract features for the machine learning analysis described in Section 14.3. Figure 85 shows a screenshot of the data capturing process.

Data collection involving psychology and perception is challenging and its validity is frequently questioned. Although it can be argued that in acting out these emotions the subject's cognitive processes might be different than the emotion depicted, it turns out that the data are perceived correctly consistently even when abstracted as described in the next section. In addition, since the choice of movements was made freely by the subjects we can stipulate that their motions are analogous to the actual display of these emotions. Even though this way of depicting emotions might be exaggerated it is perceptually salient and its variability provides an interesting challenge to affective computing.

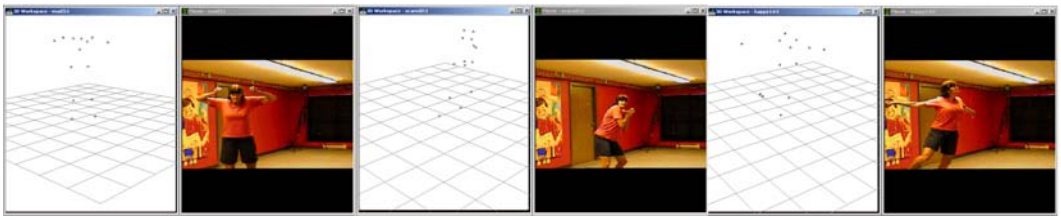


Figure 85 - Screenshot of the data capturing process. The dots on the screen correspond to the markers taped onto the human body.

14.3 Experimental Results

14.3.1 Human Perception

A user study to examine human perception of the motion-capture data was performed in order to provide context for machine learning experiments, as well as to validate the collected data. A subset of forty randomly ordered files from the database, with an equal proportion of each emotion and subject, were presented to

each subject as point light displays. In these point light displays, only the fourteen marker points are present (without stick figure lines) and the movement of the subject's emotion for a ten second period is portrayed. Point light displays were used as they directly correspond to the data provided to the automatic classifiers and their perception is not affected by other semantic cues such as facial expressions.

Sad	Joy	Anger	Fear	← Classified As
95	0	2	3	***Sad***
0	99	1	0	***Joy***
1	12	87	0	***Anger***
0	2	7	91	***Fear***

Figure 86 - Confusion matrix of human perception of 40 point light displays portraying 4 different emotions. Average recognition rate is 93%.

A group of ten subjects were tested in classification of these forty point light displays. A confusion matrix from results of this experiment is shown in Figure 86. An average recognition rate of 93% was achieved. It is worth noting that by watching a series of fourteen moving points humans can accurately identify representations of four different human emotions! This is probably achieved by looking at the dynamics and statistics of the motion parameters, which is what we use for features in the automatic system.

14.3.2 Machine Learning Experiments

From the human perception experiment described in Section 14.3.1, it can be seen that motion-capturing preserves the information necessary for identifying four emotional representations. The next step was to see if machine learning algorithms could be trained on appropriate features to correctly classify the motion-capture data into the four emotions. This section describes the feature extraction process followed by experiments with a variety of machine learning algorithms.

14.3.2.1 Feature Extraction

After the raw data is exported from the VICON system, feature extraction algorithms are run using a custom built MATLAB program for importing VICON data and extracting features. After experimentation the following dynamics of motion features were selected for training the classifiers. There were fourteen markers, each represented as a point in 3D space, $\boldsymbol{v} = [x,y,z]$, where x, y, z are the Cartesian coordinates of the marker's position. In addition, for each point the velocity (first derivative of position) $d\boldsymbol{v}/dt$ and acceleration (second derivative) $d^2\boldsymbol{v}/dt^2$ were calculated. As we are mainly interested in the dynamics of the motion over larger time scales, we consider the mean values of velocity and acceleration and the standard deviation values of position, velocity and acceleration. The means and standard deviations are calculated over the length of ten-second duration of each emotion depicted. Although it is likely that alternative feature sets could be designed, the classification experiments described in the next section show that the proposed features provide enough information for quite accurate classification results.

14.3.2.2 Machine Emotion Recognition Experiments

Five different classifiers were used in the machine learning experiments: a *logistic regression,* a *naïve bayes* with a single multidimensional Gaussian distribution modeling each class, a *decision tree classifier* based on the C4.5 algorithm, a *multi-layer perceptron backpropogation artificial neural network,* and a *support vector machine* trained using the Sequential Minimal Optimization (SMO). More details about these classifiers can be found in [72]. Experiments were performed using *Weka* [72], a tool for data mining with a collection of various machine learning algorithms.

The column labeled "All" on Figure 87 shows the classification accuracy obtained using ten-fold cross-validation on all the features from all the subjects and corresponds to a "subject-independent" emotion recognition system. The

column labeled “Subject” shows the means and standard deviations of classification accuracy for each subject separately using ten-fold cross-validation and corresponds to a “subject-specific” emotion recognition system. The last column labeled “Leave One Out” corresponds to the means and standard deviations of classification accuracy obtained by training using four subjects and leaving one out for testing.

Classifier	All	Subject	Leave One Out
Logistic	85.6 %	88.2%+-12.7%	72.8%+-12.9%
Naive Bayes	66.2 %	85.2% +- 8.8%	62.2%+-10.1%
Decision Tree (J48)	86.4 %	88.2% +- 9.7%	79.4%+-13.1%
Multilayer Perceptron	91.2 %	92.8%+-5.7%	84.6%+-12.1%
SMO	91.8 %	92.6%+-7.8%	83.6%+-15.4%

Figure 87 - Recognition results for 5 different classifiers.

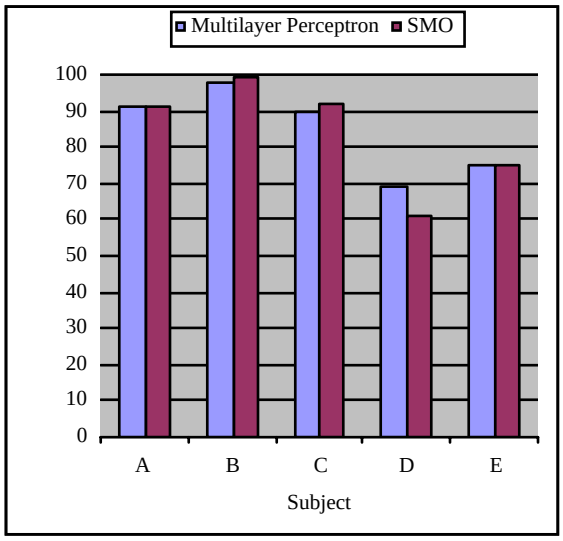


Figure 88 - Graph showing “Leave One Out” classification results for each subject using multiplayer perceptron and support vector machine learning classifiers.

Sad	Joy	Anger	Fear	← Classified As
114	0	2	9	***Sad***
0	120	4	1	***Joy***
2	3	117	3	***Anger***
10	3	4	108	***Fear***

Figure 89 - Confusion matrix for "subject independent" experiment using support vector machine classifier.

Figure 89 shows a confusion matrix for "subject independent" using the SMO classifier. As can be seen comparing the confusion matrix for human perception and automatic classification there is no correlation between the confusion errors indicating that even though computer algorithms are capable of detecting emotions they make different types of mistakes than humans.

In all the experiments the support vector machine and the multiplayer perceptron achieved the best classification results. It should be noted that training was significantly faster for the support vector machine.

14.4 **Summary**

We have presented a system for machine emotion recognition using full body skeletal movements acquired by the VICON motion capture system. We validated our data by testing human perception of the point light displays. We found that humans achieved a recognition rate of 93% when shown a ten second clip. From our machine learning experiments it is clear that a machine achieves a recognition rate of 84% to 92% depending upon how it is calculated. The SMO support vector machine and multiplayer perceptron neural network proved to be the most effective classifiers.

For a music application, after training our machine learning algorithms we ran the data from the violin player shown in Figure 90. This was a ten second clip of a composition playing a melody in the major scale. One can also see from the picture that the performer is smiling. Too our surprise, the machine classification

of this clip was happy. However, the machine only got this correct because it was trained to think that fast movement (dancers jumping up and down) is happy, and the performer happened to be performing at a fast tempo. In these types of experiments it is important to understand why the machine derives a certain answer.

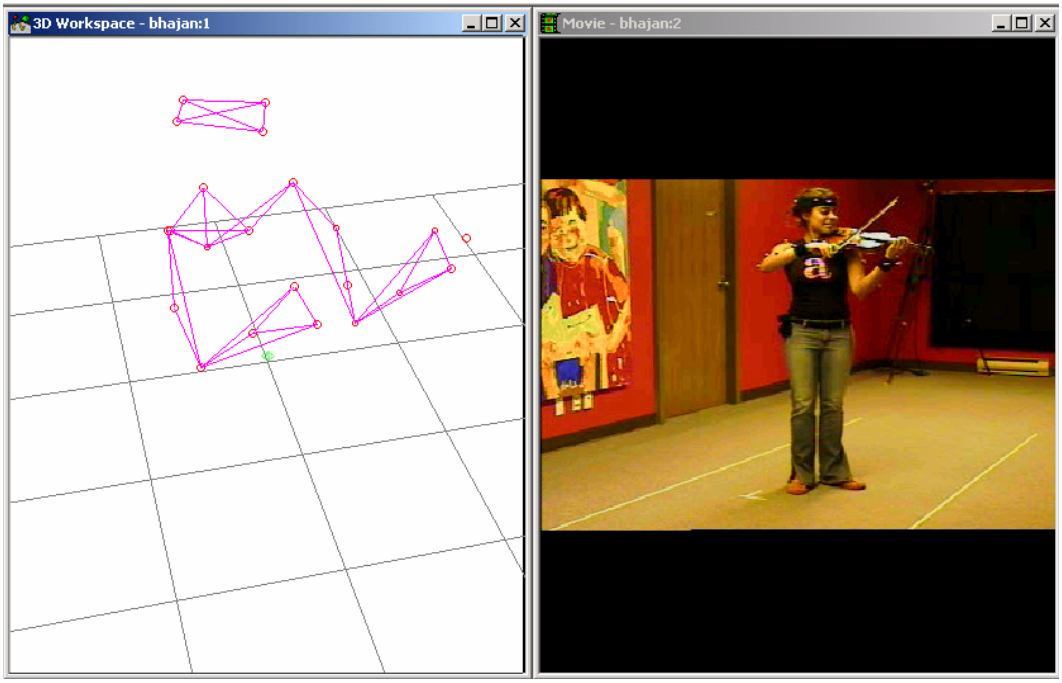


Figure 90 - What emotion is the violin player portraying?

There are many directions for future work. We are exploring the use of different feature extraction techniques. We also are collecting larger databases of subjects including more intricate detail of facial expression and hand movements. Increasing the number of emotions our system classifies to include disgust, surprise, anticipation and confusion are planned upgrades in the near future. We are moving toward a real-time multimodal system that analyzes data from microphones, video cameras, and the VICON motion sensors and outputs a meaningful auditory response. We believe that affective computing will play a major role in “intelligent” music interaction systems in the future.

Section V

Integration and Conclusions

Chapter 15

15 Integration and Music Performance

Chronological Performance Journal

This chapter is a chronological diary of progress on experiments with musical performances created using the technology described throughout this dissertation. It discusses how the research goes from the laboratory to the concert hall. This is where the real world problems and experiences are obtained, generating ideas for solutions and inspiration for new directions.

15.1 April 12, 2002 - *ETabla* in Live Performance

One of the goals of the *ETabla* project was to make an instrument that can actually be used to create an audio and visual experience that expresses the feelings of the performer and enamors the audience. The premier performance of the Electronic Tabla was held on April 25th, 2002 in Taplin Auditorium, Princeton University. Princeton undergraduate and graduate students joined faculty members and alumni in a concert mixing music from India, Africa and America,

with electronic grooves and beats. Video clips can be found online[38]. (See Figure 91)

Figure 91 - The *ETabla* in a live concert. Taplin Auditorium, Princeton University, April 25, 2002.

The *ETabla* premiered in a traditional North Indian classical song playing a Tin Taal, the traditional rhythmical cycle of sixteen beats (see Appendix A for more information). The *ETabla* was also featured in a song with an artist playing the Roland GrooveBox, an instrument that by its nature very accurately keeps time. The playability of the *ETabla* easily held up in performance with the rhythmically precise drum machine. Another piece in the concert was the "Dissonance Ritual", where the *ETabla* created atmospheric sound-scapes, triggering long lasting electronic samples. From the night's performance, the practical usability of the *ETabla* was demonstrated in accompanying compositions in a variety of musical genres. Those in the audience gave informal positive feedback on the visual feedback system projected on a screen behind the performers, as well as the switch between traditional Indian Tabla sounds and the novel electronic sounds that the *ETabla* could trigger.

[38] Available at: http://www.karmetik.com (January 2007)

15.2 June 3, 2003 - The Gigapop Ritual

The *Gigapop Ritual* was a live network performance between McGill University in Montreal, Canada, and Princeton University in New Jersey, USA. This live collaborative musical performance, weaving cyber electronics and Indian classical tradition involved high-bandwidth, bi-directional real-time streaming of audio, video, and controller data from multiple sources and players at both sites, using the GIGAPOPR framework. In composing the piece we took into account a saying by Atau Tanaka: *"Latency is the acoustics of the Internet"* [169]. We composed a piece that was appropriate for this aesthetic.

15.2.1 The composition

We composed the piece to explore multiple areas of performance over a network, using the traditional structure for North Indian classical music, as well as taking into account the framework of the network itself. The first section, known as *Alap*, was a slow call and response section between two melody-making instruments (sitar in McGill, electric violin in Princeton). These two performers left space for one another to interact to different improvised themes and phrases. The second section, presented a precomposed melody (based on *Raga Jog* and *Jai Jai Vanti*), over a structured eight-beat rhythmic cycle known as *Kherva* (performed on tabla in Princeton). The challenge and solution in performing a melody (*Ghat*) over the network was to have a leading side, and a following side. The round-trip latency of 120 ms was about the same as the echo one would hear from the back wall of a 60 foot room. Playing with other performers removed by 60 feet is somewhat common (in marching bands, antiphonal choirs, and other musical settings), and this experience was made only slightly more challenging by the large amount of equipment to be set up and tested. The performers in Princeton were the leaders, and once the data arrived in McGill, the Canadian performers simply played along,

reacting to what they heard. The third section was a free-form improvisation where musicians explored the network performance space. Performers used their custom built digital interfaces to create diverse computer generated sounds, while still focusing on interacting with the performers on the other side of the network.

Each performer left enough space for others to react, and no one played anything unless it was a response to a 'call' from another musician. Thus we were able to create a spiritual tie using the two host computers at two geographical locations, connecting performers in the Pollack Concert Hall of McGill University with performers in the Princeton Computer Science Display Wall room for a 2003 New Interfaces for Musical Expression (NIME) Conference performance. Video clips of performance can be found online[39].

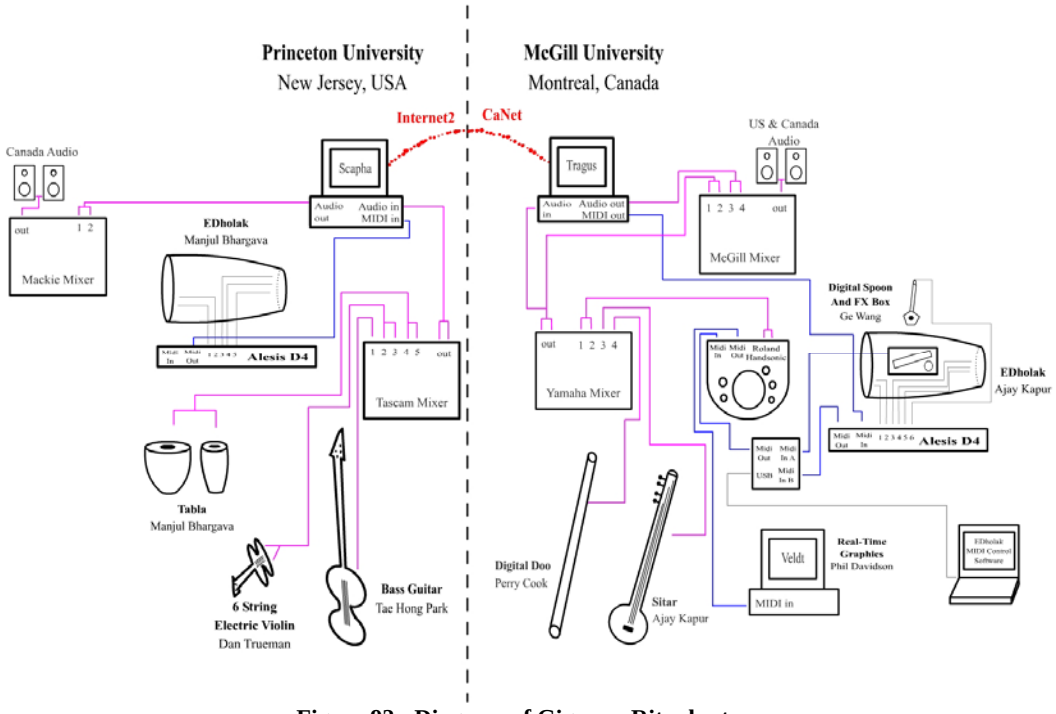


Figure 92 - Diagram of Gigapop Ritual setup.

[39] Available at: http://www.karmetik.com (January 2007)

Figure 93 - Gigapop Ritual Live Performance at McGill University with left screen showing live feed from Princeton University and right screen showing real time visual feedback of veldt.

15.2.2 veldt visual representation

Our intent was to create an environment in which the actions of both drummers were visible and distinguishable. Our implementation for this concert was to allow two players to interact through a sculptural metaphor. Using a dynamic geometry representation to allow modifications to the structures in real time, the two performers interacted through a series of operations to create a visual artifact of their drum patterns. Their strikes were dynamically mapped to a series of geometric operations that generated, deleted, deformed or detached elements of the structure and generated unique artifacts from the rhythms they played. In Figure 38 we see structures that have evolved under different mapping rules. In Figure 38 (left), for example, we chose a mapping that created smaller, separate elements rather than building from a central structure as in Figure 38 (middle). In Figure 38 (right), we chose rules which resulted in a solid, sheet-like structure. To add a convincing physical response to the addition and alteration of new elements, we used a mass-spring model to apply and distribute forces as the structures developed. In these figures, the actions of the drummer bend and distort the figure,

while secondary forces try to smooth and straighten the figure, like a plucked string whose vibrations decay to rest.

To represent the shared performance space, we experimented with several different forms of visual 'interaction' between the signals received from two performance spaces. To begin, we assigned the two drummers to separate visual spaces: one drum would excite patterns as in the *ETabla* performance, while the second was assigned to build structures. We then designated both performers as builders, so that their rhythms would build upon one another. In the next section, one performer's strikes would build while the strikes of the second would rattle or delete those structures.

15.3 June 4th, 2004 – *ESitar* Live in Japan

The *ESitar* was premiered at the Listening in the Sound Kitchen Computer Music Festival at Princeton University in November of 2003, an 8-channel composition "Saraswati's Electro-Magic" was performed in collaboration with Phil Davidson and Ariel Lazier. It was performed again as a stereo composition on June 4th, 2004 at the International Conference for New Interfaces for Musical Expression in Hamamatsu, Japan.

In the performance, all of the signals generated by the *ESitar* were sent into a computer and captured by the *pure data* program. This information could be used in many other ways, but *pd* was chosen because of the ease of capturing and manipulating the control information within the *pd* environment. The following four settings describe a number of different *pure data* patches written to demonstrate how such *ESitar* controller information can be used.

Figure 94 - "Saraswati's ElectroMagic" Performances at Princeton NJ and Hamamatsu, Japan.

15.3.1 Slide Sitar

The slide sitar patch was modeled after the sound of a slide guitar. It is a simple module that consists of a bank of oscillating comb filters. The filters have a resonance that is determined by the frequency indicated by the fret information. The control information from the fret is used to change the resonance of the filters. We also use thumb pressure to control the amplitude of the resonance oscillation. Here we interpret thumb pressure as a measure of intensity of the performance. The intensity of each pluck can be heard through the amplitude of the filters' oscillations. This is a very simple use of the control information, but such an effect could not be obtained without the detailed information provided by the *ESitar*.

15.3.2 Sympathetic Pitch

The sympathetic pitch patch plays rolls of a sampled sitar sound at the pitch indicated by the fret. In this case we mapped thumb pressure to the volume of the roll and head tilt to length of the notes and speed of the roll. Because the sounds produced do not directly correspond to the acoustic sound, this is more of a complementary patch. This is apparent when the performer increases the pressure on the thumb FSR on a beat where the strings are not strummed. In these

instances the roll becomes more prominent and can function as a rhythmic replacement for the sitar when the acoustic sound is not heard.

15.3.3 Ring Modulation and Delay

This patch also takes into account the fret information, setting the frequency of the modulation according to the pitch indicated by the frets. This patch produces a distorted sound that would not be possible to create without the accurate pitch information provided by the controller. We also set up a delay system controlled by the head and the thumb. The system allows the musician to capture output from the ring modulator and then play the material as a solo loop or in combination with other loops. The head tilt controls which loops are played or filled with new sound, while the thumb controls if or how much sound should be played or stored into the loop buffers.

15.3.4 Analysis/Re-Synthesis

Our last example is a simple analysis/resynthesis patch. We use fret information and pitch information to identify which partials are generated by the struck note. Simply taking the strongest partials would not lead to the same results because of the sympathetic strings that are constantly resonating. Once the strongest partials are chosen, sine waves with very short envelopes are synthesized which together form a disjointed representation of the acoustic sitar sound. The volume of the resynthesis is scaled to the sound of the acoustic sitar and then controlled by head tilt. We warp the volume distribution of the detected partials using thumb pressure. When greater thumb pressure is applied, the lower partials are given more weight and are more easily discernable.

15.4 November 18, 2004 - *ESitar* and Eight Robotic Turntables

The first experiment with working with robotic musical instruments was interfacing the *ESitar* with Trimpin's Eight Robotic Turntables (Figure 95). *ChucK* 1.1 was used to map gestures captured on the *ESitar* to control parameters on the robotic turntables. The thumb sensor was used to control scratching of two of the turntables. Forward signals were triggered when the first derivative of the thumb sensor was greater than zero, while reverse signals were triggered when the derivative was less than zero. The other six turntables were pre-programmed to play a pre-composed rhythm for accompaniment.

Figure 95 - *ESitar* Interfaced with Trimpin's Eight Robotic Turntables.

In this experiment, the robots were either pre-programmed to play certain motifs or the human had direct control on what the robots were playing, thus turning the robots into expensive, hard to make, visually pleasing, acoustic synthesizers. There was no human to computer improvisation. However the success of this experiment creates motivation for exploring mechanical musical systems and how they can be used to make new music with a North Indian flavor.

15.5 April 18th, 2006 – *ESitar* with DeviBot

The second experiment with robotics was to interface the *ESitar* with the first version of the custom built *DeviBot*. *DeviBot* at this stage was designed using three of Eric Singer's *ModBots [159]*. One Bot was attached to a Chinese gong, another to a small Indian frame drum, and the last to a large Indian frame drum.

Figure 96 - DeviBot and *ESitar* on Stage for First Time April 18th, 2006.

All software for this experiment was written in *ChucK*. As the language had evolved since our last experiment, it was possible to create modular code, with classes for the *ESitar*, as well as for the *DeviBot*.

The goal of this system is to generate a variation of a rhythmic accompaniment which evolves over time based on human performance. The first part of the piece with the robots was to perform "Clapping Music" by Steve Reich[40] on the two frame drums. In Clapping Music Reich uses beat point modulation to create changing rhythms by having a single rhythm shift out of phase with itself one eighth note duration at each modulation.

The next step was to set up a database of musical rhythmic phrases which each instrument could perform. We take a symbolic approach so that robotic instances can be triggered as well as different sound samples for testing purposes. Queries for the database are generated by sensor data captured from the human

[40] Steven Reich, Clapping Music (New York, NY: Universal Editions, 1972).

performer. The databases are time and tempo locked to each other using global variables. These global variables can be changed at any time, so that the machine can speed up or slow down based on the human's performance. For the initial experiments, each instrument had four rhythms which it could retrieve. In this composition, tempo from a human performance was not deduced, so a set tempo was programmed. Figure 97 shows an example of how the system can be mapped.

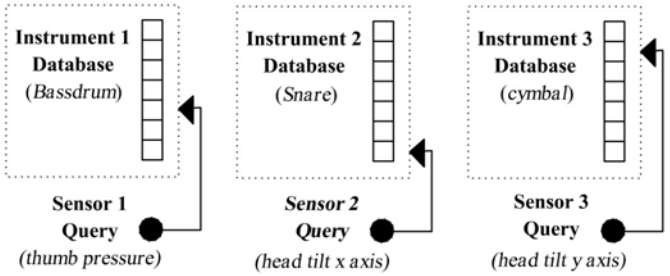


Figure 97 - MIR Framework for Human/Robot performance.

One issue to address is how the queries are generated. There are many algorithms which can be explored; however the main philosophical question is whether or not the human should have full control of the machine's performance. Another observation was discerning the difference between robotic accompaniment and sample-based accompaniment. There are many audio effect algorithms developed which can be controlled by sensor data from the human performer to create a vast amount of variety in machine performance. With the robots created to date, there is only one dimension of expression which can be controlled. Thus more work needs to be done on the robots to give them more degrees of freedom for expression.

15.6 November 6th, 2006 – *ESitar* 2.0 with *MahaDeviBot*

This experiment used the new model of the *ESitar* designed with the PIC microchip, and the beginning of the new custom built *MahaDeviBot* as seen in Figure 98. The new robot was constructed out of aluminum, a huge upgrade from

our first wooden prototype, which had unreliable fixtures that held both the instruments and the electronics in place. At this point, the robot has four Indian frame drums which it could strike. A Tin Taal rhythm and variations of a sixteen beat cycle were programmed into the memory of the robot with control signals coming entirely from the *ESitar* control box. The same frame work described in the April 18th, 2006 entry was used to generate variations of the performance rhythm of the accompanying robot. New features added to *MahaDeviBot's* repertoire were control of the global tempo from a potentiometer, and the ability to turn each drum on or off at any time using buttons on the *ESitar* control box.

Figure 98 - *MahaDeviBot* and *ESitar* in Concert on November 4th, 2006 in MISTIC Annual Concert Victoria BC.

As of now, the *MahaDeviBot* only has one timbre, so it is difficult for it to accent the "one" of a cycle. The *MahaDeviBot* needs more instruments to generate more elaborate rhythms with diverse timbres. Experiments exaggerating the dynamic loudness of robotic strikes proved fruitful; however, as a learning tool for students, the "one" should be pronounced more obviously. One idea is to build the robot a hand in which it can mechanically display the traditional hand clapping of north Indian drumming, marking the *"Sam"* and *"Khali"* (See Appendix A for more information). Another idea is to attach microphones or piezos to each drum. It would be interesting to do signal processing on the audio signal of the drum, and morph it using sensor data from the performer. Also, sometimes the arms of the instrument detach themselves during performance. If the machine had intelligence that a "strike" message was being triggered, yet mechanical problems occur not allowing the beater to strike the drum, then it should have "intelligence" turn that arm off. Piezo sensors could easily be used to

solve this problem. These sensors could also be used for automatic calibration of volume.

In terms of machine musicianship, there are three goals which need to be addressed. First, tempo from the human performer must be acquired. This can be accomplished by using data from the thumb sensor in real-time, in combination with audio signal. Second, the machine should have a sense of the performer's "one". This can maybe be accomplished either through dynamic analysis of performance data, or adding a new sensor to the *ESitar* which captures the downward stroke thru the sympathetic strings, which occasionally occurs on the "one". Third, the machine should be able to recognize *"Tihai-s"* performed by the human, which are melodies which are repeated three times to mark the ending of a section or song.

15.7 February 5th, 2007 – Meeting with Trimpin

After three months of development of the lower four frame drum construction, a major issue was that the arms would fall apart (screws fall out, zipties break loose, and rods fall out) within the first ten minutes of performance. This motivated work on improving structural integrity the *MahaDeviBot*. Metal rods were hammered into the aluminum blocks and secured with small set screws. A new mechanism for connecting the solenoid's shaft to the aluminum block was invented using a ball-baring system inspired by parts used for RC car construction. After these upgrades were installed, the robots were tested for four hour periods without disassembling. This made them ready for the next test, a meeting with Trimpin.

Trimpin gave incrediblely constructive criticism, based on his 30 years of experience in working with musical robots. He first pointed out that the solenoids in their current position are working against gravity, which makes them much

slower. He also stressed the importance of using flexible parts. When the drums and beaters are tied securely to the metal frame, all striking generates vibrations that permeate through the entire structure which can cause unwanted sounds and weakening of joints. Using bumpers, felt and other rubber insulation techniques reduces resonant frequencies in the frame and ensures more natural and human-like responses. Trimpin also explained that using 20/20[41] slotted aluminum is a wise choice, as he has been using it for decades, but that 15/15 that is used for *MahaDeviBot* is complete overkill and the system would be secure with 10/10 pieces.

We then discussed the use of different solenoids and power usage. The ratings of the solenoids are obviously important. 15-30 ohm solenoids are appropriate for striking applications, whereas 70 ohm solenoids are suitable for damping applications. The lower the resistance of the solenoid, the higher the current pulled. Our system currently runs with 15 ohm solenoids which draw close to 1.5 amps. Our power supply is only 3 amps, so when all four solenoids are triggered at the same time, there can be problems. This will also cause problems when thirteen solenoids are all in use in the future. Thus a new power supply must be acquired. Trimpin also explained that using power supplies with two leads, 24 volts and ground, and deriving 5 volts with circuitry, as currently built for the system, can cause problems in the future. He recommends a supply that has three lines, regulated 24 volts, 5 volts and ground. The current configuration will heat up the 5 volt power regulator. To lower the chance of problems, he attached a clip to cool the device, by increasing the surface area of the metal on the regulator.

[41] Available at www.8020.net (July 2007).

15.8 March 11th, 2007 – National University of Singapore Concert

Surprisingly, after one month of the *MahaDeviBot* being stored in a suitcase in Professor Lonce Wyse's office and while the *ESitar* went on a backpacking trip through the southern beaches of Thailand, everything still functioned properly for a big concert at the Annual NUS Arts Festival. This was a significant feature for the *ESitar 2.0*, as *ESitar 1.0* frets would have to be resoldered after every show. The robot performed just as well before and after the storage, without any practicing needed too keep up its' "chops". This is in stark contrast to the human performer, who left his computer behind and only had the *ESitar* to practice hours everyday.

The sound check for the show was ten hours long. Drums were raised and a stereo pair of microphones was placed underneath them. After EQing, the robot drums sounded perfect, which is a paradox because the desired sound was coming through speakers, which kind of defeats the whole point of one of our original goals. But aesthetically, this is the sound we have been searching for from the beginning. Also, we setup the MacBook Pro's built-in video camera to show half human arm and half robotic arm to the audience. Of course the image was delayed using Photo Booth, but allowed audience members in the back to see the intricacies of the movements of the mechanical parts.

Figure 99 - *MahaDeviBot* being controlled with the *ESitar* at NUS Arts Festival in Singapore.

The concert lasted fifty minutes. The first phase was an alap section introducing *Raga Yaman*. Here a "butterfly effect" was performed by the robot. The idea was to show the audience that the robot could actually move, in relation to the performance of the human performer. The robots arms would flutter at a very high rate and low velocity so that the mallets do not strike any drums. The fret position would determine which arm would move, while the thumb pressure would excite the fluttering. The next section used an audio driven program, showing the audience how the robot react to the sound of the human sitar player. Using Professor Perry Cook's "leaky integrator" *ChucK* code, the robot would strike a drum whenever the *ESitar's* audio signal went over a set threshold.

Finally a rhythm was introduced. The robot played a simple *Tin Taal* while the a composition by Vilayat Khan was performed on the *ESitar*. The challenge with the *Tin Taal* was for the performer to keep in time with the machine's sixteen beats. The robot would accent the "one" by performing dramatic loud dynamic changes. The *Tin Taal* rhythm would vary based on the performer's gestures, however more drastic variations need to be programmed in with more "intelligent" query mechanisms.

The robot also performed a *Kherva* beat, as well as a tribal drum groove with many variations. The entire concert, the tempo would keep increasing and increasing controlled by a knob on the *ESitar's* controller box. One challenge was getting the robot to switch to a new beat or even end a song. I practiced triggering the robots to stop (using a button) while playing a melody in time, but this has its difficulties during the performance, and a new system must be created. The robot also needs a call and response program. Another dream would be to build a stage-ready system to gather rhythm from a human and store it in its "memory" database in real-time.

15.9 June 9th, 2007 – *ESitar* and *MahaDeviBot* Live in New York City

The final concert before defending my dissertation was held at the International Conference for New Interfaces for Musical Expression (NIME) in New York City. Many improvements were made to the *MahaDeviBot*, including the addition of eight more arms, using the Trimpin Hammer and Trimpin BellHop mechanisms. These devices were much more stable, quiet, reliable, and professional looking. They were used to strike a variety of instruments including bells, shakers, wood shingles, gongs, and finger cymbals. This gave the entire robotic instrument a broader range of frequency and timbre. A head was also added to the *MahaDeviBot*, which could bounce up and down using a solenoid-based design. This allowed the human performer to visually comprehend the derived tempo of the machine.

Curtis Bahn was invited to perform for this concert as he had built his own version of an *ESitar* as well as an Electronic Dilruba, which is bowed Indian instrument combining the characteristics of a sitar and a cello. Curtis created an electronic bow with a two-axis accelerometer. Accelerometer data from both instruments were used to query half of the rhythms performed by the *MahaDeviBot*.

The composition began with an alap section introducing *Raga Yaman* as well as the electronics to the audience. Curtis had a variety of electronic sounds controlled by the bow which also controlled the delay lines of the audio from the Dilruba. The *ESitar* was used to control the “butterfly effect” program and gently move each of the parts on the *MahaDeviBot*, portraying to the audience that mechanics were being controlled by the hyperinstrument. The piece evolved to a *Tin Taal Theka* in which both performers played a *Ghat* by Vilayat Khan and traded solos. Sensor data from both instruments queried rhythmic variations. This sixteen-beat section ends abruptly, the musicians go silent, and all that is left is

the head of the *MahaDeviBot* bouncing up and down. Visually being conducted by the moving mechanical parts, the musicians come back in time, and the robot switches to the *Kherva* rhythm. *MahaDeviBot*, slowly increases its tempo as the jhala melodies become more developed and full. Electronics, harmonies and exotic interaction interchange until a fast *Tihai* is signaled to end the compostion. The robot successfully stops in time with performers, using signals from the sensor data (an upgrade from pushing a button). This was the final successful experiment of the entire body of work. A beautiful ending of one era, and beginning of the next.

Figure 100 - Performance at New York University at the International Conference on New Interfaces for Musical Expression June 11, 2007.

Chapter 16

16 Conclusions

Towards the age of the cultured machine

Victorian painter Emily Carr once said, *"Artist…there is no realization… only momentum towards fulfillment, something that indeed we cannot do individually, separately, only as a complete spiritual solidarity…."* [19]. The body of work described in this dissertation was truly an artistic venture calling on knowledge from a variety of engineering disciplines, musical traditions, and philosophical practices. It also called on the collaborations and expertise of many professionals, professors and students. The goal of the work was to preserve and extend North Indian musical performance using state of the art technology including multimodal sensor systems, machine learning and robotics. The process of achieving our goal involved strong laboratory practice with regimented experiments with large data sets, as well as a series of concert performances showing how the technology can be used on stage to make new music, extending the tradition of Hindustani music. This concluding chapter will present what was learned from all our research and experimentation, including a summary of contributions, discussions on techniques, the challenges faced, the advantages and disadvantages of the interdisciplinarity of this work, as well as future directions.

16.1 Summary of Contributions

This dissertation made contributions in the areas of musical gesture extraction, musical robotics and machine musicianship. However, one of the main novelties was completing the loop and fusing all three of these areas together. Using multimodal systems for machine perception of human interaction and training the machine how to use this data to “intelligently” generate a mechanical response is an essential aspect of human machine relationships in the future. The work in this dissertation presented research on how to build such a system in the specific genre of musical applications.

Much of the research in the area of computer music has primarily been based on Western music theory. This dissertation fully delves into applying the algorithms developed in the context of North Indian classical music. Most of the key contributions of this research are based on exploring the blending of both these worlds.

We presented the first hardware devices to capture traditional finger position and timing information for North Indian drumming, namely the tabla and dholak. We further extended this technology to create the first multiplayer Indian music performance system by networking two drums together. We created the first modified Indian classical string instrument to have sensors which capture performance gestures for archival of performance technique and used for creating modern multimedia concerts. We also conducted research using wearable sensors on the human performer to create a framework to obtain more information about how to preserve intricacies about the posture and performing technique from North Indian classical musicians.

In the field of musical robotics, we presented the first mechanically driven drum machine to perform Indian classical rhythms for human-to-robot performances in conjuction with multimodal machine based perception. We also

included the first detailed experimentation and documentation on how to use solenoids for musical performance.

We use machine learning and advanced signal processing techniques to further preserve and extend Indian classical music. We presented research on the first system for using multimodal acquisition for sitar performance for obtaining tempo-tracking information using a Kalman Filter. We presented a novel system to use retrieval techniques for generating robotic drumming accompaniment in real-time. We presented the first software to automatically transcribe performance of a sitar performer using multimodal acquisition methods. We also described the first method to create an audio-based “virtual sensor” for a sitar using machine learning techniques. Finally, we presented the first experiments on using motion capture data for machine-based emotion detection in the field of affective computing.

16.2 Discussions on Techniques

Through out the years of building this technology we have experimented with several sensors, microcontrollers, music protocols, music languages, sound mappings, graphical feedback mappings, robotic systems and machine musicianship techniques. The approach for all of these areas of research has evolved as we experimented with new approaches and techniques as each hardware device or software was created. This section presents a discussion on what was learned in each of these areas.

16.2.1 Sensors

We found that using FSRs on drum controllers did not obtain a quick enough response time, especially for Indian drumming, where finger strikes occur at a very quick rate. However, the gestural footprint obtained about force and position is very useful. Using piezo, we did not obtain enough information about force and

position, but the response time was fast enough for live performance. Thus using a combination of the two, as seen in the digital spoon of the *EDholak* is a perfect balance, where timing specific impulses can be triggered immediately and FSR readings can effect the other variables which are not so time dependent, but still expressive.

Using simple circuitry and a few sensors as seen in the *ESitar* to obtain gestural data of a traditional instrument has also proved to be successful. This way a trained musician in the classical form of the instrument can easily perform with the digitized version and learn new techniques with more ease than with that of an acoustically quiet instrument, which models tradition, but relies on synthesis algorithms or samplers to generate sound.

Further, using sensors on the body, as seen on the headset of the *ESitar* controller, and all the experiments with the *KiOm* and *WISP*, gathers extremely useful gestures which can be used to modify sound of a performer in real-time.
Sensor calibration, which was also programmed for the *ESitar* was a very important addition, enabling a variety of users to easily set their minimum, maximum and average sensor values for successful mapping.

Finally, sensors are very fragile and break with use over time. Thus designing robust systems for easy exchange of sensors saves a lot of frustration and time during sound checks.

16.2.2 Microcontrollers

Comparing the Parallax Basic Stamp IIsx, the Atmel AVR Atmega16, and the PIC microchip as microcontrollers for our technology, we prefer the PIC and Atmel. Programming in C as compared to PBasic (similar to Basic) allows for more complicated algorithms and onboard signal processing, which we would have not even tried to implement on a Basic Stamp IIsx. The fact that the Atmel code can be compiled using *gcc* allows development using Linux, Mac, and Windows. The eight 10-bit Analog-to-Digital converters, the faster processor

speed and other magic functions built in to the Atmega16 and PIC, also make them a logical choice for more complicated controllers which use more sensors and require more processing power.

16.2.3 Music Protocols

Converting the *ESitar* and *WISP* from communicating via MIDI to Open Sound Control (OSC) (Wright, Freed, Momeni, 2003) proved very useful for two reasons. First, we no longer had to down-sample our 10-bit Analog-to-Digital signals to 8-bit MIDI data streams. Second, as the developing team were at geographically different locations, using the OSC protocol that was optimized for modern networking technology made it easy to send controller data from one location (namely British Columbia, Canada) to the other (New York City, USA) using the internet.

16.2.4 Music Programming Languages

The research described in this dissertation used a large number of music programming languages to achieve a variety of tasks. Namely, *MAX/MSP*, *pure data*, *ChucK*, *STK Toolkit* and *Marsyas* were used (more information on each language available in Appendix E). From our experience, one must choose the right tool for the right job. If one wants to program a physical model of a sitar from scratch, then using *STK Toolkit* would be an obvious choice as the framework is completely set for that type of application. However, if one only wants to map controller data to parameters of a physical model, then using *MAX/MSP*, *pure data* or *ChucK* would all be appropriate choices based on personal preference. As of today, *Marsyas* proved to be the strongest language for doing detailed analysis of signals because of its large set of feature extraction algorithms, graphing tools, and ability to handle signals at different sampling rates. For live performance, most work was done first in *pure data*. We tended to choose academic software which is freely available and open source over

commercial programs such as *MAX/MSP*. *MAX/MSP* has the advantage of having a very large user base with many advanced features created by the community, however because the goal of our work is to share it with musicians in India, we do not want the high price of software to limit the accesiblity of our work.

Over the last three years, we have converted all our live performance code to *ChucK* as we believe it has the framework to help change the future of computer music systems. One of the strongest aspects of the language is its correlation to real programming, with its types, variables, arrays, operators, control structures, functions, and class structures. Thus, a student who has never learned how to program will have a much stronger understanding for real programming then by connecting lines to various boxes. One can also manipulate time very easily in *ChucK* using the “now” functionality and the use of shreds to control specific tasks that can be concurrently started and stopped at any time. The use of classes in the *ChucK* is also useful for group collaboration and development. Each controller built has a separate *ChucK* class, which can be easily shared and updated modularly. The *ChucK KiOm* class was jointly developed by the MISTIC team over a period of three months, where each member added the desired functionality through experimentation and experience.

16.2.5 Controller Sound Mapping

Correctly mapping the gestural signals obtained from the controllers is essential in producing meaningful sound that expresses the performers feelings.

Our first experiments with the *ETabla* were to trigger physical models of a Tabla using *STK Toolkit*. While we were very impressed with results of the sound and expressiveness, the time delays were too great for faster Indian rhythms. The *ETabla* MIDI signals also trigger sounds on any digital or analog MIDI sampler, such as the one on the R*oland Handsonic*. Thus an experienced tabla player can easily use their years of experience and training to trigger a myriad of different

sounds such as congos, djembes, bells, pot drums, and drumsets, using the *ETabla*. For a beginner tabla player, creating sounds such as a "*Na*" on a real tabla is very difficult and de-motivating. With the *ETabla*, a beginner can start creating sounds immediately giving them positive feedback to keep working on rhythm, theory, and finger strength.

One observation from the *ETabla* experiments was that simply triggering a single sound from a MIDI soundbank became very boring during performance. This motivated separating the rhythm making process from the sound production process in the creation of the *EDholak*. With this controller, one person supplies the beat while another musicians sole purpose is to modify the sounds being triggered, adjusting parameters such as the soundbank, pitch, color, depth, and pan. This proved to be very effective in performance space as deeper emotional content could now be expressed with the controller. Also, having software such as the *EDholak* MIDI Control Software to organize the order of desired soundbanks and different parameters of a sampler makes performances easier and allows the musician to focus on the music rather than memorizing where a specific patch is located.

We took a completely different approach in mapping sounds for the *ESitar*. The *ESitar* is the only instrument of the set that is not acoustically quiet when played. Thus we are modifying the sound of the real instrument, using the gestural data deduced from the microchip. We used *pure data* to achieve effects such as ring modulation, delays, additive synthesis, and sympathetic pitch, which overlay on top of the acoustic sound of the sitar during performance. Lately, we have been using *ChucK* to control MIDI messages in conjunction with commercial software such as GuitarRig and Ableton Live. This proved to be an easy way to obtain professional sound in a short amount of time. This method seems to fully keep the tradition of sitar performance intact while adding modified techniques using a computer. In the future, we will try to combine this method with physical modeling.

Further, with the *ESitar*, mapping was successful due to code written to record synchronized audio and gestural data, enabling the user to record a performance and then playback all the data, tweaking parameters to obtain desired effects and sound synthesis. Using *ChucK* and other languages of this nature, which allows full control in designing sound patches that can be customized to the performer's dreams, a user can utilize far more interesting synthesis techniques that are impossible to achieve with any sampler/sequencer.

16.2.6 Graphical Feedback

We render graphical feedback for the all the controllers using custom built software *veldt* (Davidson, Kapur, Cook 2003). *Veldt* is an application that was designed from the ground up for the purpose of visual expression and performance. We used veldt to generate graphical meaning, such as rhythmic impulses from the *ETabla* and *EDholak*, and melodic transcription from the *ESitar*. The incorporation of visual feedback in concert performance can provide the audience with another means of conceptualizing their experience of it, and an additional means of becoming actively engaged with the artist's performance. Beyond the context of concert performance, our system can also be used in a pedagogical context, allowing a student to receive a variety of feedback and coaching while learning to play these instruments.

16.2.7 Robotic Music

Overall, our main motivation in using a robotic system was our discontent for hearing rhythmic accompaniment through a set of speakers. In is our opinion, triggering pre-recorded samples does not have enough expressiveness, and physical models to-date do not sound real enough. However, having a machine perform acoustic instruments using mechanical parts has its disadvantages. We

must tune the machines' instruments for every show, which would not be necessary if we were just triggering "perfect" samples. Also, because of the nature of any mechanical system, there are imperferctions in event timings based on varying spring tension, speed and strength of previous strikes. However, this produces more realistic rhythms, as humans also have imperfections when actually "grooving".

Our experimentation with robotic systems for musical performance brought many familiar yet new challenges to working with sensors. A set of allen wrenches, screw drivers, plyers, a calliper and a dremel are carted to each performance along with a box set of extra springs, screws, washers, and spare parts. Our first designs had frameworks made of wood. This obviously is too heavy a material, and using aluminum is ideal because of its sturdiness and light weight. However, we learned from our initial prototypes that welding anything would be a mistake. All parts should be completely modular to allow for changes in the future. Thus designing our robots out of 20/20 T-slotted aluminum was a perfect material to accomplish all our goals of sustainability, modularity, mobility and professional appearance.

16.2.8 Machine Musicianship Techniques

The machine musicianship algorthims we have experimented with are still very primitative. We use machine learning techniques to help classify gestures and sounds into categories to help begin training a machine to "hear" as humans do. We even show how a computer can use machine learning to "hear" more details than a human in our regression experiments to create "virtual sensors". Using advanced digital signal processing techniques like the Kalman filter and pitch tracking we have shown how the machine can track tempo and automatically transcribe music with more information from a variety of sensors. This validates our building the *ESitar*, *KiOm* and *WISP* for preservation applications. However

all our work in this domain has not been implemented in real-time and has another revolution before it will be stage-ready.

16.2.9 Power

A constant challenge in building all the hardware devices has been powering the electronics. The *ETabla* and *EDholak* used 9 volt batteries which stuck out of their encasements. These always presented the danger of being accidently knocked off during performance. In addition, extra batteries would have to always be on hand, and a voltmeter to determine how many volts were left in each battery before performance. This motivated designing the *ESitar* with a 9-volt power supply that plugged directly into the wall. However, this presented the disadvantage of having yet another cable to set up. When beginning to design wearable technology it seemed that using batteries would make the systems more accessible. Switches were created on each *KiOm* so each user could conserve power. However, the 9-volt battery was the largest component in the *KiOm*, as well as the heaviest. The *WISP* addressed this issue by using a 3-volt lithium battery, but this was only achievable by having a USB powered wireless station which then sent gesture messages to the machine, and not a standard direct MIDI connection which will always take 5-volts. For the *MahaDeviBot*, each solenoid could draw from .75 to 1.5 amps of current. The power supply had a 3 amp rating, meaning that only 2 to 4 solenoids could be triggered at the same time. Also the MacBook Pro which is being used to run the software gets unbelievablely hot, due to power. Of course, the faster the machine, the hotter it gets. These combined problems have influenced the author to learn more about solar cells and other methods of producing sustainable energy for artistic pursuits.

16.2.10 Evaluation

A key concept through out this work has been finding methods for evaluating the success of our algorthims and techniques. We presented user studies on how an expert Tabla performer could use the *ETabla*. We presented our evaluation methods on the dynamic range and speed of the various solenoid designs of the *MahaDeviBot*. We also presented detailed analysis for each machine musicianship technique, especially in testing the success rates of various machine learning techniques. Evaluation is essential laboratory practice in the process of improving technique and allowing our research to grow beyond our laboratory and have global impact.

16.2.11 Preservation and Pedagogical Tools

The key goal of this research has been to build technology to aid in preservation of North Indian classical music. We have presented methods for recording synchronized data beyond just the audio signal, capturing more intricate gesture information from the master performing artist. It is also our goal to build pedagogical software to aid in more accessible and efficient means of dissemination of traditional techniques. In India, when one learns from a master, it is common to not be allowed to record the lesson, or even take written notes. One must only learn what one can obtain during that given time, by repeating even one phrase over and over to perfection. The student then goes to do their own *riyaz* (rehersal), until the meeting the next day. This system can work wonderfully well, but is restricted to only those who live in India near the masters. This research is presenting methods for globalization of this information, by building tools which can aid in bringing that knowledge from India and allowing students who live in North America, Europe, and other parts of the world to have equal access and opportunity to lessons from the masters.

16.3 Challenges of Interdisciplinary Research

There have been many challenges in handling the interdisciplinarity of this research. Firstly, having a large number of advisors is certainly fruitful in gathering a number of prospectives, but can be difficult when each has their own agenda for what direction the Ph.D. student should follow. Also, having to take courses in five subjects is difficult as one must learn the basics for each subject matter before being able to understand the full magnitude of the advanced courses. For example, when taking advanced digital signal processing, I would simultaneously go to the beginner level classes on signal processing to catch up on the material. The courses in mechanical engineering and psychology held similar difficulties.

Each discipline studied had its own interdisciplinarity forming a multi-level staircase of knowledge and/or confusion. In computer science, I had to learn all the music programming languages as discussed including *ChucK*, *Marsyas*, *STK Toolkit*, *MAX/MSP*, *pure data*, as well as the general languages like MATLAB, Basic, C, C++. To make matters even worse, different programs had to be run on different platform so I constantly had a machine for Windows, Linux and Mac OSX. It is impossible to remember all the details of each language and platform, so deriving methods of notes and quick look up tables of information was key to success. Electrical engineering held its own challenges. Signal processing algorithms first had to be coded off line, then converted to real-time, and then some had to be converted to work on board a microchip. Even the microchips used in this work were multi-dimensional, as we experimented with PIC, Atmel, and Basic Stamps. In mechanical engineering, the greatest challenges lay in the number of tools one had to learn to get a job done. I learned how to use a mill, lathe, drill press, dremel, as well as simple tools like a caliper.

The music component of this research held the most challenges. It was imperitive to practice everyday. But even here there were so many instruments to

practice including *ESitar*, *Etabla*, *EDholak*, *KiOm*, etc. To make it even worse, the instruments would keep changing as new software and hardware would be added. Performances took the most amount of energy. Anything that could go wrong would gone wrong, and learning to be prepared for the worst is the only way to be successful in using custom hardware on stage. Sensors and solenoids sometimes break, wires sometimes come loose, and even software sometimes crash (especially ones you did not program yourself!). Building sturdy transportation cases for the instruments proved to solve some issues, however when flying through the USA, one never knows what homeland security is going to open up and break. Only luck is involved here and good karma.

One of the main disadvantages of interdisciplinary study is that one does not become an expert in one discipline. Despite this fault and all the challenges there are many advantages that could not happen without interdisciplinarity. The very fact that there is not one discipline means that there is not a set of pre-composed rules or tradition of how research should be administerd. This allows for new research and in our case, the creation of new music and novel expressions for artistic endeavors. The successful interdisciplinary scholar quickly learns that they themselves cannot be the best at everything and will need the help of many specialists. Thus a key skill to acquire is social networking, collaboration, and leadership. Finally, the successful interdisciplinary scholar must learn to be efficient, manage time, organize workflow, schedule deadlines, and prioritize each moment.

16.4 Future Work

Future work includes a variety of directions. Overall, the technology built needs to be performed by masters in India, capturing data for analysis and preservation. Using the tools for pedagogical means has always been the chief motivator. Developing the technology into robust systems that can be used by novice

students with sleek GUI interfaces is a first step to accomplishing this goal. Further collaborating with professors, engineers, artists, musicians, and students will let this work breathe and take on a life of its own. I hope the future will allow those interested in becoming a master performer in North Indian music to use this technology and reach high heights in a shorter period of time. Also, as the technology progresses it is my dream to spawn new musical ideas and genres continuing to evolve North Indian music and computer music to new heights.

Section V

Appendix

Appendix

A

An Introduction to North Indian Classical Music

Raga, Theka, and Hindustani Theory

Once, a long time ago, during the transitional period between two Ages… people took to uncivilized ways … ruled by lust and greed [as they] behaved in angry and jealous ways, [while] demons, [and] evil spirits… swarmed the earth. Seeing this plight, Indra (The Hindu God of thunder and storms) and other Gods approached God Brahma (God of creation) and requested him to give the people a Krindaniyaka (toy) … which could not only be seen, but heard, … [to create] a diversion, so that people would give up their bad ways." [39] These Krindaniyakas which Brahma gave to humans included the Indian classical instruments used to perform Hindustani music.

Music is a medium to express emotional thought and feeling through "tone and time" [7]. Traditionally, these sounds are portrayed using rhythm, melody and harmony. In the case of Indian Classical music that is monophonic in nature, only rhythm and melodic formulas are used to express the emotions of the musician. This chapter serves as an introduction to Indian classical music theory. There are two systems of Indian classical music: Hindustani from the North, and Carnatic from the South. Though there are many similarities between the two systems, this

chapter will focus only on North Indian Classical music, which serves as the rules and traditional theory used for this dissertation.

A.1 Nad

Nad is a Sanskrit word that translates to "sound in its broadest sense usually conceived in metaphysical terms as vital power" [7]. Musical sounds have four characteristics: pitch, timbre, duration, and intensity. Pitch corresponds to frequency that is determined by the number of vibrations per second a wave propagates through the air. Timbre is described as the quality or character of a sound. Duration is defined as the time period of a sound. Intensity is defined as the strength of a sound.

In North Indian classical music, musical notes are described through seven main *swara-s*. They are known as: *Shadja* (*Sa*), *Rishab* (*Re*), *Gandhar* (*Ga*), *Madhyam* (*Ma*), *Pancham* (*Pa*), *Dhaivat* (*Dha*), and *Nishad* (*Ni*). Figure 101 visually represents these seven *swara-s* using a keyboard diagram, based on a C Scale[42]. One can equate these seven *swara-s* to the western solfege system (*Doh, Re, Mi, Fa, Sol, La, Ti*).

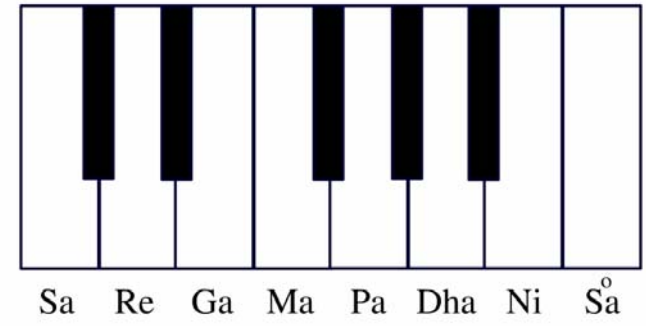


Figure 101 - The seven main *swara-s* shown using a C Scale.

By examining Figure 101, one can observe that there are black keys in-between *Sa-Re, Re-Ga, Ma-Pa, Pa-Dha,* and *Dha-Ni*. There is no black key in-between *Ga-Ma* and *Ni-Sa*. The black keys allow for *vikrit swara* (altered notes). *Shuddha swara* refers to pure, the natural notes described so far. It is possible to

[42] Orthodox Hindustani music cannot be performed on a harmonium or piano as the notes are set in place and not flexible based on *raga*.

make a note flat (*komal)* or make a note sharp (*tivra*). It is possible to have a *komal Re, Ga, Dha,* and *Ni,* which are notated by underlining the *swara* as follows: *Re*, *Ga*, *Dha*, and *Ni*. It is possible to have a *tivra Ma,* which is notated with a small mark over the a as follows: *Må*. Both *Sa* and *Pa* are fixed with no *vikrit* form, known as *achal* (immoveable *swara*). Higher and lower octaves are denoted with a ^{o} above and below the *swara* as follows: $S^{o}a$ and $S_{o}a$. The human vocal range covers three octaves: *mandra* (low), *madhya* (middle), and *tar* (high). Some string instruments can perform notes in the lower register known as *ati-mandra*, as well as the highest register known as *ati-tar*. Figure 102 portrays the twelve basic *swara-s* in more detail.

Full Name	**Short Name**	**Hindi Script**	**Solfege System**	**Scale of C**
Shadja	*Sa*	स	Doh	C
Komal Rishab	*Re*	रे		D^{b}
Rishab	*Re*	रे	Re	D
Komal Gandhar	*Ga*	ग		E^{b}
Gandhar	*Ga*	ग	Mi	E
Madhyam	*Ma*	म	Fa	F
Tivra Madhyam	*Må*	मे		$F^{\#}$
Pancham	*Pa*	प	Sol	G
Komal Dhaivat	*Dha*	धा		A^{b}
Dhaivat	*Dha*	धा	La	A
Komal Nishad	*Ni*	निं		B^{b}
Nishad	*Ni*	निं	Ti	B
Shadja	$S^{o}a$	सं	Doh	C

Figure 102 - Notation of 12 basic *swara-s*.

Each of the twelve *swara-s* also have an infinite number of frequencies in-between each interval. These are referred to as *shruti,* or microtones. "*Shru*" literally translates to "that which is audible". Because of the anatomy of the human ear, it is possible to hear only twenty-two discrete frequencies between each set of intervals. Thus there are twenty-two *shrutis* that come into play in North Indian Classical music. The use of these microtones are one of the characteristics that separate Indian music from other music in the world, and provides a complex, colorful palette for the true artist to use to draw subtle and/or extreme emotional content [7].

A.2 The Drone

Indian classical music is based on musical modes, where the meaning of each note is determined by its relation to the *adhar swara* (ground note). In western music, this is known as the tonic. This tonic, the *Sa*, is permanently fixed throughout the performance, setting a base foundation from which the melodies arise.

The *tanpura* is an instrument that has evolved over centuries, and is used to provide the drone for Indian classical performance. As shown in Figure 103, it has a hemispherical base made of a large hollowed out gourd (known as the *tumba*) which acts as the sound box and resonating chamber. The top of the gourd is an open surface, covered by wood, acting as a resonating plate (known as the *tabli*). The *Dand* is the stem, emerging out of the *tumba*, also hollow and serving as a resonating column.

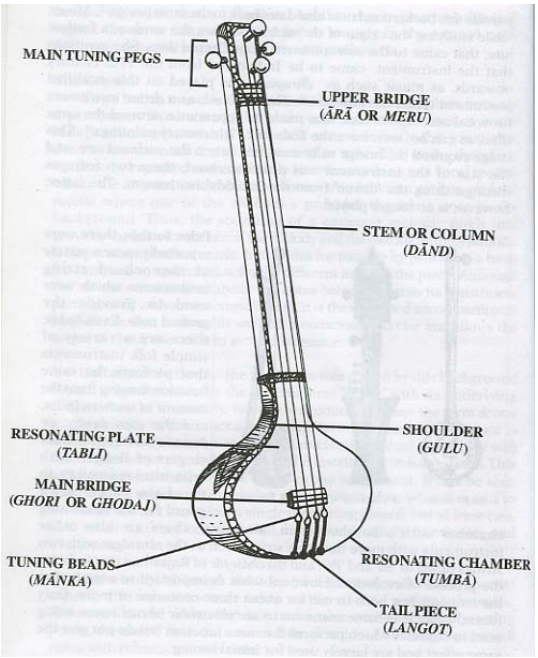


Figure 103 - The Tanpura and names of the parts of the instrument [7].

There are four strings, three made of steel and one made of copper to create the lowest pitch. Strings are typically tuned to $P_o a$ *Sa Sa* $S^o a$, but tunings change based on the raga being performed. There are four tuning pegs attached to four separate strings which extend from the tail piece of the base of the *tumba* (known as the *langot*), over the main bridge (*ghori*), along the *dand*, and over the *meru* (upper bridge) to the pegs. A performer tunes using the pegs, but can fine tune using tuning beads (known as *manka*) and *jiva*, which is silk, or cotton thread placed between the string and bridge, cushioning and affecting vibrations. The area where the strings meet the lower bridge is angled, giving the strings a larger degree of freedom to vibrate, creating the buzzing sound (known as *jawari*) uniquely characteristic of the *tanpura* [7].

A.3 The Raga System

The raga system in Indian classical music is the melodic form encompassing tonality, frequency, scale and the relationship between pitches. Each raga evokes a particular mood in the performer and listener.

There are two basic movements within any scale (*saptak*): (1) *Aroha* which is an ascending or climbing movement, and (2) *Avaroha* which is a descending or falling movement. Every raga is defined by a fixed set of unchangeable notes, and an ascending and descending order to which these notes can be performed.

A listener may still not be able to recognize a raga, even when knowing the notes, the *aroha* and the *avaroha*, because there are many ragas that use the same structure. A listener must distinguish a raga from the context and manner in which notes are used, which notes are stressed (*vadi* and *samvadi*), how notes are intonated, how *shruti-s* are incorporated, and by certain key phrases which indicate a particular raga. All of these attributes combined in a musical performance comprise a traditional raga. This is why this theory can only be learned by meditative hearing and repetition.

North Indian music has a number of guidelines which are used to define a set of notes as a raga. Firstly, each raga must have at least five notes. Secondly, each raga must have a *Sa* as the fundamental and cannot omit the *Ma* and the *Pa* simultaneously. A raga must use the full range of an octave, employing both tetra chords. The raga cannot take the *shuddha* and *vikrit* form of a note consecutively, however, one form can be used in ascending and another in descending. To clarify, a raga may have a komal *Ga* in the ascending scale, and a *shuddha Ga* on the descending, but both *Ga* and *Ga* cannot be in the ascending. There are also many aesthetic potentialities of a set of notes. There is a specific ornamentation, incorporating *shruti* (microtones) in the correct manner. The most important is the general rules about melodic movement by incorporating correct use of *aroha* and *avaroha*.

A raga which has five notes or pentatonic is known as *audav*, hexatonic is *shadav*, and heptatonic is *sampurna*. *Jati* is the name given to raga-s which have a different number of notes ascending as descending, for example *audav-shadav*, has five notes in *aroha* and six notes in *avaroha*. Further, the *aroha* and *avaroha* can be straight or twisted (*vakra)* [7].

A.3.1 Pakad

Pakad is the a term referring to the characteristic phrase which helps identify and define a raga. It is a catch phrase which is repeated several times through a piece for recognition. Synonyms for *pakad* in Hindi include *mukhya*, *anga*, and *swarup* which all mean main aspect or main form. However, it must be noted that not all raga-s have a *pakad*. *"In reality such phrases are very arbitrarily chosen by the authors…The truth is that there is not one single catch-phrase nor any definite number of phrases that form the core of a raga, because a raga is a fluent and dynamic whole [111]"*.

A.3.2 Vadi & Samvadi

In Indian classical music, each note does not have equal importance. This does not mean in terms of volume but rather in frequency of usage and relevance to defining a raga. The *vadi* is the most significant note where as the *samvadi* is the next significant. *Anuvadi* are the all the residual notes of the raga of less importance, where as *vivadi* are the set of prohibited notes not part of the raga.

The *vadi* is used frequently to start and end particular phrases. It is also important in determining the mood of a raga, by dwelling on it for a long time, elaborating its importance and its relation to the tonic. The *vadi* also plays a role in determining the time of day when a raga is performed by its position in the scale. In performance, the rhythmic accompaniment also helps stress the importance of the *vadi* by using more elaborate rhythmic patterns.

The *samvadi* serves to highlight the importance of the *vadi*. It is always found in the other tetra chord, usually in the same corresponding position within the four note set of the tetra chord [7].

A.3.3 Varna

There are five types of melodic movements (*varna)* in Indian classical music. *Sthayi* describes stability and permanence, where notes are continuously held, with small phrases in between starting and ending on the same note. *Arohi* has been described above as ascending melodic movement, and *Avarohi* describes descending melodic movement. *Sanchari* is the term used for meandering between ascending and descending motifs. *Vidari* describes melodies that are discontinuous. The distinctions in these five classes of varna are portrayed in a performer's use of *tans* [7].

A.3.4 Tan

A tan is a group of notes used to expose and expand a raga. The word comes from the root *tanama*, which means to stretch. *Tans* are generally performed at a fast tempo, at least twice the speed of the tempo of the melody. *Dugani* refers to when a *tan* is performed with two notes per beat (also known as eighth notes). *Tiguni* refers to three notes per beat (triplets) and *chauguni* refers to four notes per beat (sixteenth notes).

There are three basic types of *tans.* A *sapat tan* is straightforward, playing a series of notes in order such as *Ni, Dha, Pa, Ma, Ga, Re, Sa*. A *vakra tan* is a twisted melody that goes up and then down in one phrase like a rollercoaster. *Alankar* refers to pre-rehearsed patterns which generally incorporate the use of scale exercises. Some of these patterns are particular to a family and style of playing.

A.3.5 Ornamental Devices

A key aspect of Indian classical music is the way in which each individual note is ornamented using microtones and advanced performance techniques. This is the tonal grace of gliding from one note to the next and the subtle art of using frequencies just above and below the primary note.

There are a variety of ornamental devices. *Mindh* refers to technique of a slow glide connecting two notes, where both notes are equally important. *Kana* is a shadow or grace note of less intensity and duration before a primary note being stressed, similar to a glissando in Western music. *Murki* is a quivering trill around a main note incorporated frequencies just above and below the principal note. *Gamak* refers to heavy fast shaking of a note. *Kampan* is a fast oscillation between notes, resulting in a slight alteration in pitch. *Andolan* refers to a slow gentle oscillation between notes [7].

A.3.6 Thaat system

The *Thaat* system was introduced by Pandit Vishnu Narayan Bhatkhande (1860-1936) to classify ragas into parent-scale modes. *Thaat-s* are basic patterns of seven-note arrangements creating a classification scheme to group several ragas under one “mode type”. In every *thaat* there is one raga that bears the name of the *thaat*, while others are derived by dropping one or more notes from the parent-scale.

The *melakarta* raga system in South Indian Carnatic music theory derives the combinations of twelve notes into seventy-two *mela-s* or arrangements. This is a fool-proof framework to classify all the raga-s that can possibly exist. In contrast, Pandit Bhatkande’s *thaat* system, only classifies prevalent raga-s into ten subcategories. Figure 104 portrays the ten *thaat-s*, their corresponding notes, and their equivalent relation to the Western modal system.

Thaat	**Notes**	**Western Mode**
Bilawal	*Sa, Re, Ga, Ma, Pa, Dha, Ni, S^{o}a*	*Ionian*
Kalyan	*Sa, Re, Ga, Må, Pa, Dha, Ni, S^{o}a*	*Lydian*
Khamaj	*Sa, Re, Ga, Ma, Pa, Dha, Ni, S^{o}a*	*Mixolydian*
Bhairav	*Sa, Re, Ga, Ma, Pa, Dha, Ni, S^{o}a*	
Purvi	*Sa, Re, Ga, Må, Pa, Dha, Ni, S^{o}a*	
Marwa	*Sa, Re, Ga, Må, Pa, Dha, Ni, S^{o}a*	
Bhairavi	*Sa, Re, Ga, Ma, Pa, Dha, Ni, S^{o}a*	*Phrygian*
Asavari	*Sa, Re, Ga, Ma, Pa, Dha, Ni, S^{o}a*	*Aeolian*
Kafi	*Sa, Re, Ga, Ma, Pa, Dha, Ni, S^{o}a*	*Dorian*
Todi	*Sa, Re, Ga, Må, Pa, Dha, Ni, S^{o}a*	

Figure 104 - Thaat system. with appropriate scales.

There are many shortcomings of this system. Firstly, some ragas are borderline and can be classified as two different *thaat-s*, as there are not enough *thaat-s* to classify all raga-s. Also, *thaat-s* do not preserve information on how notes are stressed, ornamentation, and key elements which make raga-s unique. Thus, this system is not widely accepted in India, and is only described to show some connection to Western modal music [7].

A.3.7 Raga and Emotion

As mentioned above, each raga depicts a certain mood and emotion when performed. One exotic tale describes the performance of *Raga Dipak* lighting lamps on fire, as well as the performer themselves! *"Gopal Naik, commanded by the Emperor Akbhar, … sang Raga Dipak but was saved from death by burning, as his wife, realizing the danger, immediately began singing Raga Malhar in order to bring down the rain [7]"*.

The *rasa* theory arose within the context of drama. It refers to the aesthetic experience and emotional response of the audience during a performance. There are nine *rasa* states in which to categorize all *raga-s: shringara* (romantic/erotic),

haysa (comic), *karuna* (pathetic), *raudra* (wrathful), *vira* (heroic), *Bhayanaka* (terrifying), *bibhatsa* (odious), *adbhuta* (wonderous), *shanta* (peaceful, calm). [7]

A.3.8 Ragas and Time

One aspect of North Indian raga music that distinguishes it from Carnatic music is that traditionally each *raga* is associated with a time of day or time of year. The *raga-s* linked with a season are described in Figure 105.

Raga	**Season**
Hindol	Spring
Malhar	Rainy
Dipak	Summer
Megh	Monsoons
Bhairav	Autumn
Malkauns	Winter

Figure 105 - Chart of Raga-s corresponding to Season (Time of year).

Raga-s are also linked with the time of the day. These are based on a *raga-s* connection with mood, the notes present, the tetra chord to which the *vadi* belongs, and the note hierarchy. For example, *tivra Ma* is used at sunset or at night; thus, as the sun sets, *Ma* turns to *Må*. It is also common that *raga-s* performed at dawn contain a *Re*, and occasionally *Dha*. Figure 106 shows a chart relating *raga-s* and *thaat-s* to time of day. Many of the raga-s listed in this chart are not found in this dissertation and the reader is pointed to an online source for easy access to more information about each raga[43] [7].

[43] Available at: http://www.itcsra.org/sra_others_samay_index.html (January 2007)

Time of Day	Thaat	Raga
6-9 a.m. *1st Quarter of Day*	Bilawal	Alahya Bilawal, Shuddh Bilawal, Devgiri Bilawal, Shukla Bilawal
	Bhairav	Bhairav, Ahir-Bhairav,Ramkali, Jogiya, Bhairav-Bahar, Gunakari, Vibhas
	Bhairavi	Bhairavi, Bilakhani Todi, Bhupali Todi
	Kalyan	Hindol
9 a.m. - noon *2nd Quarter of Day*	Todi	Miya-ki-Todi, Gurjari Todi
	Asawari	Asawari, Komal Re Asawari, Jaunpuri, Deshi, Sindh Bhairavi
	Kafi	Sughrai, Sur Malhar
	Bilawal	Deshkar
Noon - 3 p.m. *3rd Quarter of Day*	Kafi	Brindabani-Sarang, Shuddha Sarang, Bhimpalasi, Dhanashri, Pilu, Suha
	Kalyan	Gaud-Sarang
3-6 p.m. *4th Quarter of Day*	Purvi	Purvi, Purya-Dhanashri, Shri, Triveni
	Marwa	Marwa, Purya
	Todi	Multani
	Kafi	Pat-Manjari
6-9 p.m. *1st Quarter of Night*	Kalyan	Yamen, Bhupali, Shuddha Kalyan, Hamir, Kedar, Kamod, Chhaya-Nat, Malashri
	Bilawal	Hansadhwani
9 p.m. - midnight *2nd Quarter of Night*	Bilawal	Shankara, Durga, Nand, Bihag
	Khamaj	Khamaj, Jaijaiwanti, Desh, Ragashwari, Tilak Kamod, Jhinjhoti, Kalawati, Bhinna Shadja, Gara, Tilang
	Kafi	Kafi, Bageshwari, Malhar, Miya Malhar, Gaud Malhar
Midnight- 3 p.m. *3rd Quarter of Night*	Kafi	Bahar, Kanada, Nayaki Kanada, Kaunsi Kanada
	Asawari	Darbari Kanada, Adana, Shahana Kanada
	Bhairavi	Malkauns
3 - 6 p.m. *4th Quarter of Night*	Purvi	Basant, Paraj
	Marwa	Sohoni, Lalit, Bhatiyar, Bhankar
	Bhairav	Kalingda

Figure 106 - Chart describing correspondence between Raga-s, Thaat-s and time of day [7].

A.4 Theka

Rhythm lays the framework for all music, arranging sound events over time. The traditional drum of North Indians music that provides rhythm is known as the *tabla*. Musical enhancement is the major role of the *tabla* in Hindustani music. *Theka*, which literally means “support”, is the Indian word for simple accompaniment performed by a *tabla* player. The importance of the *theka* underscores the role of the *tabla* player as timekeeper. An even more specific

definition of *theka* is the conventionally accepted pattern of *bols* which define a *tal*. The word *tal* literally means *clap*, for the clapping of hands is one of the oldest forms of rhythmic accompaniment [39].

The most fundamental unit of this rhythmic system is the *matra*, which translates to “beat”. In many cases the *matra* is just a single stroke. Just as sixteenth, or eighth notes maybe strung together to make a single beat, so too may several strokes of *tabla* be strung together to have the value of one *matra*. The next higher level of structure is *vibhag*, which translates to “measure” or “bar”. These measures may be as short as one beat or longer than five. Usually, however, there are two, three, or four *matras* in length. These *vibhags* are described in waves or claps. A *vibhag* which is signified by a clap of the hands is said to be *bhari* or *tali*. Conversely, a *vibhag* which is signified by a wave of the hand is said to be *khali* [39]. Whereas in Western classical music there are usually an equal number of beats per measure (3-3-3) or (4-4-4), it is common in Indian Classical music to have a different number of *matras* per *vibhag* such as (2-3-2-3) or even (5-2-3-4) [7].

The most common *theka* is known as *Tin Taal* (which translates to “three claps”), where there are 16 *matras*, divided into four *vibhags*. Its arrangement is:

Clap, 2, 3, 4,

Clap, 2, 3, 4,

Wave, 2, 3, 4,

Clap, 2, 3, 4,

The first line is the ***bhari*** *vibhag*, the third line is a *khali vibhag*, where as the other two lines are *tali* vibhags. In performance, the cycle of sixteen beats is repeated over and over. This cycle, known as *avartan*, refers to the highest level of conceptual rhythmic structure. The repetition of the cycle gives special significance to the first beat. This beat, known as *sam*, is a point of convergence between the Tabla player and the melodic soloist. Whenever a cadence is indicated it usually ends on the *sam*, with the soloist landing on the *vadi* or

samvadi. This means that the *sam* may be thought of as both the beginning of some structures as well as the ending of others [39]. The *khali* plays an important role, warning the soloist of the approaching *sam.*

The mnemonic syllables, known as *bol*, represent the various strokes of the Tabla, which are described in Chapter 5. Figure 107 represents a chart of most common *theka-s*.

Theka	**Matras**	**Vibhag Division**	**Bols**
Dadra	6	3-3	**Dha** Dhin Na | Dha Tin Na
Rupak	7	3-2-2	Tin Tin Na | Dhin Na | Dhin Na
Kaharwa	8	2-2-2-2	**Dha** Ge | Na Ti | Na Ke | Dhin Na
Jhap-tal	10	2-3-2-3	**Dhin** Na | Dhin Dhin Na | Tin Na | Dhin Dhin Na
Dipchandi tal	14	3-4-3-4	**Dha** Ge - | Dha Ge Tin - | Na Ke - | Dha Ga Dhin -
Tintal	16	4-4-4-4	**Dha** Dhin Dhin Dha | Dha Dhin Dhin Dha | Dha Tin Tin Na | Na Dhin Dhin Dha

Figure 107 - Common *Thekas* with *Bol* patterns [7].

A.4.1 Laya

Laya refers to the tempo and pulsation of music. A performance begins with *alap*, where the soloist introduces each note in the raga one at a time. The melodic rules of the raga are revealed, one by one, slowly rising from the lower octaves to the upper octaves. There is no accompaniment by a percussion instrument in the *alap*. However, meter is certainly present at a very slow tempo.

There are three main tempos in Indian classical music: *Vilambit* (slow; 30-60 BPM), *Madhya* (medium; 60-120 BPM), and *Drut* (fast; 120-140 BPM). The *alap* is followed by a *vilambit ghat* (composition), accompanied by tabla, with improvisations. The music increases in intensity moving to *Madhya laya* with a

new *ghat*, and finally *drut laya*. *Ghat-s* at all *laya* are intertwined with *tan-s* presenting the soloist's skill and improvisation ability.

A *tihai*, a pattern repeated three times ending on the *sam*, helps transitions between sections, end improvisations, and prepares cadences. A *chakradhar* is similar to *tihai-s*, but generally is longer in length, is more climactic, and is used in the *drut laya*, ending on the *sam*.

Appendix

B

B
Physical Computing
Using Sensors & Microcontrollers

This chapter gives an overview and brief explanation of the technology used to build systems for capturing non-trivial musical data from the a performing artist. It is a subset of a larger field known as Human-Computer Interfaces (HCI) or more recently referred to as Physical Computing. This chapter will first define a microcontroller and its basic functionality. Next, a variety of sensors will be described which were used in this research. Next, there is a section describing a variety of actuators including motors and solenoids, with basic circuit diagrams. The chapter ends with a description of common music protocols used to communicate information from the microcontrollers to the laptop.

B.1 Microcontrollers

Microcontrollers are small, low-cost computers, designed to accomplish simple tasks on programs loaded in Read Only Memory (ROM). Microcontrollers are the electronic heart of the music controllers and robotic systems designed for this research. *"They act as gateways between the physical world and the computing world"* [166].

Microcontrollers are very inexpensive, starting as low as $5 US, and are thus ubiquitous in their use. One can find microcontrollers in many common

devices such as cellular phones, digital cameras, mp3 players, washing machines and even advanced light switches. Microcontrollers have three main functions: receiving data from sensors, controlling simple actuators, and sending information to other devices. In this research, three types of microcontrollers were used: PIC Microchip, Basic Stamp and Atmel.

B.1.1 PIC Microchip

A large group of devices in this dissertation use a PIC Microchip[44]. Specifically a PIC 18f2320 is used. This model has many built in functions such as a variety of basic digital inputs and outputs, ten analog-to-digital converters, two pulse width modulation pins, and USART serial communication pins.

Digital inputs are used to trigger events based on a high input or low input. The most common use of this function is for buttons and switches. Analog-to-digital converters translate voltage readings from sensors to bits that microcontrollers can use to deduce continuous physical information.

To control motors and solenoids, pulse width modulation (PWM) is used. PWM creates a varying output voltage. To understand how PWM works, a good analogy is turning a light switch on and off rapidly and evenly, which is equivalent to keeping a light constantly on at 50% of its full power. Similarly, a series of pulses is sent to the pin, and the average voltage is the resulting pseudo-analog voltage.

The PIC 18f2320 is run at 40 megahertz using a crystal and appropriate capacitors to set the cycle time. The microcontroller runs the assembly code written into ROM which is flashed in using a microchip burner. For most projects described, all code is written in C which is compiled into assembly code before being burned onto the chip.

The PIC 18f2320 can be run at variety of low current voltages. For the projects described in this research, because the MIDI protocol requires five volts

[44] Available at: http://www.microchip.com (November 2006)

to one of the pins, the PIC is run on five volts using a 7805 voltage regulator. The Pin Diagram is shown in Figure 108.

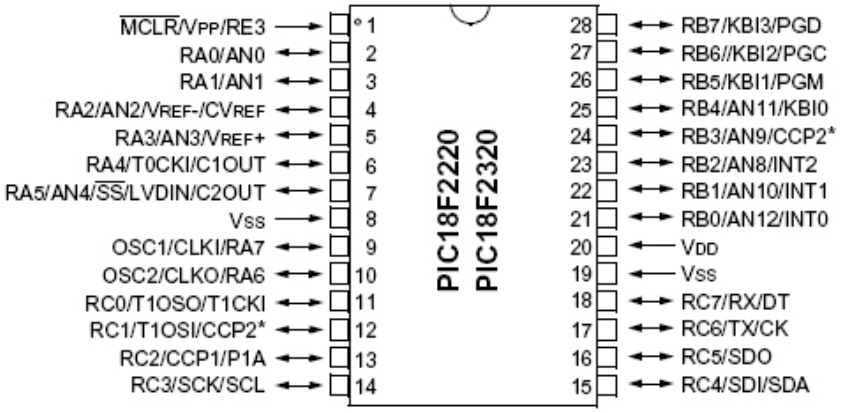


Figure 108 - PIC 18f2320 Pin Diagram.

B.1.2 Basic Stamp

Some devices in this research use a Basic Stamp microcontroller. The Basic Stamp is a programmable micro controller, developed by Parallax[45], Inc. There is a large variety of BASIC Stamps however devices in this research were first developed using the BASIC Stamp II (shown on left of Figure 109) and then upgraded to the BASIC Stamp IIsx (shown on right of Figure 109), which has a faster processing speed.

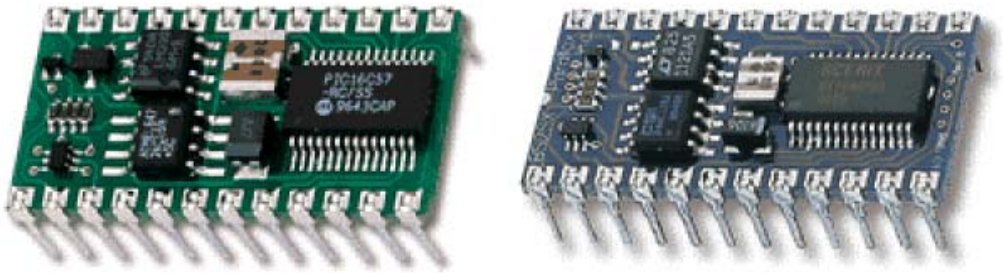


Figure 109 - Basic Stamp II and Basic Stamp IIsx.

[45] Available at: http://www.parallax.com (November 2006)

The BASIC Stamp is programmed by Windows software provided by Parallax. The programming language is PBASIC (Parallax BASIC) which is based on the BASIC programming language. Code is transferred from the computer to the powered BASIC Stamp via a serial port on the carrier board. The code is stored in the EEPROM memory after being tokenized. Programming elements, such as constants, comments, and variable names, are not stored in the BASIC Stamp, so descriptive names and comments are included in PBASIC code for devices. The BASIC Stamp II only has room for about 500 lines of code, executed at 4000 instructions per second, whereas the BASIC Stamp IIsx has room for 4000 lines of code, executed at 10,000 instructions per second. Thus the BASIC Stamp IIsx executes 2.5 times as fast for time sensitive commands. The author points readers to [166] and [181] for more details on implementation and projects using the Basic Stamp.

B.1.3 Atmel

The Atmel[46] series of microcontrollers were also used during experimentation in this research. Specifically, devices in the research used an Atmel AVR ATMega16 microcontroller. The low-cost 8-bit microcontroller has eight built-in 10-bit analog to digital converters as well as twenty-four general purpose digital I/O pins. The microcontroller is housed on an AVRmini development board to provide clocking, serial ports, connector access to the chip, power regulation and programming circuits. The microcontroller program, which reads ADC inputs and transmits MIDI messages, is written in C with the help of the Procyon AVRlib support library available at CCRMA. The author points readers to [196] for more details on implementation and projects using the Atmel for music applications.

B.2 Sensors

Sensors are a type of transducer which measure physical data from the world for machine perception. In my research, they serve as the machines', eyes, ears, and

[46] Available at: http://www.atmel.com (November 2006)

feeling receptors. This section describes common sensors used to obtain pressure, rotation, and position for music applications. Force sensing resistors, piezoelectric sensors and accelerometers are described.

B.2.1 Force Sensing Resistors

Force sensing resistors (FSRs) convert mechanical force into electrical resistance. FSRs used in this research are manufactured by Interlink Electronics[47] and can be purchased at their online store. These sensors use the electrical property of resistance to measure the force (or pressure) exerted by a user. Essentially, they are force to resistance transducers: the more pressure exerted, the lower the resistance drops. The particular FSRs are made of two main parts: a resistive material applied to a piece of film, and a set of digitizing contacts applied to another film. The resistive material creates an electrical path between a set of two conductors. When force is applied, conductivity increases as the connection between the conductors is improved.

B.2.2 Piezoelectric Sensors

Piezoelectric (Piezo) sensors take advantage of the piezoelectric effect in which mechanical energy is converted to electrical energy. Electrical charge results from the deformation of polarized crystals when pressure is applied. These sensors respond very quickly and are thus a great choice of drum interfaces which need a very quick response time. Piezo sensors detect very small force changes and produce a varying voltage when bent. Thus they also can be used to make microphones and capture audio signals.

B.2.3 Accelerometers

Accelerometers are sensors that measure acceleration using an electrical mass spring system. Accelerometers can easily be used to deduce tilt or rotation in three axes by wiring three components together.

---

[47] Available at: http://www.interlinkelectronics.com (November 2006)

B.3 Actuators

B.3.1 Motors

Motors are devices to create motion controlled by the microcontroller. The motors discussed in this section create rotary motion, which can be translated into linear motion with appropriate system design. There are four basic types of motors: DC, gear head, RC servo, and stepper. The *BayanBot* (discussed in Chapter 9) uses a stepper motor [166], so it is the only motor that is described in this section.

An important fact to remember is that the motors are powered by a separate power source than that of the microcontroller. Each motor is rated with the maximum amount of voltage which can be supplied before it dies. In general, the speed of the motors can be controlled with PWM as discussed above. Stepper motors use four digital output pins to control rotation.

Servo motors are different from other motors in that they don't turn continuously once set into motion. Rather, they move in a series of precise steps, as can be inferred from the name. The center shaft of the motor has several magnets mounted, while surrounding coils are alternately given current, creating magnetic fields which repulse or attract the magnets on the shaft, causing the motor to rotate.

The advantage of stepper motors is the precise control of position, while the tradeoff is the slowness in action. However, stepper motors can provide high torque at low speeds which make them appropriate for the *BayanBot*.

In order to switch directions so the motor can turn clockwise and counterclockwise, an H-Bridge is used. The circuit diagram is given in Figure 110.

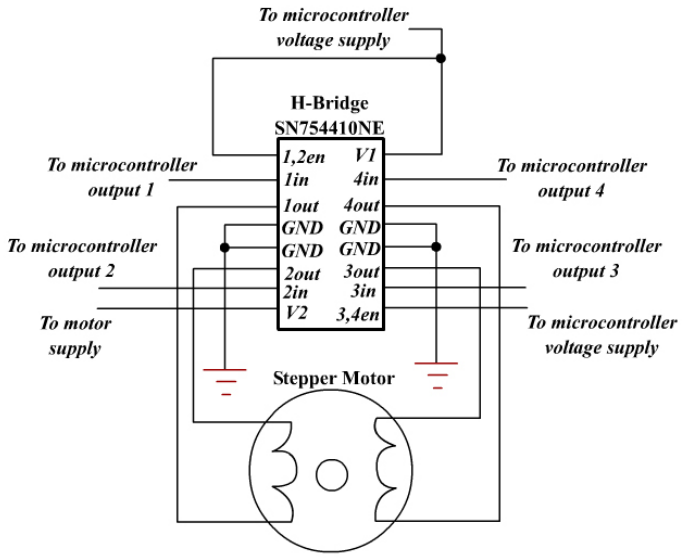


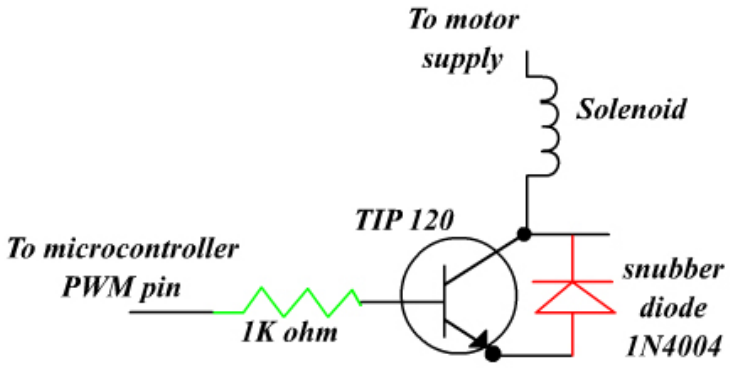
Figure 110 - Stepper Motor Circuit Diagram.

B.3.2 Solenoids

A solenoid [166] is a special type of motor which creates linear motion. It consists of a coil of wire with an iron shaft in the center. When current is supplied to the coil, a magnetic field is created and the shaft is displaced. When the current is removed, the magnetic field is no longer present and the shaft returns to its original position. The time period between supplying current and turning it off must be short or the solenoid will overheat and stop working. The greater the initial voltage supplied to the solenoid, the greater the immediate displacement, resulting in a harder striking motion. Thus it can be used in conjunction with Pulse Width Modulation (PWM) to supply variable control of striking power. There are two types of solenoids; ones that can pull and ones that push.

The circuit diagram for operating a solenoid is shown in Figure 111. When a coil of wire is moving in a magnetic field, it induces a current in the wire. Thus when the motor is spinning near a magnet and then is turned off, the magnetic field induces a current in the wire for a brief time. This back voltage can damage electronics, especially the microcontroller. To avoid this, a snubber diode is used to block the current from going the wrong way. As well, a transistor is used to

switch the higher voltage power of the motor to the low voltage power of the microcontroller.



.

Figure 111 - Circuit Diagram for using a Solenoid

B.4 Music Protocols

B.4.1 MIDI

MIDI is short for Musical Instrument Digital Interface. It is a communication protocol, which allows electronic instruments (such as keyboards, synthesizers, and musical robots) to connect and interact with each other. Starting in 1983, MIDI was developed in cooperation with the major electronic instrument companies such as Roland, Yamaha, and Korg. The companies created a standard interface, to solve inter-instrument communication problems, and thereby generating more sales. Since than, the protocol has evolved to fit the needs of professional musicians, as larger amounts of controllers and sounds were created.[48]

[48] http://www.midi.com

MIDI is transmitted at 31,250 bits per second. Each message has one start bit, eight data bits, and one end bit, which means the maximum transmission rate would be 3215 bytes per second. When the first bit is set to 1, the byte is a status byte. Status byte denotes MIDI commands such as NOTE ON, NOTE OFF, and CONTROL CHANGE, and communicates which channel (0-15) to send information. The status byte also determines the length of the message; messages are generally one, two, or three bytes in length. An example of a common message is illustrated below:

10010000	00111100	01000000
Note On	Channel 0 Note #60	Velocity = 64

The NOTE ON command will trigger a MIDI device to turn on a sound. The pitch byte will tell the device to play Note 60, which is middle C on a piano sound bank. The velocity byte will tell the device how loudly to play the note [32]. On a standard MIDI device there are three five-pin ports (IN, OUT, THRU) that transmit and receive MIDI information. The IN port receives and processes MIDI commands, while the OUT transmits it. The THRU port receives and processes MIDI information and transmits the same message through the OUT port. Musical Robots receives data through a MIDI IN port, triggering solenoids to fire at certain speeds. The MIDI OUT port can be used to send feedback to the microchip about timing, force and position of strikes from robots or humans.

The circuits for MIDI OUT (Figure 112) and MIDI IN (Figure 113) are shown below.

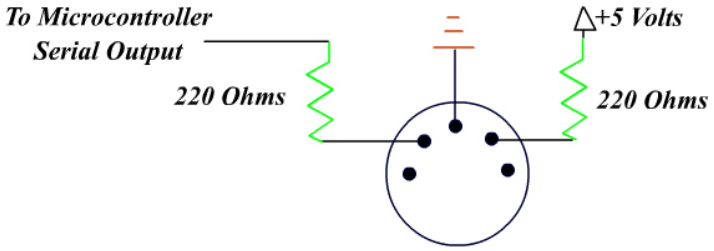


Figure 112 - MIDI Out Circuit Diagram.

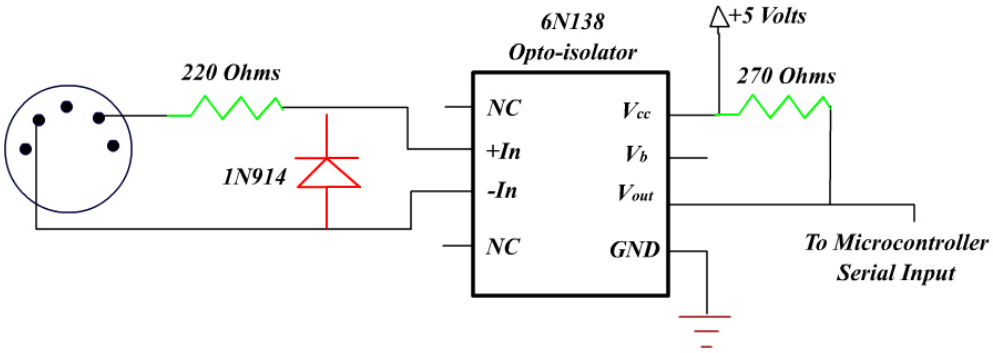


Figure 113 - MIDI In Circuit Diagram.

B.4.2 OpenSound Control

A newer music protocol was designed at University of California Berkeley which is based on a client/server architecture [199]. Messages are created using a URL format allowing for more descriptive names than the hexadecimal names in MIDI. An example message could be "/*ESitar*/thumb/120" or "/*ETabla*/Dha/56". Data is transmitted using UDP to the correct IP address using Ethernet cables or even wireless transmission.

Appendix

C

C Machine Learning

Decision Trees, Neural Networks, Support Vector Machines

Machine learning refers to computer programming which results in a machine learning a specialized system with experience. There are endless examples of applications of this, including automatic automobile transportation, machine-based emotion recognition, and even medical assistive technology which enable computers to further aid in the hospital. In the audio field, applications range from automatic instrument type recognition [70], to automatic beat detection [3] and transcription [16], to voice recognition [201], to genre classification [178], to audio-based gesture recognition [86].

In this chapter, a variety of machine learning algorithms will be presented. Machine learning heuristics are commonly known as classifiers which map unlabelled data to discrete classes. The classifiers that will be discussed include: ZeroR Classifier, k-Nearest Neighbor, Decision Trees, and Artificial Neural Networks.

In explaining the details of the different algorithms, this paper will refer to *feature data* as the input and *classes* as the output to the classifiers. *Feature Extraction* is the process of reducing massive amounts of data (from an audio source in these examples) to perceptive, compact numerical representation for the

classifiers to use [114]. *Training* refers to the process in which the machine takes a labeled set of data (with feature data mapped to correct class) and trains the classifier. *Prediction* refers to the process in which the machine takes feature data and tries to predict the correct class based on training.

C.1 ZeroR Classifier

ZeroR is the simplest of classifiers. It is generally used to provide ground truth for a data set, as it serves as a worst case scenario. All other algorithms should perform better than ZeroR.

The algorithm counts the number of instances for each class during training. When training is complete, *(i.e. mode == done)* the classifier simply predicts for every instance it encounters the class with highest number of instances [72].

C.2 k-Nearest Neighbor

k-Nearest Neighbor (kNN) is a much more complicated algorithm than the ZeroR classifier. During training, the machine stores the N-dimensional feature set along with the corresponding class.

During prediction, the machine takes the test data and calculates the distance in the N-Dimensional feature space to every point in the training set. These distances are stored in an array *Distance()* in *Marsyas*, along with their corresponding classes taken from the training points. Next, the algorithm must find the *k-smallest* values in the *Distance()* array. In finding the minimum values, it is important to only go through the *Distance()* array once (and not *k* times) for speed in real-time applications. Thus another array *kMin()* will keep track of the k-smallest distances at any given time from its one iteration through *Distance()*. *kMin()* is initialized with the first k-values of *Distance()*, and a variable *kMax* points to the maximum value of *kMin*. Then the algorithm starts stepping through the entire *Distance()* array, comparing each value with *kMax*. If *kMax* is greater

than any value in *Distance()*, the value is placed in *kMin* and a new *kMax* is calculated and the pointer adjusted. This process continues to the end of the *Distance()* array. Finally, the *kMin* array stores the k-smallest values, with their appropriate class values. To make a prediction, the algorithm picks the class with the most occurrences in the *kMin* array. [51]

A pictorial view of a kNN classifier is shown in Figure 114. In this example, the feature space is two dimensional and can be represented in an x-, y-coordinate space. As can be seen, the prediction point would be classified as *class 1* based on the proximity to its "nearest neighbor".

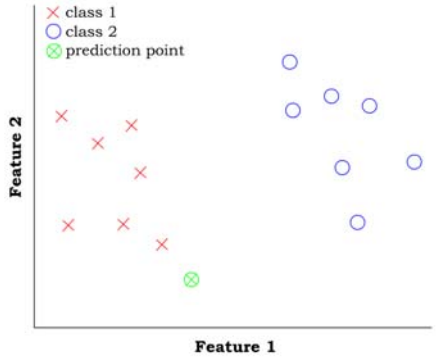


Figure 114 - Illustration of kNN Classifier showing class 1 and class 2 points with 2 features and a prediction point which would be classified as class 1 based on the proximity.

C.3 Decision Trees

A Decision Tree is a machine concept learning algorithm which uses a tree data structure. Decision Trees are appropriate for problems that have a fixed set of values for their attributes (*i.e. Temperature - Hot, Cold, Warm*) but the Trees can be extended to handle real values (*i.e. Temperature - 57,35,53,79*). Also, the algorithm is robust enough to handle training data with errors or missing attribute values.

C.3.1 An Example Problem

In order to explain the details of how a decision tree works an example problem will be presented. Suppose it is desired to train a machine to determine whether an audio file is traditional Indian Classical music or Western music, using the attributes of drum, string, and wind instrument type. If the following training data were collected, then the corresponding decision tree would be created as seen in Figure 115:

Tabla, Sitar, Bansuri - Indian Classical Music

Snare Drum, Sitar, Saxophone = Western Music

Dholak, Guitar, Saxophone - Western Music

Dholak, Sitar, Bansuri - Indian Classical Music

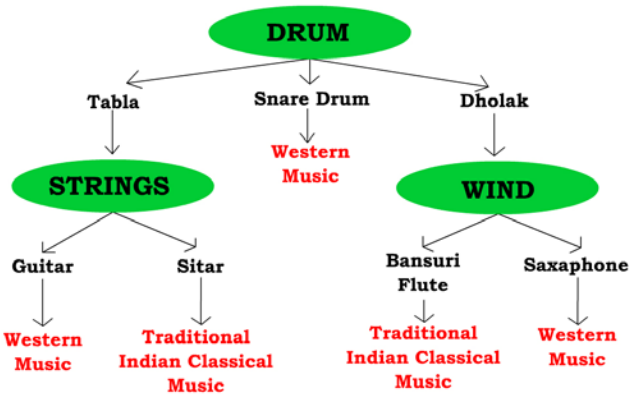


Figure 115 - Illustration of Decision Tree for example problem of classifying Traditional Indian Classical Music and Western Music using attributes of drums, strings, wind instrument type.

As seen by Figure 115, the data is sorted from root to leaf node. Each node specifies an *attribute* of the instance. Each branch corresponds to the *value* of the attribute, and the leaf node determines the *class* of the instance. An advantage of Decision Trees over all other classifiers is that a tree structure makes the classification scheme readable by the human eye, as seen in Figure 115.

C.3.2 ID3 Algorithm

The ID3 algorithm constructs the tree by determining which attribute should be tested at the root of the tree. This is done by a statistical test for each attribute which finds the best one for the root node. All descendents of the root are chosen in a similar fashion, and thus this algorithm is a *greedy* algorithm, which does not backtrack to earlier decisions. This process is accomplished by calculating *Information Gain* which is quantitative measure of the worth of an attribute.

It is important to understand *Entropy* before Information Gain can be explained properly. Entropy characterizes the (im)purity of an attribute. Given a collection S, with positive(+) and negative (-) examples, Entropy(S) = $-p_+ \_lg(p_+)-p_- \_lg(p_-)$ where p_+ are the proportion of positive examples and p_- is the proportion of negative examples. Notice that if $p_+ = 1$ then $p_-=0$ and Entropy(S)=0. On the other extreme, if $p_+=p_-$ then Entropy = 1. This concept can be extended to targets that can take *c* different values by the following equation: Entropy(S) = $P_{c\ i=1}\ -p_i lg(p_i)$.

Information Gain is described in terms of entropy by:

$$Gain(S,A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

where *Gain(S,A)* is the information gain, *Values(A)* is the set of all values for attribute *A*, and S_v is the subset of values that have value *v*. In a way, information gain can be seen as the expected reduction in entropy caused by partitioning by a certain attribute. The greedy algorithm selects the attribute with the highest information gain, (so in the example problem, the Drums attribute had the highest information gain) removes it from the set of possible attributes (as each attribute can be used only once as a node) and then moves to the left node and repeats the process.

A limitation of the ID3 algorithm is that it never selects an attribute for a node and then never backtracks to reconsider its choice. This makes the heuristic susceptible to converging to a locally (not global) optimal solution. However an advantage of the ID3 algorithm is that all the training examples are used for a

statistically-based hypothesis (not incremental on individual training examples) which makes it robust to errors in individual or ”noisy” training data. [114]

C.4 Artificial Neural Networks

A neural network is one of many machine learning techniques used as a classification framework. It is modeled after the biological learning process of the human brain. In this section the way the human brain’s learning process functions will be discussed, followed by how an artificial neural network is set up and how it learns based on the brain model.

C.4.1 The Human Brain

The human brain is composed of a vast network of interconnecting structures known as neurons. The interconnections between these myriad of neurons are synaptic tissue that ever-change to learn how to solve a specific problem. Learning occurs by example, as the synaptic interconnections adjust themselves.

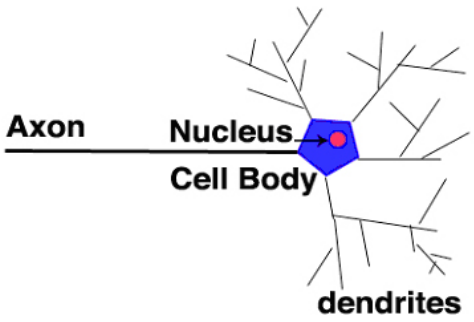


Figure 116 - Illustration of neuron in brain: nucleus, cell body, dendrites and axon.

Figure 116 shows the components of a neuron. At the center of this structure within the cell body is a nucleus that collects signals from its surrounding dendrites. Signals are sent from the neuron through a structure known as an axon, which splits into many branches. The synapse, as shown in Figure 117, converts the energy from the axon to signals that an interconnecting neuron can

process. A neuron sends a spike of electrical activity down its axon, when it receives input that is larger than its inhibitory input. Learning occurs by setting the influence of the synapse to effect changes of one neuron on another.

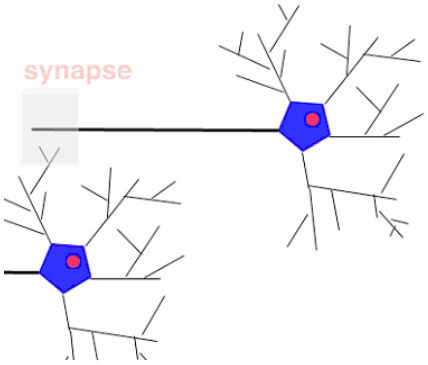


Figure 117 - Illustration of synapse converting signals from an axon to a dendrite.

C.4.2 An Artificial Neuron

The modeling of the human brain's neural network to an artificial neural network is grossly idealistic and is based on our limited knowledge of the fine details of the real networks of the brain. However, as seen in the experimentation of this report, the artificial neural networks prove to be very successful for certain applications.

Figure 118 shows a simple artificial neuron. The neuron has many inputs (dendrites) and an output (axon). The neuron has two modes: training and testing. In the training mode, a neuron can be trained to trigger (or not) based on the given input from the dendrites. In testing mode, when an exact taught input pattern is detected, its trained output becomes the output. However, if the input is not exactly the same as one of the training inputs then whether or not the neuron triggers is more complicated.

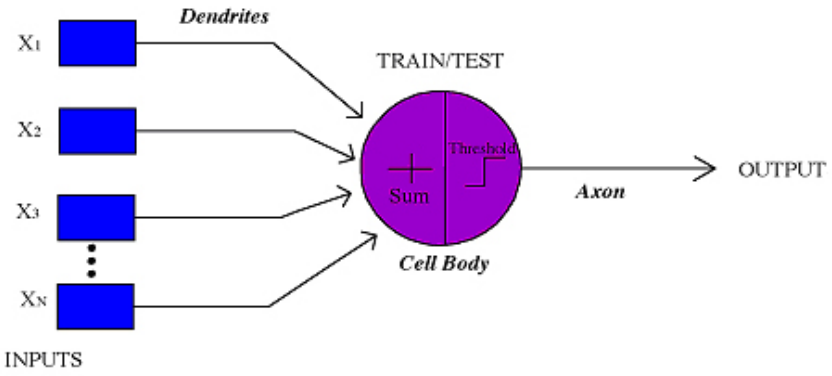


Figure 118 - Illustration of an artificial neuron.

There are several algorithms for determining whether a neuron should trigger based on a given test input. One technique is the Hamming distance, which is similar to a nearest neighbor algorithm. As an example, say a three input neuron is taught to output 1 when a, b, c, are 111 and 110 and outputs 0 when input is 000 and 010. As seen in Table 3, input 001 would output 0 because it has one term different from training input 000 and two and three different terms from 110 and 111. Input 011 remains ambiguous because it is one term different then 010 (trained to 0) and 111 (trained to 1).

a	0	0	0	0	1	1	1	1
b	0	0	1	1	0	0	1	1
c	0	1	0	1	0	1	0	1
out	**0**	**0**	**0**	**0/1**	**0/1**	**1**	**1**	**1**

Table 3 - Input (a, b, c) into artificial neural network with corresponding output using Hamming distance rule.

C.4.3 A Weighted Artificial Neuron

A more complicated neuron has weights on the inputs as seen in Figure 119. The weights W_n are multiplied by the value X_n and summed. If this value is greater than a certain threshold T then the neuron triggers.

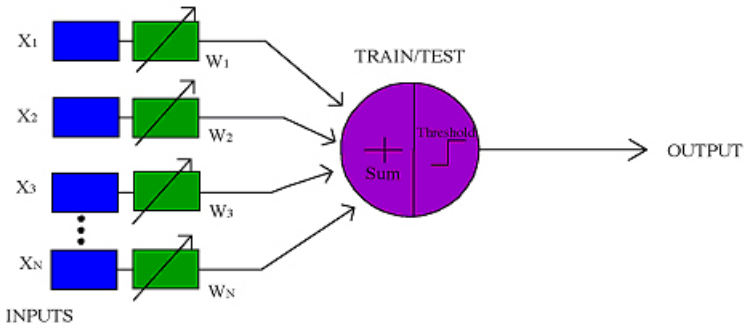


Figure 119 - Illustration of a weighted artificial neuron.

Thus the neuron has the ability to change its weights and threshold value in order to deal with a data set, making it much more powerful.

C.4.4 Artificial Neural Network Architecture

There are several types of architectures for neural networks. In this research project a feed forward network is used. This allows a signal to only move in one direction, from input to output, without any feedback loops. Figure 120 shows a diagram of layer of a feed forward neural network.

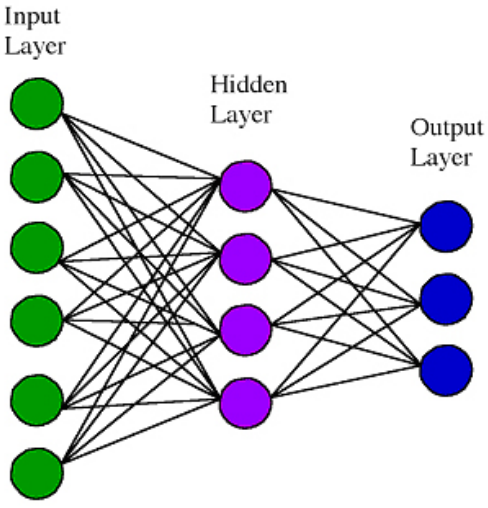


Figure 120 - Illustration of three layer architecture of an artificial neural network.

The neural network consists of three layers: input, hidden, and output. In audio analysis the input values are selected feature data. Thus the number of neurons in the input layer is precisely equal to the number of feature items. The hidden layer activity is determined by the weights and signals of the input layer. The output layer activity is determined by the weights and signals of the hidden layer. The number of neurons in the output layer is precisely equal to the number of classes. For example, if a drum made three sounds A, B, C, then the number of neurons in the output layer would be three. [2, 99, 114]

Appendix

D

D
Feature Extraction
Reducing Data to a Manageable Size

Feature extraction is an important step in machine learning to reduce data for the machine to a manageable size. These features are compact numerical representations of a signal, which can be easily used for classification experiments, regressive analysis, or machine-based comprehension of music being performed by the musician.

D.1 Audio-Based Feature Extraction

Perceptive features can be calculated for each musical sound. These parameters are extracted from both the time and frequency domains of a signal. These musical features correspond to different characteristics such as pitch, timbre, and inharmonicity. [208] In this section, the following features are extracted for each signal: ramp time, zero crossing, root mean square, spectral centroid, spectral rolloff, spectral flux, linear predictive coding coefficients, mel-frequency cepstral coefficients, and subband mean and variance in the wavelet domain.

D.1.1 Time and Frequency Domain

Vibrations between approximately 20 and 20,000 Hertz tha are received by the human ear can be perceived as sound, created by the oscillations of objects such as vocal chords, musical instruments, and speakers. These vibrations are

converted to the realm of digital audio by recording the sound using a microphone, which converts the varying air pressure into varying voltage. An analog-to-digital converter measures the voltage at regular intervals of time. For all recordings in this analysis, there are 44,100 samples per second. This is known as the sampling rate (SR). The data the computer stores after the analog-to-digital conversion is the sound as a function of time. [164] Figure 121 is a graph of a sound of a *Bayan* as a function of time.

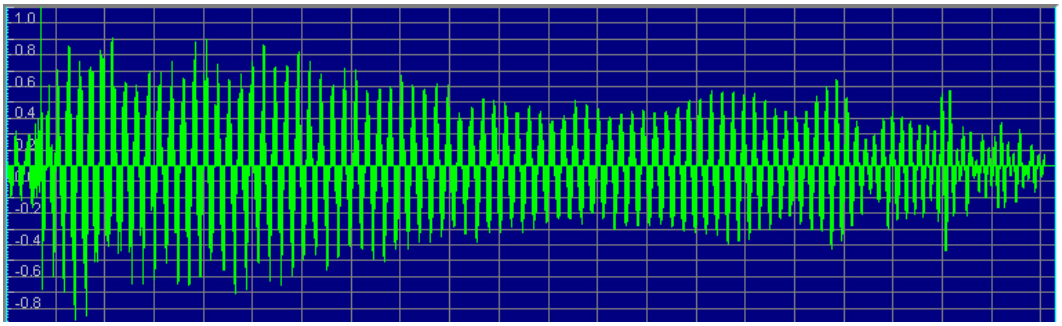


Figure 121 - Graph of sound of the *Bayan* as a function of time.

When one plucks a string or blows air through a tube, it begins a repeating pattern of movement, known as an oscillation. If a sound has a repeating pattern of movement it has a tone and a pitch (harmonic), which distinguishes it from noise (inharmonic). The tone and pitch of the sound can be determined by a sine wave with a particular frequency [164]. The cochlea, an organ in the inner ear enables humans to detect these frequencies. The cochlea is a spiral shaped organ of bone and tissue, with thousands of miniscule hairs that vary in size. The shorter hairs resonate with higher frequencies, while the longer hairs resonate with lower frequencies. So the cochlea converts the air pressure to frequency information, which the brain can use to classify sounds [29].

The Fourier Transform is a mathematical technique that converts sounds represented in the time domain to sound represented in the frequency domain [164].

Fourier analysis is based on the important mathematical theorem formulated by Joseph Fourier (1768-1830): *"Any periodic vibration, however complicated, can be built up from a series of simple vibrations, whose frequencies are harmonics of a fundamental frequency, by choosing the proper amplitudes and phases of these harmonics"* [144]. The Fourier Transform takes a periodic function of time F(t) and turns it into a summation of cosine and sine waves. A periodic function is transformed into the Fourier Series by the equation below:

$$F(t) = \sum_{m=1}^{\infty} \left(a_m \cos\left(2\pi k f_o t\right) + b_m \sin\left(2\pi k f_o t\right) \right)$$

The term a_m is the average waveform. Coefficients b_m and c_m are the weights of the cosine and sine terms, which describe different frequencies. [164]

With the Fast Fourier Transform (FFT), one can find the peaks of a sound, in the frequency domain. These peaks are known as modes.

D.1.2 Ramp Time

Ramp time is the number of samples from the beginning of the sound file to the first peak *R* as shown in Figure 122. This is calculated by rectifying the signal and then low pass filtering it (This is how one gets the envelope of a signal). Then the highest peak is found in the signal. For all the sound files in this research project, it is known that the maximum value of the envelope is the first peak, because drum hits have a very strong attack, which always contain more power then the rest of the sound.

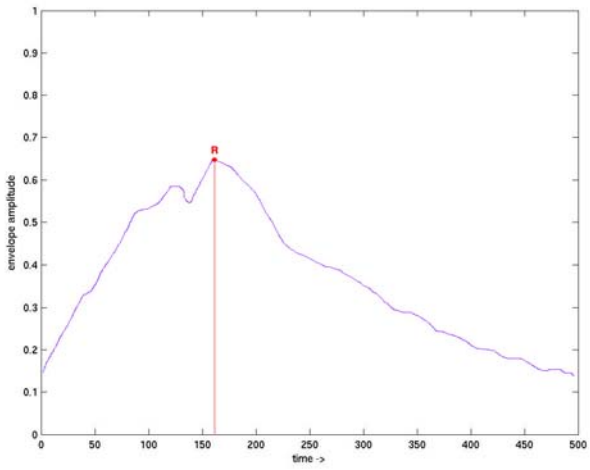


Figure 122 - Graph showing where ramptime finds maximum value of first peak and returns number of samples to point R.

D.1.3 Root Mean Square

A very important feature that is commonly used is root mean square (RMS). The RMS value gives an approximation of the variation of energy in the time domain. One RMS algorithm involves squaring the input signal to a filter and then calculating the square root of the output [208]. It is also common to sum the RMS values and then divide by the number of samples to get one average value. This is also known as the average power of a signal. Another way of calculating power is to use autocorrelation with zero lag.

D.1.4 Spectral Centroid

The spectral centroid is the center of gravity of a signal in the frequency domain. It gives an indication of how rich a sound is in harmonics. A simple way to calculate this value is shown in the following equation [178, 208]

$$centroid = \frac{\sum_{k=0}^{N/2} k * |X(k)|}{\sum_{k=0}^{N/2} |X(k)|}$$

An example of this is shown in Figure 123, which shows one frame of a FFT of a signal, with its corresponding centroid. Once again, averaging this value over time to give one value to the neural net is common practice.

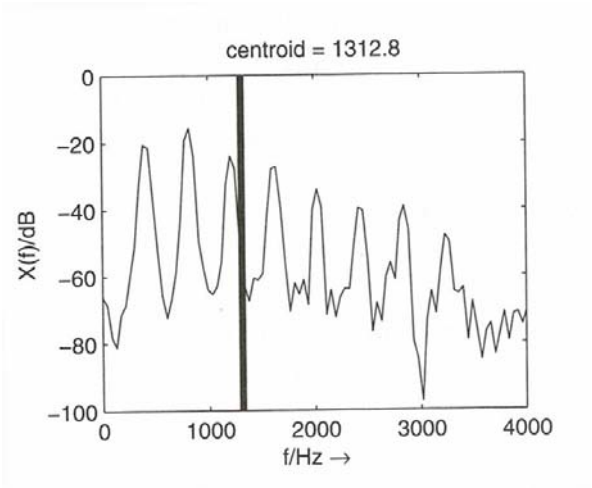


Figure 123 - Graph showing spectral centroid for one frame of a signal in the frequency domain. [208]

D.1.5 Spectral Rolloff

Spectral rolloff, like spectral centroid, is a feature that describes spectral shape. It is defined as the frequency R_t below which 85% of the magnitude of the spectrum is concentrated. If $M_t[n]$ is the magnitude of the spectrum then the spectral rolloff is given by:

$$\sum_{n=1}^{R_t}(M_t[n]) = .85 * \sum_{n=1}^{N}(M_t[n])$$

A value for spectral rolloff is calculated for each frame of the Short Time Fourier Transform (STFT), and then divided by the number of frames to give one feature value for the neural net. [178]

D.1.6 Spectral Flux

Spectral Flux measures the amount of local change over time in the frequency domain. It is defined by squaring the difference between normalized magnitudes in the frequency domain of frame *t* and *t-1*. If $N_t[n]$ and $N_t[n-1]$ are defined by the normalized magnitude of frame *t* and *t-1*, then the spectral flux F_t is given by:

$$F_t = \sum_{n=1}^{N} \left(N_t[n] - N_{t-1}[n] \right)^2$$

In should be noted that magnitudes are normalized by dividing each value in every frame by the RMS value of that frame. [178] F_t is calculated for each frame and then averaged over time to have one value for spectral flux.

D.1.7 Zero Crossing

The zero crossing feature is a simple technique which counts the number of times the signal crosses zero in the time domain. This can be useful to give a very rough estimation of pitch, or to give a characteristic of the attack part of a signal, by finding a number to represent the noise at the beginning of the signal. Figure 124 shows a simple signal in the time domain with a zero crossing value 8.

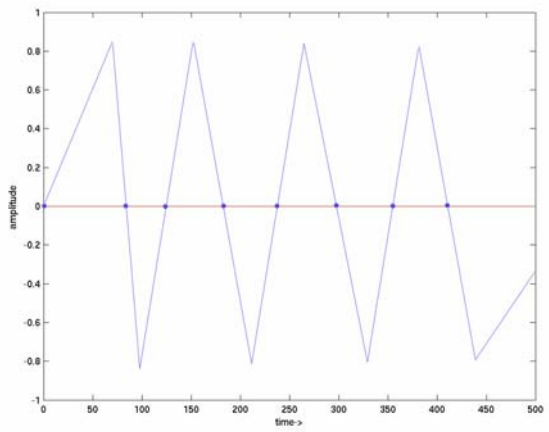


Figure 124 - Graph showing zero crossing feature finding eight points where time domain signal crosses zero.

D.1.8 Linear Predictive Coding

Linear predictive coding (LPC) produces an estimation $\bar{x}(n)$ for a sample value x(n) as a linear combination of previous sample values. This is shown by:

$$\bar{x}(n) = \sum_{k=1}^{L} a_k x(n-k)$$

where a_k are the coefficients of the predictor. The z-transform of this equation is given by:

$$\overline{X}(z) = \sum_{k=1}^{L} a_k z^{-k} X(z)$$

Thus, using LPC coefficients, a sound can be represented in terms of coefficients to an IIR filter. [33] *Matlab* has a built in function *lpc* which calculates the *L lpc* coefficients given a signal and a number *L*. These coefficients can have an imaginary component, thus the magnitude can be calculated and stored.

D.1.9 Mel-Frequency Cepstral Coefficients

Mel-frequency Cepstral Coefficients (MFCC) are the coefficients of the Fourier transform representation of the log magnitude spectrum. After the STFT is calculated, FFT bins are grouped and smoothed according to mel-frequency scaling. 13 coefficients are created to represent each frame. [100] Each of the thirteen coefficients is averaged over time to give an array of length thirteen.

D.1.10 Wavelet Features

The wavelet transform provides a temporal and frequency representation of a signal. This is useful because often a certain frequency component occurs once at a certain instance in time, and one needs to calculate the frequency and time of this occurrence. The wavelet transform uses a filter bank framework as shown in Figure 125. The filter bank decomposes a signal into a low resolution approximation signal and detail signals, realized by a high pass and low pass FIR filter pair. The filter pairs used in this research is a Daubechies pair of length eight. [44] The filtered signals are down-sampled by 2 ensuring that the length of the original signal is preserved. The low resolution approximation can be filtered again to yield another signal pair. This filter process can be repeated *K* times when the signal is of length 2^K (zero padding is used in this feature to ensure a multiple of 2). Thus there will be *K* detail signals and one low resolution approximation. The majority of the detail signals are sufficiently small in magnitude that they can be ignored in compression. [102] *Matlab Uv_Wave Toolbox* (developed by University of Vigo) is used in this research to translate

signals to the wavelet domain. A feature is extracted from the wavelet domain by taking the *K* average values for each sub band. [179] Another feature is calculated by finding the variance from the mean for each sub band.

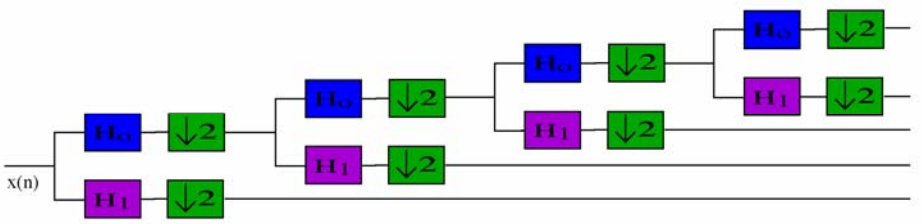


Figure 125 - Diagram of a four level filter bank wavelet transform using FIR filter pairs H0 (low pass) and H1 (high pass).

D.1.11 Subband Analysis

Sub band analysis techniques are also employed to determine energy in specific bands. There are four bands: 0-200 Hz, 200-1000 Hz, 1000-3000 Hz, 3000-20,000 Hz. Linear phase FIR filters of length 71 are designed to separate the signals into four separate subbands [102]. In some experiments, the energy in each band is measured during the attack phase, which gives a rough estimation as to what modes of the sound are being excited.

Appendix

E

E
Computer Music Languages
Programming Languages for Music DSP

This chapter introduces the computer music programming languages used in this dissertation. A brief synopsis of the strengths and functionality of each language are described.

E.1 *STK Toolkit*

The Synthesis ToolKit[49] (STK) [33] is a set of open source audio signal processing and algorithmic synthesis classes written in the C++ programming language led by Professor Perry R. Cook and Professor Gary Scavone, while they were at Stanford University. STK was designed to facilitate development of music synthesis and audio processing software, with an emphasis on cross-platform functionality, real-time control, ease of use, and educational example code. The Synthesis ToolKit is particularly novel because of its collection of physical models of instruments. Physical models model the time domain physics of the instrument and take advantage of the one-dimensional paths in many systems (ex. strings, narrow pipes) replacing them with delay lines (waveguides) [33].

[49] http://ccrma.stanford.edu/software/stk/

E.2 *ChucK*

ChucK[50] [189] is a concurrent, strongly-timed audio programming language developed under Professor Perry R. Cook by Ge Wang and the SoundLab team at Princeton University. Programmable on-the-fly, it is said to be *strongly-timed* because of its precise control over time. Concurrent code modules that independently process different parts of the data can be added during runtime, and precisely synchronized. *ChucK* has STK modules imported directly into the language. It has a real-time architecture making it easy to prototype performance applications for controllers and robotics. *ChucK* is freely available online.

E.3 *Marsyas*

Marsyas [51] [177] is a software framework for rapid prototyping and experimentation with audio analysis and synthesis with specific emphasis on music signal and music information retrieval. It was developed by Professor George Tzanetakis and his team at University of Victoria. It is based on a data-flow architecture that allows networks of processing objects to be created at run-time. A variety of feature extraction algorithms both for audio and general signals are provided. In addition *Marsyas* provides integrated support for machine learning and classification using algorithms such a k-nearest neighbor, Gaussian mixture models and artificial neural networks. *Marsyas* is freely available online.

E.4 *Pure data* (pd)

Pure data[52] (pd) is a real-time graphical programming environment for audio, MIDI and video signal processing, developed by Professor Miller S. Puckette at University of San Diego. Pd's visual environment allows for users who do not have a background in computer science with a suitable structure to learn how to

[50] http://chuck.cs.princeton.edu/
[51] http://*Marsyas*.sourceforge.net
[52] http://puredata.info/

understand signals. Advanced users can write external classes which can be used to extend the capabilities of the pre-written code. Pure-data is freely available online.

E.5 *MAX/MSP*

MAX/MSP[53] is a commercially available graphical environment for audio, video and multimedia signal processing. Developed over the signal processing architecture of pure-data, CEO David Zicarelli and his team at Cycling ’74 in San Francisco have created a new paradigm for GUI-based new media programming.

[53] http://www.cycling74.com/

Appendix

F

F Publications

Chronological Ordering of Work Disseminated for this Dissertation

F.1 Refereed Academic Publications

F.1.1 Refereed Journal Papers

1. Kapur, A., Wang, G., Davidson, P., & P.R. Cook, "Networked Performance: A Dream worth Dreaming?", *Organised Sound*, 10(3), pp. 209-219, October 2005.
2. Kapur, A., Davidson, P., Cook, P.R., Driessen, P.F. & W.A. Schloss, "Preservation and Extension of Traditional Techniques: Digitizing North Indian Performance", *Journal of New Music Research*, 34(3), pp. 227-236, September 2005.
3. Kapur, A., Davidson, P., P.R. Cook, P.F. Driessen, & W.A. Schloss, "Evolution of Sensor-Based *ETabla*, *EDholak*, and *ESitar*", *Journal of ITC Sangeet Research Academy*, vol. 18, Kolkata, India, November 2004.
4. Kapur, A., Essl, G., Davidson, P. & P. R. Cook, "The Electronic Tabla Controller", *Journal of New Music Research*, 32(4), pp. 351-360, 2003.

F.1.2 Refereed Conference Papers

5. Benning, M., Kapur, A., Till, B., and G. Tzanetakis. “MultiModal Sensor Analysis on Sitar Performance: Where is the Beat?” *Proceeding of the IEEE International Workshop on Multimedia Signal Processing (MMSP)*. Crete, Greece, October 2007.
6. Kapur, A., Percival, G., Lagrange, M., and G. Tzanetakis. “Pedagogical Transcription for Multimodal Sitar Performance,” *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria, September 2007.
7. Kapur, A., Trimpin, Singer, E., Suleman, A., and G. Tzanetakis. “A Comparison of Solenoid-Based Strategies for Robotic Drumming,” *Proceeding of the International Computer Music Conference (ICMC)*, Copenhagen, Denmark, August 2007.
8. Benning, M., Kapur, A., Till, B., Tzanetakis, G., and P. F. Driessen. “A Comparative Study on Wearable Sensors for Signal Processing on the North Indian Tabla,” *Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, Victoria, Canada, August 2007.
9. Kapur, A., Singer, E., Benning, M. S., Tzanetakis, G. and Trimpin. "Integrating Hyperinstruments, Musical Robots, & Machine Musicianship for North Indian Classical Music," *Proceedings of the International Conference for New Interfaces for Musical Expression*, New York, USA, June 2007.
10. Kapur, A., Tzanetakis, G., Schloss, A., Driessen, P.F., & E. Singer, “Towards the One-Man Indian Computer Music Performance System,” *Proceedings of the International Computer Music Conference*, New Orleans, USA, November 2006.
11. Kapur, A., Tindale, A.R., Benning M.S., & P.F. Driessen, “The *KiOm*: A Paradigm for Collaborative Controller Design,” *Proceedings of the*

International Computer Music Conference (ICMC), New Orleans, USA, November 2006.

12. Kapur, A. & E. Singer, "A Retrieval Approach for Human/Robot Musical Performance," *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Victoria, Canada, October 2006.
13. Tzanetakis, G., Kapur A., & A.R. Tindale, "Learning Indirect Acquisition of Instrumental Gestures using Direct Sensors," *Proceedings of the IEEE Workshop on Multimedia Signal Processing (MMSP)*. Victoria, Canada, October 2006.
14. Kapur, A., "21st Century Ethnomusicology", *Proceedings of Music and the Asian Diaspora*, Princeton, New Jersey, April 2006. Invited Keynote Speaker.
15. Tzanetakis, G., Kapur A., & R.I. McWalter, "Subband-based Drum Transcription for Audio Signals," *Proceedings of the IEEE International Workshop on Multimedia Signal Processing (MMSP)*. Shanghai, China, November 2005.
16. Kapur, A. "Past to Present: Evolution and Preservation of Traditional Techniques using Computer Music Theory", *MusicAcoustica*, Beijing, China, October 2005.
17. Kapur, A., Kapur, A., Virji-Babul, N., Tzanetakis, G. & P.F. Driessen, "Gesture-Based Affective Computing on Motion Capture Data", *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*. Beijing, China. October 2005.
18. Kapur, A., Tzanetakis, G., Virji-Babul, N., Wang, G., & P.R. Cook, "A Framework for Sonification of Vicon Motion Capture Data", *Proceedings of the International Conference on Digital Audio Effects (DAFX)*, Madrid, Spain, September 2005.
19. Kapur, A., McWalter, R. I., & G. Tzanetakis, "New Music Interfaces for Rhythm-Based Retrieval", *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, England, September 2005.

20. Kapur, A., “A History of Robotic Musical Instruments”. *Proceedings of the International Computer Music Conference (ICMC).* Barcelona, Spain, September 2005.
21. Driessen, P., Schloss, W. A., Tzanetakis, G., McNally, K. & A. Kapur, “Studio Report: University of Victoria Music Intelligence and Sound Technology Interdisciplinary Centre (MISTIC)”. *Proceedings of the International Computer Music Conference (ICMC).* Barcelona, Spain, September 2005.
22. Kapur, A., Yang, E.L., Tindale, A.R., & P.F. Driessen, “Wearable Sensors for Real-Time Musical Signal Processing”. *Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM).* Victoria, Canada. August, 2005.
23. Tindale, A.R., Kapur, A., Tzanetakis, G., Driessen, P.F., & W. A. Schloss, “A Comparison of Sensor Strategies for Capturing Percussive Gestures”, *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME).* Vancouver, Canada, May 2005.
24. Wang, G., Misra, A., Kapur, A., & P.R. Cook, “Yeah, *CHUCK* IT=> Dynamic, Controllable Interface Mapping”, *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME),* Vancouver, Canada, May 2005.
25. Tindale, A.R., Kapur, A., Tzanetakis, G., & W.A Schloss, "Indirect Acquisition of Percussion Gestures Using Timbre Recognition," *Proceedings of the Conference on Interdisciplinary Musicology (CIM),* Montreal, Canada, March 2005.
26. Kapur, A., Davidson, P., Cook, P. R., Driessen, P., & W. A. Schloss, "Digitizing North Indian Performance," *Proceedings of the International Computer Music Conference (ICMC),* Miami, Florida, November 2004. *Winner of the Journal of New Music Research Distinguished Best Paper Award ICMC 2004.*

27. Tindale, A., Kapur, A., & I. Fujinaga, "Towards Timbre Recognition of Percussive Sounds", *Proceedings of the International Computer Music Conference (ICMC)*, Miami, Florida, November 2004.
28. Kapur, A., Benning, M., & G. Tzanetakis, "Query-By-Beat-Boxing: Music Retrieval for the DJ," *Proceedings of International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Spain, October 2004.
29. Tindale, A., Kapur, A., Tzanetakis, G., & I. Fujinaga, "Retrieval of Percussion Gestures Using Timbre Classification Techniques", *Proceedings of International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Spain, October 2004.
30. Kapur, A., Tzanetakis, G., & P.F. Driessen, "Audio-Based Gesture Extraction on the *ESitar* Controller", *Proceedings of Conference on Digital Audio Effects (DAFX)*, Naples, Italy, October 2004.
31. Kapur A., “Digitizing to Preserve Traditional North Indian Technique”, *Society of Ethnomusicology Northwest Chapter Meeting*, Victoria, BC, February 2004. *Winner: Thelma Adamson Best Student Paper Award SEM 2004.*
32. Kapur, A., Lazier, A., Davidson, P., Wilson, R. S., & P.R. Cook, "The Electronic Sitar Controller", *Proceedings of the International Conference on New Instruments for Musical Expression (NIME)*, Hamamatsu, Japan, June 2004.
33. Kapur, A., Essl, G., Davidson, P. & P. R. Cook, "The Electronic Tabla Controller", *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, Dublin, Ireland, pp. 77-81. May 2002.

F.2 Publications by Chapter

Chapter	Appendix F Reference Number
5	2, 3, 4, 14, 16, 19, 23, 26, 31, 33
6	1, 2, 3, 14, 16, 23, 26, 31
7	2, 3, 5, 6, 9, 10, 12, 13, 14, 16, 19, 21, 26, 31, 32
8	5, 8, 10, 11, 14, 16, 17, 18, 21, 22
9	7, 9, 10, 12, 20
10	5, 9, 10
11	9, 10, 12, 15, 19
12	6
13	13, 30
14	17, 18

F.3 Interdisciplinary Chart

	Sensors	Robotics	Machine Learning
Computer Science	1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 16, 17, 18, 19, 22, 23, 24, 26, 30, 31, 32, 33	5, 9, 10, 12, 14, 16, 20, 24	13, 14, 15, 16, 17, 18, 19, 25, 27, 28, 29, 30
Music	1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 19, 22, 23, 24, 26, 30, 31, 32, 33	5, 7, 9, 10 12, 14, 20, 24	13, 14, 15, 16, 19, 25, 27, 28, 29, 30
Electrical Engineering	1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 16, 19, 22, 23, 24, 26, 30, 31, 32, 33	5, 7, 9, 10 12, 14, 16, 20, 24	13, 14, 16, 19, 30
Mechanical Engineering	7, 9, 10	7, 9, 10, 20	
Psychology	17, 18		17, 18

Bibliography

References, History, Citations

[1] R. Aimi, "New Expressive Percussion Instruments," in *Media Laboratory*, vol. Masters. Boston: MIT, 2002.

[2] D. L. Alkon, "Memory Storage and Neural Systems," in *Scientific America*, pp. 42-50, 1989.

[3] M. Alonso, G. Richard, and D. Bertrand, "Tempo and Beat Estimation of Musical Signals," presented at International Conference on Music Information Retrieval, Barcelona, Spain, 2004.

[4] C. G. Atkeson, J. Hale, F. Pollick, M. Riley, S. Kotosaka, S. Schaal, T. Shibata, G. Tevatia, S. Vijayakumar, A. Ude, and M. Kawato, "Using Humanoid Robots to Study Human Behavior," *IEEE Intelligent Systems: Special Issue on Humanoid Robotics*, vol. 15, pp. pp. 46-56, 2000.

[5] J. J. Aucouturier, F. Pachet, and P. Hanappe, "From Sound Sampling to Song Sampling," presented at International Conference on Music Information Retrieval, Barcelona, Spain, 2004.

[6] M. Babbitt, "Who Cares if You Listen?," *High Fidelity*, 1958.

[7] S. Bagchee, *NAD: Understanding Raga Music*. Mumbai, India: Ceshwar Business Publications, Inc. , 1998.

[8] N. A. Baginsky, "The Three Sirens: A Self Learning Robotic Rock Band."

[9] C. Bahn and D. Trueman, "Interface: Electronic Chamber Ensemble," presented at International Conference on New Interfaces for Musical Expression, Seattle, WA, 2001.

[10] A. Barbosa, "Displaced Soundscapes: A Survey of Networked Systems for Music and Art Creation," *Leonardo Music Journal*, vol. 13, pp. 53-59, 2003.

[11] R. Barger, S. Church, A. Fukada, J. Grunke, D. Keisler, B. Moses, B. Novak, B. Pennycook, Z. Settel, J. Stawn, P. Wiser, and W. Woszczyk, "AES White Paper:

Networking Audio and Music Using Internet2 and Next Generation Internet Capabilities," presented at Audio Engineering Society, New York, 1998.

[12] Bean, "Techno Taiko with a Twist," in *Electronic Musician Magazine*, pp. pp. 124-125, 1998.

[13] M. J. Blackand and Y. Yacoob, "Recognizing Facial Expressoins in Image Sequences Using Local Parameterized Models of Image Motion," *International Journal of Computer Vision*, vol. 25, pp. 23-48, 1997.

[14] T. Blaine and C. Forlines, "Jam-O-World: Evolution of the Jam-O-Drum Multi-Player Musical Controller into the Jam-O-Whirl Gaming Interface," presented at Proceedings of the International Conference on New Interfaces for Musical Expression (NIME), Dublin, 2002.

[15] P. Boersma, "Accurate Short-Term Analysis of the Fundamental Frequency and the Harmonics-to-Noise Ratio of a Sampled Sound," presented at Proceedings of the Institute of Phonetic Sciences, Amsterdam, 1993.

[16] P. Brossier, J. Bello, and M. Plumbley, "Fast Labelling of Notes in Music Signals," presented at International Conference on Music Information Retrieval, Barcelona, Spain, 2004.

[17] R. G. Brown and P. Y. C. Hwang, *Introduction of Random Signals and Applied Kalman Filtering.*: John Wiley & Sons, Inc. , 1992.

[18] A. Camurri, "Multimodal Interfaces for Expressive Sound Control," presented at International Conference on Digital Audio Effects, Naples, Italy, 2004.

[19] E. C. T. A. o. E. Carr, *Growing Pains*. Toronto: Oxford University Press, 1946.

[20] T. Cemgil, "Bayesian Music Transcription," vol. Ph.D. Netherlands: Radboud University of Nijmegen, 2004.

[21] C. Chafe and R. Leistikow, "Levels of Temporal Resolution in Sonfication of Network Performance," presented at International Conference On Auditory Display, Helsinki, Finland, 2001.

[22] C. Chafe, R. S. Wilson, R. Leistikow, D. Chisholm, and G. Scavone, "A Simplified Approach to High Quality Music and Sound over IP," presented at International Conference on Digital Audio Effects, Verona, Italy, 2000.

[23] C. Chafe, R. S. Wilson, and D. Walling, "Physical Model Synthesis with Applications to Internet Acoustics," presented at IEEE International Conference on Acoustics, Speech, and Signal Processing, 2002.

[24] J. Chen and A. Chen, "Query by Rhythm: An Approach for Song Retrieval in Music Databases," presented at International Workshop on Research Issues in Data Engineering, Continuous Media Databases and Applications, 1998.

[25] L. S. Chen, T. S. Huang, T. Miyasato, and R. Nakatsu, "MultiModal Human Emotion/Expression Recognition," presented at International Conference on Automatic Face and Gesture Recognition, Nara, Japan, 1998.

[26] P. R. Cook, "A Meta-Wind-Instrument Physical Model, and a Meta-Controller for Real Time Performance Control," presented at Proceedings of the International Computer Music Conference, 1992.

[27] P. R. Cook, "Physically Informed Sonic Modeling (PhISM): Percussive Synthesis," *Computer Music Journal*, vol. 21, 1997.

[28] P. R. Cook, "Pico I for Seashells and Interactive Electronics," presented at International Mathematic Symposium, Rovaniemi, Finland, 1997.

[29] P. R. Cook, *Music, Cognition and Computerized Sound: An Introduction to Psychoacoustics*. Cambridge, MA: MIT Press, 1999.

[30] P. R. Cook, "Principles for Designing Computer Music Controllers," presented at International Conference on New Interfaces for Musical Expression, Seattle, WA, 2001.

[31] P. R. Cook, "Principles for Designing Computer Music Controllers," presented at International Conference for New Interfaces for Musical Expression., Seattle, WA, 2001.

[32] P. R. Cook, "Serial Communications Example," 2001.

[33] P. R. Cook, *Real-Time Synthesis for Interactive Applications*. Natick, MA: AK Peters, 2002.

[34] P. R. Cook and C. Leider, "SqueezeVox: A New Controller for Vocal Synthesis Models," presented at Proceedings of the International Computer Music Conference, 2000.

[35] P. R. Cook, D. Morrill, and J. O. Smith, "A MIDI Control and Performance System for Brass Instruments," presented at Proceedings of the International Computer Music Conference, 1993.

[36] P. R. Cook and G. Scavone, "The Synthesis Toolkit (STK)," presented at International Computer Music Conference, Beijing, China, 1999.

[37] P. R. Cook and D. Trueman, "NBody: Interactive Multidirectional Musical Instrument Body Radiation Simulations, and a Database of Measured Impulse Responses," presented at International Computer Music Confrence, Ann Arbor, USA, 1998.

[38] D. R. Courtney, "Repair and Maintenance of Tabla," *Percussive Notes*, vol. 31, pp. 29-36, 1993.

[39] D. R. Courtney, *Fundamentals of Tabla: Complete Reference for Tabla*, vol. 1. Houston, TX: Sur Sangeet Services, 1995.

[40] R. Cowie, E. Douglas-Cowie, N. Tsapatsoulis, G. Votsis, S. Kollias, W. Fellenz, and J. G. Taylor, "Emotion Recognition in Human-Computer Interaction," *IEEE Signal Precessing Magazine*, vol. 18, pp. 32-80, 2001.

[41] S. Dahl, "Playing the Accent - Comparing Striking Velocity and Timing in an Ostinato Rhythm Performed by Four Drummers," *Acta Acustica united with Acustica*, vol. 90, 2004.

[42] R. B. Dannenberg, "An On-line Algorithm for Real-Time Accompaniment," presented at International Computer Music Conference, Paris, France, 1984.

[43] R. B. Dannenberg, B. Brown, G. Zeglin, and R. Lupish, "McBlare: A Robotic Bagpipe Player," presented at International Conference on New Interfaces for Musical Expression, Vancouver, Canada 2005.

[44] I. Daubechies, "Orthonormal Bases of Compactly Supported Wavelets," *Commun, Pure Applied Math*, vol. 41, pp. 909-996, 1988.

[45] M. DeMeijer, "The Contribution of General Features of Body Movement to the Attribution of Emotion," *Journal of Nonverbal Behavior*, vol. 13, pp. 274-268, 1989.

[46] M. Denki, "Tsukuba Series," Japan.

[47] B. C. Deva, *Indian Music*. New Delhi: Indian Council for Cultural Relations, 1974.

[48] S. Dixon, "A Lightweight Multi-Agent Musical Beat Tracking System," presented at Pacific Rim International Conference on Artificial Intelligence, 2000.

[49] S. Dixon, E. Pampalk, and G. Widmer, "Classification of Dance Music by Periodicity Patterns," presented at International Conference on Music Information Retrieval, 2003.

[50] M. V. Dorssen, "The Cell."

[51] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification, Second Edition*: Wiley Interscience, 2000.

[52] I. Essa and A. Pentland, "Coding Analysis Interpretation Recognition of Facial Expressions," *IEEE Transaction Pattern Analysis and Machine Intelligence*, vol. 19, pp. 757-763, 1997.

[53] G. Essl, "Physical Wave Propagation Modeling for Real-Time Synthesis of Natural Sounds," in *Princeton Computer Science Department.*, vol. Ph.D. Princeton, NJ: Princeton University, 2002.

[54] G. Essl and P. R. Cook, "Banded Waveguides: Towards Physical Modeling of Bar Percussion Instruments," presented at International Computer Music Conference, Beijing, China, 1999.

[55] S. S. Fels and G. E. Hinton, "Glove-TalkII: A neural Network Interface which Maps Gestures to Parrallel Formant Speech Synthesizer Controls.," *IEEE Transactions on Neural Networks*, vol. 9, pp. 205-212, 1998.

[56] R. Fernandez and R. W. Picard, "Modeling Driver's Speech under Stress," presented at ISCA Workshop on Speech and Emotions, Belfast, 2000.

[57] J. L. Flanagan and R. M. Golden, "Phase Vocoder," *Bell System Technical Journal*, pp. 1493-1509, 1966.

[58] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments.*, Second ed. New York: Springer Verlag, 1998.

[59] G. Föllmer, "Musikmachen im Netz Elektronische, ästhetische und soziale Strukturen einer partizipativen Musik," vol. Ph.D.: Martin-Luther-Universität Halle-Wittenberg, 2002.

[60] J. Freeman, C. Ramakrishnan, K. Varnik, M. Neuhau, P. Burk, and D. Birchfield, "Adaptive High-Level Classification of Vocal Gestures Within a Networked

Sound Instrument," presented at International Computer Music Conference, Miami, Florida, 2004.

[61] O. Gillet and G. Richard, "Automatic Labelling of Tabla Symbols," presented at International Conference on Music Information Retrieval 2003.

[62] M. Goto and Y. Maraoka, "Real-time Rhythm Tracking for Drumless Audio Signals - Chord Change Detection for Musical Decisions," presented at International Conference in Artificial Intelligence: Workshop on Computational Auditory Scene Analysis, 1997.

[63] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An Experimental Comparison of Audio Tempo Induction Algorithms," *IEEE Transactions on Speech and Audio Processing* vol. 14, 2006.

[64] F. Gouyon, F. Pachet, and O. Delerue, "On the use of ZeroCrossing Rate for an Application of Classification of Percussive Sounds," presented at International Conference on Digital Audio Effects, 2000.

[65] M. Gurevich, C. Chafe, G. Leslie, and S. Taylor, "Simulation of Networked Exsemble Performance with Varying Time Delays: Characterization of Ensemble Accuracy," presented at International Computer Music Conference, Miami, Florida, 2004.

[66] M. Gurevich and S. v. Muehlen, "The Accordiatron: A MIDI Controller For Interactive Music," presented at International Confernece on New Interfaces for Musical Expression, Seattle, WA, 2001.

[67] T. Hahn and C. Bahn, "Pikapika - The Collaborative Composition of and Interactive Sonic Character," *Organized Sound*, vol. 7, 2003.

[68] A. Z. Hajian, D. S. Sanchez, and R. D. Howe, "Drum Roll: Increasing Bandwidth Through Passive Impedance Modulation," presented at IEEE Robotics and Automation Conference, Albuquerque, USA, 1997.

[69] M. Helmuth, "Sound Exchange and Performance on Internet2," presented at International Computer Music Conference, 2000.

[70] P. Herrera, G. Gouyon, and S. Dubnov, "Automatic Classification of Musical Instrument Sounds," *Journal of New Music Research*, vol. 32, pp. 3-21, 2003.

[71] H. Hesse, *The Glass Bead Game (Magister Ludi)*. New York: Henry Holt and Company, 1943.

[72] H. Ian, E. Frank, and M. Kaufmann, *Data Mining: Practical Machine Learning Tools with Java Implemenations*. San Francisco, CA, 2000.

[73] H. Ishii, "Bottles: A Transparent Interface as a Tribute to Mark Weiser," *IEICE Transactions on Information and Systems*, vol. E87, 2004.

[74] H. Ishii, R. Fletcher, J. Lee, S. Choo, J. Berzowska, C. Wisneski, C. Cano, A. Hernandez, and C. Bulthaup, "MusicBottles," presented at SIGGRAPH, 1998.

[75] JBot, "Captured! by Robots Web Site."

[76] S. Jorda, "Faust Music On Line (FMOL): An Approach to Real-Time Collective Composition on the Internet," *Leonardo Music Journal*, vol. 9, pp. 5-12, 1999.

[77] S. Jorda, "Afasia: the Ultimate Homeric One-Man-Multimedia-Band," presented at International Conference on New Interfaces for Musical Expression, Dublin, Ireland, 2002.

[78] S. Jorda and A. Barbosa, "Computer Supported Cooperative Music: Overview of Research Work and Projects at the Audiovisual Institute – UPF," presented at MOSART Workshop on Current Research Directions in Computer Music, 2001.

[79] M. Kajitani, "Development of Musician Robots," *Journal of Robotics and Mechatronics*, vol. 1, pp. pp. 254-255, 1989.

[80] M. Kajitani, "Simulation of Musical Performances," *Journal of Robotics and Mechatronics*, vol. 4, pp. pp. 462-465, 1992.

[81] T. Kaneda, S. Fujisawa, T. Yoshida, Y. Yoshitani, T. Nishi, and K. Hiroguchi, "Subject of Making Music Performance Robots and Their Ensemble," presented at ASEE/IEEE Frontiers in Education Conference, San Juan, Puerto Rico, 1999.

[82] B. S. Kang, C. H. Han, S. T. Lee, D. H. Youn, and C. Lee, "Speaker Dependent Emotion recognition using Speech Signals," presented at ICSLP, 2000.

[83] A. Kapoor, S. Mota, and R. W. Picard, "Towards a Learning Companion that Recognizes Affect," presented at Emotional and Intelligent II: The Tangled Knot of Social Cognition. AAAI Fall Symposium, North Falmouth, MA, 2001.

[84] A. Kapur, M. Benning, and G. Tzanetakis, "Query-By-Beatboxing: Music Information Retrieval for the DJ," presented at International Conference on Music Informatoin Retrieval, Barcelona, Spain, 2004.

[85] A. Kapur, A. Lazier, P. Davidson, R. S. Wilson, and P. R. Cook, "The Electronic Sitar Controller," presented at International Conference on New Interfaces for Musical Expression, Hamamatsu, Japan, 2004.

[86] A. Kapur, G. Tzanetakis, and P. F. Driessen, "Audio-Based Gesture Extraction on the ESitar Controller," presented at International Conference on Digital Audio Effects, Naples, Italy, 2004.

[87] A. Kapur, G. Tzanetakis, N. Virji-Babul, G. Wang, and P. R. Cook, "A Framework for Sonification of Vicon Motion Capture Data," presented at International Conference on Digital Audio Effects, Madrid, Spain, 2005.

[88] A. Kapur, E. L. Yang, A. R. Tindale, and P. F. Driessen, "Wearable Sensors for Real-Time Musical Signal Processing," presented at IEEE Pacific Rim Conference

Victoria, Canada, 2005.

[89] T. v. Kasteren, "Realtime Tempo Tracking using Kalman Filtering," in *Computer Science*, vol. Masters. Amsterdam: University of Amsterdam, 2006.

[90] V. Kaul, "Talvin Singh," in *India Today*, 1998.

[91] J. B. Keller and S. I. Rubinow, "Asymptotic Solution of Eigenvalue Problems," *Annuals of Physics*, vol. 9, pp. 24-75, 1960.

[92] J. Kippen, "Tabla Drumming and the Human-Computer Interaction," *The World of Music*, vol. 34, pp. 72-98, 1992.

[93] A. P. Klapuri, "Automatic Music Transcription as we Know it Today.," *Journal of New Music Research*, vol. 33, pp. 269-282, 2004.

[94] R. Knapp, "Bioelectric Controller for Computer Music Applications," *Computer Music Journal*, vol. 14, pp. 42-47, 1990.

[95] K. S. Kothari, *Indian Folk Musical Instruments*. New Delhi, India: Sangeet Natak Akademi, 1968.

[96] S. Y. Kung, *Digital Neural Networks*. Englewood Cliffs, NJ: PTR Prentice Hall, 1993.

[97] A. Lazier and P. R. Cook, "Mosievius: Feature-Based Interactive Audio Mosaicing," presented at International Conference on Digital Audio Effects, London, England, 2003.

[98] G. Lewis, "Too Many Notes: Computers, Complexity and Culture in Voyager," *Leonardo Music Journal*, vol. 10, pp. 33-39, 2000.

[99] R. P. Lipmann, "An Introduction to Computing with Neural Nets," in *IEEE ASSP Magazine*, 1987.

[100] B. Logan, "Mel-frequency Cepstral Coefficients for Music Modeling," presented at International Conference on Music Information Retrieval, 2000.

[101] A. Loscos, Y. Wang, and W. J. J. Boo, "Low Level Descriptors for Automatic Violin Transcription," in *Proceedings of International Conference for Music Information Retrieval (ISMIR)*. Victoria, BC, 2006.

[102] W.-S. Lu, *Digital Signal Processing III Lecture Notes.*, 2003.

[103] T. Machover and J. Chung, "Hyperinstruments: Musically Intelligent and Interactive Performance and Creativity Systems," presented at International Computer Music Conference, 1989.

[104] C. MacMurtie, "Amorphic Robot Works."

[105] M. Marshall, M. Rath, and B. Moynihan, "The Virtual Bodhran - The Vodhran.," in *Proceedings of the International Conference on New Interfaces for Musical Expression.* . Dublin, Ireland, 2002.

[106] K. Mase and T. Yonezawa, "Body, Clothes, Water and Toys: Media Towards Natural Music Expressions with Digital Sounds," presented at International Conference on New Interfaces for Musical Expression, Seattle, WA, 2001.

[107] M. Mathews and A. Schloss, "The RadioDrum as a Synthesizer Controller," presented at International Computer Music Conference, 1989.

[108] A. Mazalek and T. Jehan, "Interacting with Music in a Social Setting," presented at Conference on Human Factors in Computing Systems, 2000.

[109] K. McElhone, *Mechanical Music*. USA: Shire Publications, 1999.

[110] R. B. McGhee, E. R. Bachmann, X. Yun, and M. J. Zyda, "Real-Time Tracking and Display of Human Limb Segment Motions Using Sourceless Sensors and a

Quaternion-Based Filtering Algorithm - Part 1: Theory," vol. Technical Report NPS-MV-01-001. Monterery, CA: Naval Postgraduate School, 2000.

[111] Meer, *Hindustani Music in the 20th Century*.

[112] R. R. Menon, *Discovering Indian Music*. Mumbai, India: Somaiya Publications PVT. LTD. , 1974.

[113] D. Merrill, "Head-Tracking for Gestural and Continuous Control of Parameterized Audio Effects," presented at International Conference on New Interfaces for Musical Expression, Montreal, Canada, 2003.

[114] T. M. Mitchell, *Machine Learning*: McGraw-Hill Companies, Inc. , 1997.

[115] G. Monahan, "Kinetic Sound Environments as a Mutation of the Audio System," presented at Musicworks, Toronto, Canada, 1995.

[116] D. Morrill and P. R. Cook, "Hardware, Software, and Compositional Tools for a Real-Time Improvised Solo Trumpet Work," presented at Proceedings of the International Computer Music Conference, 1989.

[117] T. Nakano, J. Ogata, M. Goto, and Y. Hiraga, "A Drum Pattern Retrieval Method by Voice Percussion," presented at International Conference on Music Information Retrieval 2004.

[118] H. Newton-Dunn, H. Nakono, and J. Gibson, "Block Jam," presented at SIGGRAPH, 2002.

[119] C. Nichols, "The vBow: A Virtual Violin Bow Controller for Mapping Gesture to Synthesis with Haptic Feedback," *Organized Sound*, vol. 7, 2002.

[120] H. Ohta, H. Akita, and M. Ohtani, "The Development of an Automatic Bagpipe Playing Device," presented at International Computer Music Conference, Tokyo, Japan, 1993.

[121] D. Overholt, "The Overtone Violin," presented at International Conference on New Interfaces for Musical Expression, Vancouver, 2005.

[122] F. Pachet, "The Continuator: Musical interaction with Style " presented at International Computer Music Conference, Goteborg, Sweden, 2002.

[123] M. Pantic, "Toward an Affect-Sensitive Multimodal Human-Computer Interaction," *IEEE*, vol. 91, 2003.

[124] J. Paradiso, "The Brain Opera Technology: New Instruments and Gestural Sensors for Musical Interaction and Performance," *Journal of New Music Research*, vol. 28, pp. 130-149, 1999.

[125] J. Paradiso, "Wearable Wireless Sensing for Interactive Media," presented at First International Workshop on Wearable & Implantable Body Sensor Networks, London, 2004.

[126] J. Patten, H. Ishii, and G. Pangaro, "Sensetable: A Wireless Object Tracking Platform for Tangible User Interfaces," presented at Conference on Human Factors in Computing Systems, 2001.

[127] J. Patten, B. Recht, and H. Ishii, "Audiopad: A tag-based Interface for Musical Performance," presented at International Conference on New Interfaces for Musical Expression, Dublin, Ireland, 2002.

[128] J. Paulus and A. Klapuri, "Measuring the Similarity of Rhythmic Patterns," presented at International Conference on Music Information Retrieval, 2002.

[129] M. Peinado, B. Herbelin, M. M. Wanderley, B. L. Callennec, D. Thalmann, and D. Meziatil, "Towards Configuarable Motion Capture with Polorized Inverse Kinematics " *Sensor* 2004.

[130] R. W. Picard, *Affective Computing*: MIT Press, 1997.

[131] R. W. Picard, "Towards Computers that Recognize and Respond to User Emotions," *IBM System Journal*, vol. 39, pp. 705-719, 2001.

[132] R. W. Picard and J. Healey, "Affective Wearables," *Personal Technologies*, vol. 1, pp. 231-240, 1997.

[133] F. E. Pollick, H. Paterson, A. Bruderlin, and A. J. Sanford, "Perceiving Affect from Arm Movement," *Cognition*, vol. 82, pp. B51-B61, 2001.

[134] B. Pritchard and S. S. Fels, "The GRASSP Environment: Gesturally-Realized Audio, Speech and Song Performance," presented at International Conference on New Interfaces for Musical Expression, Paris, 2005.

[135] M. Puckette, "Combining Event and Signal Processing in the MAX Graphical Programming Environment," *Computer Music Journal*, vol. 15, pp. 68-77, 1991.

[136] M. Puckette, "*Pure data*: Another Integrated Computer Music Environment," presented at Second Intercollege Computer Music Concerts, Tachikawa, Japan, 1996.

[137] M. Puckette, V. Sorensen, and R. Steiger, "Lemma1," presented at Performed Live at Milos Jazz Club at the International Computer Music Conference, Thessaloniki, Greece, 1997.

[138] J. R. Quinlan, "Learning with Continuous Classes," presented at 5th Australian Joint Conference on Artificial Intelligence, 1992.

[139] G. W. Raes, "Automations by Godfried-Willem Raes."

[140] C. V. Raman, "Experiments with Mechanically played Violins," presented at Indian Association of Cultivation of Science, India, 1920.

[141] C. Roads, "The Tsukuba Musical Robot," *Computer Music Journal*, vol. 10, pp. 39-43, 1986.

[142] J. H. Roh and L. Wilcox, "Exploring Tabla Using Rhythmic Input.," presented at CHI, Denver, CO, 1995.

[143] T. D. Rossing, *The Science of Sound*: Addison-Wesley Publishing Company, 1999.

[144] T. D. Rossing, F. R. Moore, and P. A. Wheeler, *The Science of Sound*. San Francisco: Addison Wesley, 2002.

[145] R. Rowe, *Machine Musicianship*. Cambridge, MA: MIT Press, 2004.

[146] R. Rowe and N. Rolnick, "The Technophobe and the Madman: An Internet2 Distributed Musical," presented at International Computer Music Conference, Miami, Florida, 2004.

[147] J. Ryan and C. Salter, "TGarden: Wearable Instruments and Augmented Physicallity," presented at International Conference on New Interfaces for Musical Expression, Montreal, Canada, 2003.

[148] R. Sanyal, *Musical Instruments: The Indian Approach*: Sangeet Research Academy, 1985.

[149] F. A. Saunders, "Journal of Acoustic Society of America," *9*, 1937.

[150] H. Sawanda, N. Onoe, and S. Hashimoto, "Acceleration Sensor as an Input Device for Musical Environment," presented at International Computer Music Conference, 1996.

[151] D. J. Schiano, S. M. Ehrlich, K. Rahardja, and K. Sheridan, "Face to Interface: Facial Affect in Human and Machine," presented at CHI, 2000.

[152] E. Schierer, "Tempo and Beat Analysis of Acoustic Musical Signals," *Journal of the Acoustical Society of America* vol. 103, pp. 588-601, 1998.

[153] A. W. Schloss, "Using Contemporary Technology in Live Performance: The Dilemma of the Performer," *Journal of New Music Research*, vol. 32, pp. 239-242, 2003.

[154] J. Schnell, M. Rovan, and M. Wanderley, "Escher - Modeling and Performing Composed Instruments in Real-Time," presented at IEEE International Conference on System, Man and Cybernetics, San Diego, USA, 1998.

[155] D. Schwarz, "A System of Data-Driven Concatenative Sound Synthesis," presented at International Conference on Digital Audio Effects, 2000.

[156] C. Shanhabi, R. Zimmermann, K. Fu, and S. D. Yao, "Yima: A Second Generation Continuous Media Server," in *IEEE Computer Magazine*, pp. 56-64, 2002.

[157] S. Sharma, *Comparative Study of Evolution of Music in India & the West*. New Delhi, India: Pratibha Prakashan, 1997.

[158] L. C. D. Silva, T. Miyasato, and R. Nakatsu, "Facial Emotion Recognition Using Multimodal Information," presented at FG, 2000.

[159] E. Singer, J. Feddersen, C. Redmon, and B. Bowen, "LEMUR's Musical Robots," presented at International Conference on New Interfaces for Musical Expression, Hamamatsu, Japan, 2004.

[160] E. Singer, K. Larke, and D. Bianciardi, "LEMUR GuitarBot: MIDI Robotic String Instrument," presented at International Conference on New Interfaces for Musical Expression, Montreal, Canada, 2003.

[161] J. O. Smith, "Digital Waveguide Modeling of Musical Instruments." Palo Alto, CA, .002.

[162] J. Solis, M. Bergamasco, S. Isoda, K. Chida, and A. Takanishi, "Learning to Play the Flute with an Anthropomorphic Robot," presented at International Computer Music Conference, Miami, Florida, 2004.

[163] J. Stam and e. Fiume, "Turbulent Wind Fields for Gaseous Phenomena," presented at SIGGRAPH, 1993.

[164] K. Steiglitz, *A Digital Signal Processing Primer with Applications to Digital Audio and Computer Music*. Menlo Park, CA: Addison-Wesley Publishing Company, 1996.

[165] J. Stelkens, "peerSynth: a P2P multi-user software synthesizer with new techniques for integrating latency in real-time collaboration," presented at International Computer Music Conference, Singapore, 2003.

[166] D. O. Sullivan and T. Igoe, *Physical Computing*. Boston: Thomson Course Technology, 2004.

[167] A. Takanishi and M. Maeda, "Development of Anthropomorphic Flutist Robot WF-3RIV," presented at International Computer Music Conference, Michigan, USA, 1998.

[168] Y. Takegawa, T. Terada, and S. Nishio, "Design and Implementation of a Real-Time Fingering Detection System for Piano Performance," presented at Proceedings of the International Computer Music Conference, New Orleans, USA, 2006.

[169] A. Tanaka, "Interfacing Material Space and Immaterial Space: Network Music Projects, 2000," *The Journal of the Institute of Artificial Intelligence of Chukyo University*, 2000.

[170] B. C. Till, M. S. Benning, and N. Livingston, "Wireless Inertial Sensor Package (WISP)," presented at International Conference on New Interfaces for Musical Expression, New York City, 2007.

[171] A. R. Tindale, A. Kapur, G. Tzanetakis, and I. Fujinaga, "Retrieval of Percussion Gestures Using Timbre Classification Techniques," presented at International Conference on Music Information Retrieval, Barcelona, Spain, 2004.

[172] A. R. Tindale, A. Kapur, G. Tzanetakis, and W. A. Schloss, "Indirect Acquisition or Percussion Gestures Using Timbre Recognition," presented at Conference on Interdisciplinary Musicology, Montreal, Canada, 2005.

[173] T. Tolonen and M. M. Karjalainen, "A Computationally Efficient Multipitch Analysis Model," *IEEE Transactions on Speech and Audio Processing*, vol. 8, pp. 708-716, 2000.

[174] Trimpin, *Portfolio*. Seattle, Washington.

[175] Trimpin, *SoundSculptures: Five Examples*. Munich MGM MediaGruppe Munchen, 2000.

[176] D. Trueman and P. R. Cook, "BoSSA: The Deconstructed Violin Reconstructed," presented at Proceedings of the International Computer Music Conference, Beijing, China, 1999.

[177] G. Tzanetakis and P. R. Cook, "*Marsyas*: a Framework for Audio Analysis," *Organized Sound*, vol. 4, 2000.

[178] G. Tzanetakis and P. R. Cook, "Musical Genre Classification of Audio Signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, 2002.

[179] G. Tzanetakis, G. Essl, and P. R. Cook, "Audio Analysis using the Discrete Wavelet Transform," presented at Conference in Music Theory Applications, 2001.

[180] G. Tzanetakis, A. Kapur, and R. I. McWalter, "Subband-based Drum Transcription for Audio Signals," presented at IEEE International Workshop on Multimedia Signal Processing, Shanghai, China, 2005.

[181] B. Verplank, C. Sapp, and M. Mathews, "A Course on Controllers," presented at International Conference on New Interfaces for Musical Expression, Seattle, WA, 2001.

[182] D. Ververidis, C. Kotropoulos, and I. Pitas, "Automatic Emotional Speech Classification," presented at IEEE International Conference on Audio, Speach and Signal Processing, Montreal, Canada, 2004.

[183] B. Vines, M. M. Wanderley, C. Krumhansl, R. Nuzzo, and D. Levitin, "Performance Gestures of Musicians: What Structural and Emotional Information do they Convey?," presented at Gesture-Based Communication in Human Computer Interactiion - 5th International Gesture Workshop, Genova, Italy, 2003.

[184] R. A. Vir, *Learn to Play on Sitar*. New Delhi: Punjab Publications, 1998.

[185] B. Wait, *Mirror-6 MIDI Guitar Controller Owner's Manual*. Oakland, California: Zeta Music Systems, Inc. , 1989.

[186] H. G. Wallbott, "Bodily Expression of Emotion," *European Journal of Social Psychology*, vol. 28, pp. 879-896, 1998.

[187] M. M. Wanderley, "Non-Obvious Performer Gestures in Instrumental Music," presented at Gestrure-Based Communication in Human-Computer Interaction, 1999.

[188] M. M. Wanderley, "Quantitative Analysis of Non-obvious Performer Gestures," presented at Gesture and Sign Language in Human-Computer Interaction, 2002.

[189] G. Wang and P. R. Cook, "*ChucK*: A Concurrent, On-the-fly Audio Programming Language," presented at International Computer Music Conference, Singapore, 2003.

[190] G. Wang, A. Misra, A. Kapur, and P. R. Cook, "Yeah, *CHUCK* IT => Dynamic, Controllable Interface Mapping," presented at International Conference on New Interfaces for Musical Expression Vancouver, Canada, 2005.

[191] G. Weinberg, "Interconnected Musical Networks: Bringing Expression and Thoughtfulness to Collaborative Music Making," in *Media Lab*, vol. Ph.D. Cambridge, MA: MIT, 2002.

[192] G. Weinberg, S. Driscoll, and M. Parry, "Haile - A Preceptual Robotic Percussionist," presented at International Computer Music Conference, Barcelona, Spain, 2005.

[193] G. Weinberg, S. Driscoll, and T. Thatcher, "Jam'aa - A Middle Eastern Percussion Ensemble for Human and Robotic Players," presented at International Computer Music Conference, New Orleans, 2006.

[194] G. Wienberg, R. Aimi, and K. Jennings, "The Beatbug Network -- A Rhythmic System for Interdependent Group Collaboration," presented at Proceedings of the International Conference on New Interfaces for Musical Expression (NIME), Dublin, 2002.

[195] M. M. Williamson, "Robot Arm Control Exploring Natural Dynamics," vol. Ph.D. . Boston: Massachusetts Institute for Technology, 1999.

[196] R. S. Wilson, M. Gurevich, B. Verplank, and P. Stang, "Microcontrollers in Music Education - Reflections on our Switch to the Atmel AVR Platform," presented at International Conference on New Interfaces for Musical Expression, Montreal, Canada, 2003.

[197] T. Winkler, "Making Motion Musical: Gesture Mapping Strategies for Interactive Computer Music," presented at International Computer Music Conference, San Francisco, CA, 1995.

[198] A. Woolard, *Manual*. Tustin, CA: VICON Motion Systems, 1999.

[199] M. Wright, A. Freed, and A. Momeni, "OpenSound Control: State of the Art 2003," presented at International Conference on New Interfaces for Musical Expression, Montreal, Canada, 2003.

[200] M. Wright and D. Wessel, "An Improvisation Environment for Generating Rhythmic Structures Based on North Indian Tal Patterns," 1998.

[201] J. T. Wu, S. Tamura, H. Mitsumoto, H. Kawai, K. Kurosu, and K. Okazaki, "Neural Network Vowel-Recognition Jointly Using Voice Features and Mouth Shape Image," *Pattern Recognition*, vol. 24, pp. 921-927, 1991.

[202] A. Xu, W. Woszczyk, S. Z., B. Pennycook, R. Rowe, P. Galanter, J. Bary, G. Martin, J. Corey, and J. Cooperstock, "Real Time Streaming of Multi-channel Audio Data through the Internet," *Journal of the Audio Engineering Society*, 2000.

[203] J. Yin, T. Sim, Y. Wang, and A. Shenoy, "Music Transcription using an Instument Model," presented at Proceedings of International Conference on Audio Signal Processing 2005.

[204] Y. Yoshitomi, S. Kim, T. Kawano, and T. Kitazoe, "Effect of Sensor Fusion for Recognition of Emotional States using Voice, Face Image, and Thermal Image of Face," presented at ROMAN, 2000.

[205] D. Young, "The Hyperbow Controller: Real-Time Dynamics Measurement of a Violin Performance," presented at International Conference on New Interfaces for Musical Expression, Dublin, Ireland, 2002.

[206] D. Young and I. Fujinaga, "Aobachi: A New Interface for Japanese Drumming," presented at International Conference on New Interfaces for Musical Expression, Hamamatsu, Japan, 2004.

[207] A. Zils and F. Pachet, "Musical Mosaicing " presented at International Conference on Digital Audio Effects, 2001.

[208] U. Zolzer, *DAFX: Digital Audio Effects*. England: John Wiley and Sons, Ltd. , 2002.