Automation, Collaboration, & E-Services

Gil Weinberg
Mason Bretan
Guy Hoffman
Scott Driscoll

Robotic Musicianship

Embodied Artificial Creativity and Mechatronic Musical Expression

Springer

Automation, Collaboration, & E-Services

Volume 8

Series Editor

Shimon Y. Nof, PRISM Center, Grissom, Purdue University,
West Lafayette Indiana, IN, USA

The Automation, Collaboration, & E-Services series (ACES) publishes new developments and advances in the fields of Automation, collaboration and e-services; rapidly and informally but with a high quality. It captures the scientific and engineering theories and techniques addressing challenges of the megatrends of automation, and collaboration. These trends, defining the scope of the ACES Series, are evident with wireless communication, Internetworking, multi-agent systems, sensor networks, cyber-physical collaborative systems, interactive-collaborative devices, and social robotics—all enabled by collaborative e-Services. Within the scope of the series are monographs, lecture notes, selected contributions from specialized conferences and workshops.

More information about this series at http://www.springer.com/series/8393

Gil Weinberg · Mason Bretan ·
Guy Hoffman · Scott Driscoll

Robotic Musicianship

Embodied Artificial Creativity and Mechatronic Musical Expression

Springer

Gil Weinberg
Georgia Institute of Technology
Atlanta, GA, USA

Guy Hoffman
Cornell University
Ithaca, NY, USA

Mason Bretan
Novato, CA, USA

Scott Driscoll
Atlanta, GA, USA

ISSN 2193-472X ISSN 2193-4738 (electronic)
Automation, Collaboration, & E-Services
ISBN 978-3-030-38929-1 ISBN 978-3-030-38930-7 (eBook)
https://doi.org/10.1007/978-3-030-38930-7

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To all of our collaborators who made this work possible.

Foreword

Robotic Musicianship does, as the authors point out, sound like a contradiction in terms. "Robotic" is one of the most dismissive descriptions imaginable of a human musician's playing style. A "robotic" performance is one that is considered inexpressive, that incorporates none of the microinflections of loudness, timbre, and pitch that make a performance, as it is said, "come to life." Our human idioms give away the game here as to what our expectations for music performance are. Beyond that, the way we talk about them is not very respectful toward robots.

The fact of the matter is that robots, for the most part, do what we tell them to do. Or, as is more commonly the case in contemporary research, they do what we have helped them learn how to do. If they do not perform expressively, it is because we have not understood musical expression sufficiently to make its real-time implementation computationally tractable. Or, that we have not collected and represented data in such a way that we can provide meaningful targets from which a computer (the brains of a robot, after all) could learn expression.

There could be no better team to blast through the facile misconceptions attaching to *Robotic Musicianship* than the authors of the present volume. Gil Weinberg, Guy Hoffman, Mason Bretan, Scott Driscol and their group at the Georgia Institute of Technology have been the recognized leaders in building music robots for decades now. Their research program has been squarely directed at designing, implementing, and demonstrating in high-profile venues the possibilities of robotic musicianship, with results that are undeniably compelling.

The visual component of their work is critical: others have worked on the design of interactive music systems that react in a convincing way to a real-time musical context (this is hard enough), but pure through an audio output. We know, however, that much of human musical interaction is guided by musicians watching one another, particularly during improvisation. Moreover, seeing a robot perform the output of these systems shows an audience exactly what the computational system is adding to the whole—something that can be difficult to parse out from audio alone, particularly in the context of ensemble playing.

Chapters in this text demonstrate the human-centric and musically motivated strategies of their agenda: they aim to help the robots "listen like a human", but "play like a machine". To "listen like a human" requires a sophisticated understanding of music cognition, and the traditions and research that have developed in that field over a period of decades. To "play like a machine" requires, indeed, that a machine be built. And then that the machine be able to execute the (expressively nuanced) musical gestures that will provide a meaningful counterpart to live human musicians.

That is a tall order of business. We see emulation of human cognition, musical insights sufficient to compose compelling and appropriate additions to an ongoing musical texture, and the mechanical engineering necessary to complete all of this in real time, in the context of a live public performance. Certainly, this is a challenge on the order of, say, building a self-driving car, or a robot that could navigate a busy sidewalk.

And that challenge is met here with acumen, rigor, insight, and strategic design. This is an engineering text. The authors, scientists, and engineers all, present this consummate feat of design and implementation here in a way that makes their advances available and reproducible by those following in their considerable wake. Much of what they present here has been developed with the benefit of input from human subjects, tests that provide important insights for the psychology of music. Signal processing, genetic algorithms, machine learning, robotics, algorithmic composition, wearables, music cognition and artificial intelligence are all involved, and all presented in a way that is practicable for the expert yet approachable by the layman.

Robotic performance is not only a matter of emulating what a human might do—indeed, a machine can perform material that would be quite impossible for a human musician. The authors demonstrate how their creations can not only render music expressively, but can expressively perform passages that could not be produced by their human counterparts. This is but one of the many ways in which their work is an extension and elaboration and contribution to human musical expression, rather than an effort of substitution. Robotic musicianship is not about replacing human musicians, but a celebration and riotous new expansion of the possibilities of music-making. These musician/engineer authors have spent decades building at the cutting edge of interactive music, performed by robots, in real time, before a live audience. What they have built is no gee-whiz science-fair project, but substantial engineering and real contribution to the possibilities of music-making. We will all be richer for it. Seeing (and hearing) is believing—just watch, and listen to them.

New York City, NY Robert Rowe

Preface

We arrived at Robotic Musicianship from different directions. Gil is an improvising musician, interested in bringing his algorithmic improvisation ideas into the acoustic world; Scott is a mechanical engineer looking to bring his technical and musical interests together; Guy is a Human Robot Interaction (HRI) researcher who finds music a rich domain to explore social human-robot interaction; and Mason is a machine learning scientist and a musician whose research combines both disciplines. This diversity in backgrounds and interests exemplifies the wide interdisciplinary nature of the field of Robotic Musicianship. Since 2005 we have brought together this wide range of perspectives, collaborating among ourselves and others at the Georgia Tech Center for Music Technology, to develop the robotic musicians described in this book.

The first robot, Haile, was the product of a collaboration between Gil, a new assistant professor of music technology at Georgia Tech, and Scott, a mechanical engineering master's student. The unfunded project led to a hand-built wooden robot, probably one of the few wooden robots out there, with exposed electronics and a rugged look. A video posted online of Scott improvising with Haile was noticed by an NSF director, who emailed to request a proposal for developing a more advanced robot that would help public perception of robots through music. The proposal that ensued led to the development of Shimon, a marimba playing robot, which helped facilitate new research directions into melodic and harmonic robotic improvisation.

A coincidental meeting between Gil and Guy at a conference led to the idea of a social head for Shimon. Guy then joined the Robotic Musicianship group at GTCMT as a postdoctoral researcher to develop gestural HRI applications for Shimon. After Guy was invited to present his work at a conference in Germany, a decision was made to create a small and portable version of Shimon for future conferences. The new personal robot, called Shimi (aka Travis), was developed in a record speed of six weeks to be ready for future demonstrations. It was then selected as one of the few finalists at the prestigious TechCrunch Disrupt startup competition, although the commercialization effort that followed was unsuccessful.

When Mason joined the Robotic Musicianship group as a Masters and later a Ph. D. student, he helped bring Shimon and Shimi to the next level by developing machine learning based improvisation and emotional gestures for the robots. Mason

was also the main researcher behind the development of our wearable robotic musicianship platforms, following a request by amputee drummer Jason Barnes. After losing his arm in an electrical accident, Jason asked us to build a musical robotic prosthetic that would allow him to play drums again. Using EMG, ultrasound, and machine learning, we developed different prosthetic arms for Jason that not only brought back his lost abilities but also allowed him to play like no other human can. Jason is now featured as the "Fastest Drummer (using Prosthetic)" in the 2020 Guinness World Book of Records.

In addition to pursuing our respective research interests, we have also been drawn to Robotic Musicianship by the exciting performance opportunities the field offers. With our robotic musicianship platforms we have traveled across the world, performing dozens of concerts across four continents. Some of our performances and presentations were commissioned by prestigious venues such as The Kennedy Center for the Arts, Aspen Ideas Festival and the World Economic Forum in Davos. Others took place in festivals in cities such as St. Petersburg, Brisbane, Istanbul, and Shanghai. We also performed in small venues in front of small audiences such as in an art gallery in Paris, a private event in Berlin and a nightclub in Atlanta. A concert actually led to this book, when we met our Editor, Shimon Nof, who asked us to write this book after a concert he attended at Purdue University.

Traveling and presenting newly developed prototypes that have just come out of the lab can also lead to memorable, yet anxiety ensuing experiences. Today we can look back fondly at stressful experiences such as a performance in Paris, where Haile stopped playing due to a broken wire, leading Gil and Scott to continue improvising as a human duo while calculating escape routes out of the venue; or the time when a few hours before a live presentation on The TODAY show, Shimon stopped working, sending Gil to the streets of New York in a frenzied search for a replacement USB hub, with Mason frantically debugging Shimon in the studio (the live demonstration went well at the end). Guy's nervous performance in at an Atlanta Concert Venue started before the audiences entered the hall. A different performance in a prestigious festival in Munich ended with Guy not being able to return to Georgia Tech for 3 months due to visa issues.

These concerts and performances are an important part of our Robotic Musicianship work and we encourage readers to look online for videos that could help demonstrate the musical outcome of the work described in the book. We hope readers will find both the technical and artistic aspects of our work as interesting and inspiring as we do.

Sincerely,

Atlanta, GA, USA	Gil Weinberg
Atlanta, GA, USA	Scott Driscoll
Ithaca, NY, USA	Guy Hoffman
Novato, CA, USA	Mason Bretan

Acknowledgements We would like to thank our collaborators: Roberto Aimi, Ryan Nikoloaidis, Mark Godfrey, Alex Rae, Brian Blosser, Travis Thatcher, Mitch Parry, Sisi Sun, Trishul Malikarjuna, Annie Zhang, Jay Clark, Marcelo Cicconnet, Philip Mullins, Deepak Gopinath, Roozbeh Khodambashi, Tyler White, Lea Ikkache, Zach Kondak, and Richard Savery.

Contents

Chapter 1
Introduction

1.1 Abstract

The purpose of this book is to introduce the concepts that define robotic musicianship and the underlying technology that supports them. Research in robotic musicianship focuses on the construction of machines that can produce sound, analyze and generate musical input, and interact with humans in a musically meaningful manner. In this chapter, we first discuss motivations for pursuing robotic musicianship, and review past and ongoing research efforts. In subsequent chapters we dive more deeply into these topics and describe specific designs, algorithms, and evaluations that we have developed and applied to our own robotic musicians.

1.2 Why Robotic Musicianship

The term '*robotic musicianship*' may seem like a contradiction-in-terms. The word 'robotic' often bears negative connotations in art, typically referring to a lack of expression. Conversely, 'musicianship' is usually used to describe varying levels of an individual's ability to apply musical skills and knowledge in order to convey artistry, creativity, and expression beyond the facets of merely reading notes from a score. Together these two words describe a growing field of research that merges science, art, and technology.

There are two primary research areas that constitute the field of Robotic Musicianship: (1) *Musical Mechatronics* —the study and construction of physical devices that generate sound through mechanical means [1] (2) *Machine musicianship*—the development of algorithms and cognitive models of music perception, composition, performance, and theory [2]. Robotic musicianship brings these two disciplines together. Researchers within this space design musical robots that have the underlying musical intelligence to support both expressive solo performance and interaction with human musicians [3].

G. Weinberg et al., *Robotic Musicianship*, Automation, Collaboration,
& E-Services 8, https://doi.org/10.1007/978-3-030-38930-7_1

Research in robotic musicianship has seen growing interest from scientists, musicians, engineers and artists in recent years. This can be attributed to the interdisciplinary nature of the field that is amenable to a wide range of scientific and artistic endeavors. First, music provides a common platform for research because of its cross-cultural elements such as rhythm and pitch. Second, robotic musicianship requires the integration of multiple disciplines including engineering, computation, music, and psychology, bringing together a wide range of scholars to address the challenges involved in making a machine artistically expressive. Third, different aspects of research in robotic musicianship, such as timing, anticipation, expression, mechanical dexterity, and social interaction have the potential to lead to broader impacts in other fields from cognitive science, human-robot interaction (HRI) to mechanics. Moreover, music is one of the most time-sensitive mediums where events separated by mere milliseconds are perceptually noticeable. Developing HRI algorithms for anticipation or synchronization in a musical context can, therefore, have impact on other HRI scenarios, where accurate timing is required. By building and designing robotic musicians, scholars can also better understand the sophisticated interactions between the cognitive and physical processes in human music making. By studying and reconstructing sound production mechanisms, robotic musicians can help shed light on the role of embodiment in music making, exploring how music perception and cognition interact with human anatomy.

Other motivations to develop robotic musicians are cultural and artistic in nature. Robert Rowe, a pioneer of the field of interactive music systems, writes, "if computers interact with people in a musically meaningful way, that experience will bolster and extend the musicianship already fostered by traditional forms of music education…and expand human musical culture" [2]. Similarly, the goal of robotic musicianship is to supplement human creativity and enrich the musical experience for humans, rather than imitate or replace it. By bringing together perceptual and generative computation with physical sound generators we can create systems capable of (1) rich, acoustic sound production, (2) intuitive, physics-based visual cues from sound producing movements, and (3) expressive physical behaviors through sound accompanying body movements. Moreover, there is artistic potential in the non-human characteristics of music-generating machines including increased precision, freedom of physical design, and the ability to perform humanly impossible fast computations. Discovering how best to leverage these characteristics to benefit music performance, composition, consumption, and education is a worthy effort as music is an essential component of human society and culture.

As social robots begin to penetrate public spaces, music can play an important role in aiding the process of integrating personal robotics into everyday life. Music is shared, understood, and appreciated by billions of people across many different cultures. Such ubiquity can be leveraged to yield a sense of familiarity between robots and people by using music to encourage and facilitate human-robotic interaction. As more people begin to integrate robots into their everyday lives, music can help establish trust and confidence in this new technology.

But the recent interest in robotic musicianship has not come without popular criticism and misconceptions, some of which focus on the concern that the ultimate

outcome of this research might be to replace human musicians.[1,2,3] Though much of this concern may stem from the natural awe and unease accompanied by the introduction of new technology, it is important to reiterate both our own and Rowe's sentiment that by placing humans alongside artistically capable machines we are establishing an environment in which human creativity can grow, thus, enriching human musical culture rather than replacing it.

A related criticism stems from the belief that the creation of music—one of the most unique, expressive, and creative human activities—could never be replicated by automated "soul-less" machines, and that the product of such efforts will never lead to "good music".[4] We address this potential criticism by emphasizing its great potential for enhancing human musical creativity and expression. For example, although in some cases the need for live musicians diminished when new technology was introduced (such as the introduction of talking films, or the MIDI protocol), technology has also introduced new opportunities in sound and music making, such as with sound design and recording arts. Similarly, disk jockeys introduced new methods for expressive musical performance using turn tables and pre-recorded sounds. Further, robotic musicianship has the potential to advance the art form of music by creating novel musical experiences that would encourage humans to create, perform, and think about music in new ways. From utilizing compositional and improvisational algorithms that humans cannot process in a timely manner (using processes such as fractals or cellular automata) to exploring physical sound production capabilities that humans do not possess (from speed to timbre control), robotic musicians bear the promise of creating music that humans could never create by themselves and inspire humans to explore new musical experiences, invent new genres, expand virtuosity, and bring musical expression and creativity to uncharted domains.

The rest of this chapter will review past and ongoing research in robotic musicianship, starting with exploring different methods for sound production, moving to machine listening, and then to generative algorithms that enable the robot to complete different musical tasks such as improvisation or accompaniment. The chapter concludes with the design of social interactions and visual cues based on embodiment, anthropomorphism, and expressive gestures.

1.3 Sound Production and Design—Survey

Acoustic musical instruments produce sound in a variety of ways. For example, resonating chambers can be excited by bowing strings, striking a covering membrane, and blowing air. When designing autonomous sound generators it is important to

[1] http://www.asylum.com/2011/01/19/robot-rock-bands-expressive-machines-musical-instruments/.

[2] http://nymag.com/news/intelligencer/robot-jobs-2013-6/.

[3] http://www.wired.com/2012/12/ff-robots-are-already-replacing-us/2/.

[4] http://www.smithsonianmag.com/history/musicians-wage-war-against-evil-robots-92702721/?no-ist.

consider the acoustic properties of the resonator. Robots that play traditional acoustic instruments can be classified by those which emulate the human body and natural mechanics applied to playing an instrument and those which augment acoustic instruments to produce sound and create a physical affordance that is not possible in the instrument's archetypal state, such as in the case of prepared piano [1].

1.3.1 Traditional Instruments

Traditional acoustic instruments are designed to be played by humans in terms of their size, ergonomic design and input affordances that accommodate the human body. Robots, on the other hand, are not constrained to human form and can, thus, be designed to play musical instruments in a different manner than humans do. While emulating people can be mechanically challenging, some musical robot designers find it helpful to reference and build on human physiology and the various physical playing techniques employed by musicians. Other designers and researchers attempt to break free from the constraints of human form and explore aspects of instrument playability related to speed, localization, and dynamic variations that are not attainable by humans. In this section we will explore examples of different design methodology for different instruments and discuss their inherent advantages and disadvantages.

1.3.1.1 Percussive Instruments

The mechanics involved in human percussion playing are complex and require controlled movement of and coordination between the wrist, elbow, and fingers, and with some instruments, the legs and feet. In the case of hand drumming, subtle changes in the tightness of a hand grip on the drumstick, rate at which the wrist moves up and down, and distance between the stick and drum head provide drummers with expressive control over their performance. Human drummers use different kinds of strokes to achieve a variety of timbres and playing speeds. For example, for creating two distinct hits separated by a large temporal interval, drummers usually use a single-stroke in which the wrist moves up and down for each strike. For a double hit separated by a very small temporal interval, drummers tend to use a multiple-bounce stroke in which the wrist moves up and down once to generate the first strike, while subsequent strikes are achieved by adjusting the fingers' grip on the drumstick to generate a bounce. Multiple motors, gears, solenoids, and a sophisticated control mechanism are required to imitate such functionality.

Several robotic systems have been designed to imitate velocity curves of single-stroke wrist and arm movements. The *Cog* robot from MIT uses oscillators for smoother rhythmic arm control [4, 5]. Another approach to smooth motion control is to use hydraulics as done by Mitsuo Kawato in his humanoid drummer [6]. Some robotic designs are inspired by the general physical shape of humans rather than

specific human muscle and joint control. The humanoid keyboard playing *WABOT* has two hands for pressing keys and feet for pushing pedals [7]. More recently, Yoky Matsuoka developed an anatomically correct piano-playing hand [8].

Designers of robotic percussion instruments can also achieve their sonic and expressive goals by using simpler engineering methods than those that are required to replicate human physical control. A common approach for such percussive robotic devices is to use solenoids that strike percussion instrument with sticks or other actuators. Erik Singer's LEMUR bots [9] and portions of *Orchestrion* (designed for Pat Metheny[5]) use such an approach. Singer's projects use non-anthropomorphic designs in which individual solenoids are used to strike individual drums.

In addition to a simplified actuation design, such non-humanoid robots can produce musical results that are not humanly possible. For example, in Orchestrion each note on a marimba can be played faster than human speed because a solenoid is placed on each one of the keys. This approach allows for increased flexibility in choosing different types of push and pull solenoids that can be used to achieve high speed and dynamic variability [10]. Unlike human percussionists, the single strokes generated by solenoids or motors can move at such great speeds so that it is not necessary to model human double-stroke functionality.

While solenoid based designs are mechanically simple and capable of a wide range of expression, their small range of movement cannot provide useful visual cues to humans. Humanoid robots, on the other hand, can provide human co-players and audience members with a framework that they are more familiar with, thus, allowing them to better understand the relationship between sound generation and the accompanying robotic movements and gestures. Additionally, the visual cues provided by the larger human-size motion and gestural movements allow the human musicians to anticipate the robot's actions and improve synchronization. Some robots combine functions of both design methods in an attempt to simultaneously gain the benefits of the inhuman sound producing capabilities and important visual cues. The marimba playing robot, *Shimon* (Fig. 1.1), has four arms each with two solenoid activators for striking the keys. The movements of the arms combined with the eight fast moving solenoids provides advantages from both design methods [11]. Additional percussive systems are listed in Table 1.1.

1.3.1.2 Stringed Instruments

Similarly to percussive robots, many string playing robots focus on achieving simple, yet robust excitation and pitch manipulation rather than mimetic control and anthropomorphism. Sergi Jorda's *Afasia* system [17] is one such example, where solenoids are used to pluck strings, while pitch is controlled by either push solenoids placed along the bridge or motors that move the bridge itself. Singer's *GuitarBot*, on the other hand, uses a wheel embedded with guitar picks that can rotate at variable

[5] http://www.theorchestrionproject.com/.

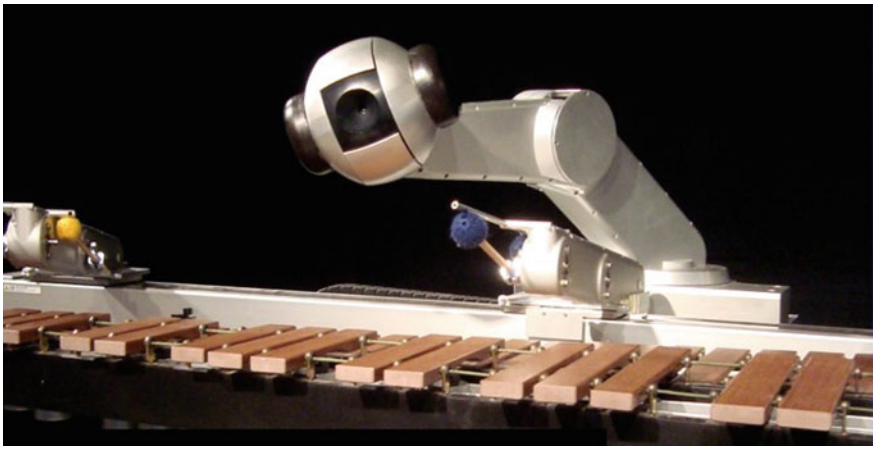


Fig. 1.1 *Shimon* of Georgia Tech Center for music technology

Table 1.1 Percussive robots

Developer	Robotic musician
Logo's Foundation	**Vibi**—An automated vibraphone with individual dampers for each bar
	Troms—Seven single drums each outfitted with a set of strikers positioned at various locations on the drumhead
	Vacca—Series of cowbells equipped with different style hammers driven by solenoids [12, 13]
Expressive Machines Musical Instruments (EMMI)	**MADI**—*Multi-mallet automatic drumming instrument* consists of 15 solenoid strikers positioned around a single snare drum allowing the system to take advantage of different timbres available at specific locations on the drum [14]
Trimpin	**Automated idiophones**—Pitched percussion outfitted with individual solenoids that drive strikers that can be interchanged and adapted to achieve different timbres [15]
Karmetik	**Notomotion**—A drum equipped with several rotary and pull solenoids with a rotating mounting structure allowing for a wide range of timbres
	Raina—A rainstick attached to a drive train that slowly rotates [16]
Georgia Tech Center For Music Technology (GTCMT)	**Haile**—Anthropomorphized robotic percussionist equipped with a linear motor and solenoid for striking a drum [3]

Table 1.2 String playing robots

Developer	Robotic musician
Logo's Foundation	**Hurdy**—An automated bowed bass instrument that uses several motors to bow the strings
	Aeio—An automated cello in which the strings are excited by using two electromagnets driven by a two phase signal on opposite sides of the string [12, 13]
EMMI	**PAM**—PAM: Poly-tangent Automatic multi-Monochord [14]
Trimpin	**Krantkontrol**—An installation of 12 guitar-like instruments with a plucking mechanism and solenoids for fretting
	"If VI was IX"—An installation containing hundreds of guitars with pitch and plucking acuators [15]
Baginsky	**Aglaopheme**—A slide guitar with solenoids for plucking and fretting and a motor used to alter the position of the bridge [18]
Victoria University of Wellington and New Zealand School of Music	**MechBass**—A four string modular robotic bass guitar player. Each string is plucked using a stepper motor and a solenoid attached to a carriage moves along the string via a belt drive for fretting [19]

speeds to pluck strings and relies on a moving bridge assembly that rides on pulleys driven by a DC motor.

A different, more anthropomorphic approach is taken by Chadefaux et al., which utilizes motion capture of human harp playing to design of a robotic harp playing finger. Silicone in various shape and stiffness is placed on the robotic fingertips in order to replicate different vibrational characteristics [20]. An anthropomorphic violin bowing arm was developed by Shibuya et al. The shoulder, elbow, and wrist were modeled to create expressive playing by controlling the bow's pressure, position, and rate of movement [21, 21, 22]. Similarly, Toyota's robotic violin player replicates human form and motion.[6] See additional string playing robots in Table 1.2.

1.3.1.3 Wind Instruments

One of the main challenges for wind-instrument playing robots is to produce the necessary energy fluctuations that would make the instrument resonate at desired frequencies and volumes. Some designers address this challenge using non-humanoid designs and function. Roger Dannenberg's *McBlare* is a non-humanoid robotic bag-

[6] http://www.toyota-global.com/innovation/partner_robot/.

Table 1.3 Wind playing robots

Developer	Robotic musician
Logo's Foundation	**Ob**—An automated oboe in which the reed is replaced with a acoustic impedance converter that models a real reed in a human mouth cavity [12, 13]
National ICT Australia	**Robot Clarinet**—A robotic clarinet player that controls for blowing pressure, lip force, and lip position to affect pitch, sound level, and spectrum [23]

pipe player that is based on an air compressor that provides air and controls the power to robotic "fingers" which open and close the sound holes. While the speed of McBlare's control mechanisms surpasses that of expert bagpipe performers it cannot equal the subtle human skill of adapting finger movements to the characteristics of the reed [24]. Another example of a non-anthropomorphic robotic wind instrument is the Logo's Foundation's *Autosax* [12], which uses "compression drivers and acoustic impedance converters that feed the drive signal to the instrument via a capillary."

A different approach for wind instrument playing robots focuses on replicating and understanding human physiology and anatomy. Ferrand and Vergez designed a blowing machine that explores the nonlinear dynamics natural human airflow by modeling the human mouth [25]. Their system's performance was affected by the servo valve's friction, cut-off frequency, and mounting stiffness of the artificial lips. The Takanishi Group at Waseda University developed a humanoid flute-playing robot as well as a sax-playing robot [26, 27]. Their systems include multiple degree-of-freedom (DOF) actuators for controlling finger dexterity, lip and tongue control, and lung control using a valve controlled air pump. Additional wind playing robots are listed in Table 1.3.

1.3.2 Augmented and Novel Instruments

While the examples described in previous sections demonstrate how musical robots are designed to play traditional instruments based on how humans play those instruments, some robotic researchers have been exploring new ways to play traditional instruments. Such methods can be useful for acquiring additional sonic variety and playability. Trimpin's *Contraption Instant Prepared Piano* is an example for such instrument augmentation. It enables mechanical bowing, plucking, and vibrating of piano strings to increase the harmonic and timbral range of the instrument. In many augmented instruments the human performer and robotic mechanisms share the control of the sound producing characteristics. This can lead to new musical experiences that combine the traits inherent to the interaction of two individual musicians and the interaction between performer and instrument. Richard Logan-Greene's systems,

Table 1.4 Augmented and novel instruments

Developer	Robotic musician
Logo's Foundation	**Dripper**—A rain machine which controls the precise size and frequency of each drip [12, 13]
Augmented Instruments Lab (Queen Mary University)	**Magnetic Resonator Piano**—An acoustic grand piano augmented with electromagnets inside the instrument to create vibrations in the strings [28] **Digital Bagpipe**—An electronic bagpipe chanter interface that is equipped with infrared sensing to record finger positions for later analysis [29]
Karmetik	**ESitar**—A hyperinstrument that uses sensors to measure human gesture during sitar performance and interface directly with a computer [30]

Brainstem Cello and *Actionclarinet*,[7] can provide such shared interactive control as the instruments are outfitted with sensors that respond to the human performer dampening the strings or manipulating pitch in real-time.

There are also examples of utterly new instruments in which automated mechanical functions are necessary to create a performance. Trimpin's "Conloninpurple" installation is made up of marimba-like bars coupled with aluminum resonating tubes and solenoid triggers. Each key is dispersed throughout a room to create a massive instrument capable of "3-dimensional" sound. Logan-Greene's *Submersible* is a mechanical instrument that raises and lowers pipes in and out of water to modify their pitch and timbre (Fig. 1.2) [31]. Additional augmented instruments are listed in Table 1.4.

1.4 Musical Intelligence

The ability to produce sound is only one important aspect of robotic musicianship. Designing cognitive models and algorithms that enable machines to behave like skilled musicians is another priority. Intelligent musical machines must embody the traits necessary for applications encompassing performance, interaction with other musicians, and improvisation. The ultimate goal of robotic musicianship is to create robots that can demonstrate musicality, expression, and artistry while stimulating innovation and newfound creativity in other interacting musicians. In order to achieve artistic sophistication, a robotic musician must have the ability to extract information relevant to the music or performance and apply this information to its underlying musical decision processes.

[7] http://zownts.com/.

Fig. 1.2 *Submersible* by Richard Logan-Greene

1.4.1 Sensing and Perception

1.4.1.1 Machine Listening

Providing machines with the ability to sense and reason about their environments is an important objective of robotics research. For robotic musicians, the emphasis on sensing and analysis of environmental input focuses on the auditory domain, although visual sensing is important for musicianship as well. Extracting meaningful information from an audio signal is referred to as *machine listening* [2]. There are numerous features, in different levels of complexity and perceptual relevance, that can describe the characteristics of a musical signal. Lower-level features typically describe physical characteristics of the signal while higher-level descriptors, though often more difficult to compute, tend to correlate more strongly with human musical perception, including elements such as mood, emotion, and genre. These higher-level descriptors are usually classified by using a combination of low-level (amplitude and spectral centroid) and mid-level (tempo, chord, and timbre) features (see Table 1.5).

Determining which attributes of the sound should be used for robotic perception relies on the desired functionality of the robot and its interaction with human musicians. For example, when a robotic musician designer intends for the system to interact with a drummer, audio analysis techniques that focus on rhythmic and timing qualities would be preferred over an analysis of pitch and harmony. In fact, for many of the drumming robots described above, audio processing algorithms have indeed focused on aspects related to timing. A first step in rhythmic analysis is to detect musical events such as individual notes. This process is referred to as *onset detection* and can be implemented using a variety of methods based on the derivative

Table 1.5 Relevant 'machine listening' terms

Loudness—A low level feature that correlates with the human perception of a signal's amplitude. Different methods of measurement exist for addressing perceptions at different frequencies
Mel Frequency Cepstral Coefficients (MFCCs)—A compact representation of the spectral envelope that is computed from the cepstral representation of a signal using a mel-frequency scale
Pitch—A perceptual feature of sound that correlates with frequency. However, the relationship is non-linear in that a perceived interval (such as an octave) at higher frequencies is separated by a larger number of hertz than at lower frequencies
Root Mean Square (RMS)—A low level feature that is used to describe the magnitude of an audio signal
Spectral Centroid—A low level feature describing the center of gravity within the frequency domain
Spectral Flux—A low level feature describing the change in spectral shape and computed by calculating the average difference between consecutive spectral frames. It is often used for onset detection because peaks arise at pitch changes and the beginning of new notes
Timbre—A characteristic of sound that is often described as a sound's *color* or *texture*. It allows us to differentiate between two sound sources that have equal pitch and loudness. Many definitions of timbre exist, but each has been criticized by researchers and there is currently no consensus on any one particular definition. It is easier to describe timbre as what it is not: pitch and loudness. Timbre is a multi-dimensional property meaning (unlike pitch and loudness) it cannot be ordered on a scale from low to high. Typically, audio features that describe the spectral shape of a signal (such as centroid, flux, and MFCCs) are used to describe the timbre of a sound
Tonality—A predominant system used in Western music that is determined by the relationships among a song's *key*, *chords*, and *notes*. It provides context to harmonic structure by describing the specific functions of individual notes and chords
Tonalness—A signal property describing the amount of *tonal* and *noisy* components in a signal. Signals that exhibit high periodicity are expected to be perceived as tonal and those with inherently low periodicity (such as white noise) are perceived as noisy

of energy with respect to time, spectral flux, and phase deviation [32]. Instruments which exhibit longer transients (such as cello or flute) and rich playing techniques (such as vibrato and glissandi) can pose challenges for these forms of detection. It is also a subject of debate whether the system should detect all physical onsets or all humanly perceptible onsets. Some psychoacoustically informed methods support the natural human perceptions in different portions of the frequency spectrum [33, 34].

An analysis of onsets over time can provide additional musically meaningful information related to note density, repetition, and beat. The notion of beat, which can correspond to the rate at which humans tap their feet to the music, is characteristic to many musical genres. Beat is often computed by finding the fundamental periodicity of perceived onsets using autocorrelation [35]. The *Haile* robot, for example, continuously estimates the beat of the input rhythms and uses the information to inform its response [11]. Barton's *HARMI* (Human and Robot Musical Improvisation) software

explores rhythmic texture by detecting multiple beat rates, not just those which are most salient to human listeners. This technique allows for more subtle expression by increasing rhythmic and temporal richness and allowing the robot to "feel" the beat in different ways. As a result the robot can detect "swing" in jazz music and the more rubato nature intrinsic to classical music [36].

Other analysis methods help robots to perceive the pitch and harmonic content of music. Pitch detection of a monophonic signal can been achieved with relatively robust and accurate results, while polyphonic pitch detection is more challenging [37]. Extracting the melody from a polyphonic source is often done by finding the most salient pitches and using musical heuristics (such as the tendency of melodic contours to be smooth) to estimate the most appropriate trajectory [38].

Detecting harmonic structure in the context of a chord (three or more notes played simultaneously) is an important aspect of robotic musicianship perception. Chord detection and labeling in tonal music is often performed by using "chroma" features, where entire spectra of the audio input is projected into 12 bins representing the 12 notes (or chroma) of a musical octave. The distribution of energy over the chroma is indicative of the chord that is being played. In Western tonal music, hidden Markov models (HMMs) are often used to address this challenge, as specific chord sequences tend to occur more often and applying these chord progression likelihoods can be beneficial for chord recognition and automatic composition [39, 40].

Pitch tracking and chord recognition are useful for robotic musicians when performing score following. The robot can listen to other musicians in an ensemble and attempt to synchronize with them by estimating their location within a piece [41, 42]. Otsuka et al. describe a system that switches between modules based on melodic and rhythmic synchronization depending on a reliability measure of the melody detection [43]. Other improvising robotic systems such as Shimon use the chords in the music as a basis for their note generative decisions.

As an alternative to utilizing audio analysis for machine listening, symbolic protocols such as Musical Instrument Digital Interface (MIDI) and Open Sound Control (OSC) can be used to simplify some of the challenges inherent to signal analysis. MIDI is sometimes preferred because it can greatly reduce inaccuracies of signal processing methods and is robust against environmental noise. A MIDI message can describe the pitch, duration, and velocity of a note as well as other parameters such as vibrato or panning. One caveat of MIDI is that it requires a digital interface and the instrumentalists lose the ability to produce acoustic sound which might be desired.

Open Sound Control (OSC) is another messaging protocol that enables added flexibility over MIDI by allowing any additional information to be shared outside of the standard pitch, duration, and velocity parameters [44]. Entire note sequences, high precision numerical values, or other higher level musical parameters can be communicated. OSC is particularly useful for sending information describing non-discrete values from other sensing methods describing motion, location, or physiological response. These other sensing methods and their relevance to musical interactions are described in the next section.

1.4.1.2 Machine Vision and Other Sensing Methods

Visual and physical cues can provide meaningful information during a musical interaction. A performer's movement, posture, and physical playing technique can indicate information related to timing, emotion, and even the amount of experience with the instrument. Incorporating additional sensors can improve the machine's perceptive capabilities, enabling it to better interpret the musical experience and perceptions of the interacting musicians.

Computer vision techniques have proven to be useful for human-robot musical interaction. Pan et al. program their humanoid marimba player to detect the head nods of an human musician [45]. The nods support a natural communication so that humans and robots can exchange roles during the performance. Similarly, Solis's anthropomorphic flutist robot uses computer vision to create instinctive interactions by detecting the presence of a musical partner. Once detected the robot starts to listen and evaluate its partner's playing [41]. Shimon used a Microsoft Kinect to follow a person's arm movements to control the dynamics and tempos of pre-programmed motifs. Shimon was also programmed to track infra-red (IR) lights mounted at the end of mallets allowing a percussionist to train the system with specific gestures and to use these gestures to cue different sections of a precomposed piece [46]. Mizumoto and Lim incorporated vision techniques to detect gestures to improve beat detection and synchronization [42, 47]. Their robotic theremin player used score following to synchronizes with an ensemble by combining auditory and visual cues. Similarly, the *MahaDeviBot* utilized a multimodal sensing technique to enhance tempo and beat detection.

Instead of using vision sensors, instrumentalists can use Inertia Measuring Units to measure acceleration and orientation of their limbs that provide information regarding their movement. Hochenbaum and Kapur combined onsets detected by the microphone and such wearable sensors to detect the tempo of a piece in real-time [48]. They also extended their multimodal sensing techniques to classify different types of drum strokes and extract other meaningful information from human performance [49]. The accelerometers were placed on human drummers' hands to automatically annotate and train a neural network to recognize the different drum strokes from audio data. Inter-onset-interverals between onsets of both the audio and accelerometer data were examined, showing greater consistency between the physical and auditory onsets for expert musicians. Such multimodal analyses which combine the physical and auditory spaces are becoming more standard in performance analysis. Mobile phones can provide another means of multimodal communication with a robot. Using the iPhone application, *Zoozbeat* [50], a human musician can provide symbolic notation to Shimon through a graphical user interface and accelerometer data describing his or her movement [51].

The ability to synchronize musical events between musicians often requires the system to know when an event will occur before it actually does. In music, these events are usually realized in the form of a musical score, however, in some interactions such as improvisation there is no predefined score, which complicates anticipation and synchronization among players. Cicconet et al. explored visual cues based antic-

ipation with a vision system that predicts the time in which drummers will play a specific drum based on of their movements [52]. Shimon was programmed to use these predictions to anticipate drummers' strikes in order to play simultaneously. The anticipation vision algorithm analyzes the upward and downward motion of the drummer's gesture by tracking an IR sensor attached to the end of a mallet. Using the mallet's velocity, acceleration, and location the system predicts when the drum stick will make contact with a particular drum.

1.4.2 Music Generation

In addition to computational perception, robotic musicians have explored a variety of approaches for music generation, both in software and hardware. Generative functions such as composition, improvisation, score interpretation and accompaniment can rely on statistical models, predefined rules, or abstract algorithms, as well as a variety of actuation techniques, as described below.

1.4.2.1 Algorithmic Composition

Autonomous composition systems are typically classified as being rule-based, data-driven, or a hybrid of data statistics and rules.

Rule based systems usually formulate music theory and structure into formal grammars and can creatively map the robot's perceptual modules to its sound generating modules. The Man and Machine Robot Orchestra at Logos has performed several compositions which employ a variety of such mappings [12] for musical expression. In the piece, *Horizon's for Three* by Kristof Lauwers and Moniek Darge, pitch is detected from a performer playing an electric violin and then directly mapped to notes generated by the machine. The movements of the performers are mapped to control the wind pressure of a robotic organ. In another piece, *Hyperfolly* by Yvan Vander Sanden, the human musician presses buttons that trigger responses from the machines. Similarly, Pat Metheny controls robotic instruments in his *Orchestrion* setup by playing specific notes or using other triggers to generate precomposed sections of his compositions [53].

Though using traditional AI techniques such as ANNs and Markov Chains may capture some of the compositional tendencies of humans, it is also common for composers and researchers to adapt a wide variety of algorithms to music generating functions. Often the behaviors of certain computational phenomena can yield interesting and unique musical results. Chaos, cellular automata, genetic, and swarm algorithms have all been re-purposed for musical contexts and a few specific to robotic musicianship performances.

1.4.2.2 Human-Robot Interaction

In interactive systems, robotic musicians can be programmed to trigger and modify pre-composed musical phrases. For example, the robot Kritaanjli [54] can accompany improvising musicians by providing the underlying melodic content of Indian ragas while using additional user input to switch between or modify the ragas. Shimon Robot uses predefined melodic phrases in a piece called *Bafana* by Gil Weinberg, inspired by African marimba ensembles. Shimon listens for specific motifs and can either synchronize or play these motifs in canon, creating interlocking rhythms and harmonies. At another point in the piece Shimon begins to change motifs using stochastic decision processes that can surprise the human performers and drive their response to improvised directions [46, 55].

Some interaction scenarios designed for robotic musicians employ a hybrid of statistical models and rules. Different modes may rely on different note generation techniques, which can be triggered by interacting musicians either through the music (playing a specific phrase or chord), physical motion (completing a predefined gesture) or other sensor (light, pressure, etc.). Miranda and Tikhanoff describe an autonomous music composition system for the Sony AIBO robot that employs statistical models as well as rules [56]. The system is trained on various styles of music and a short phrase is generated based off of the previous chord and first melodic note of the previous phrase.

Weinberg et al. use a real-time genetic algorithm to establish human-robot improvisation [57]. Short melodic phrases recorded by a pianist form the initial base population. For the fitness function, dynamic time warping (DTW) is used as a similarity metric between the base population and phrases played by a human performer in real time. Mutating the phrases allows for more complexity and richer melodic content by transposing, inverting, and adding semitones to the phrases. The algorithm was implemented in the robot *Haile*. The performance uses a call and response methodology in which the human performer and robot take turns using each others music to influence their own responses.

Decentralized musical systems have been explored by Albin et al. using musical swarm robots [58]. As part of the system, several mobile robots are equipped with smart-phones and solenoids that strike pitched bells. Each robot's location is mapped to different playing behaviors, which manipulates the dynamics and rhythmic patterns played on the bell. New rhythmic and melodic phrases emerge when the robots undergo a set of swarming behaviors such as predator-prey, follow-the-leader, and flocking. Additionally, humans can interact with the robots by moving along the installation allowing the robots to respond to their presence using a face detection application running on the phones.

1.4.2.3 Modeling Human Performers

When reading from a pre-composed score robotic musicians do not need to decide what notes to play, rather they have to be equipped with algorithms that would allow

them to decide how to play the notes in order to create an expressive performance. Designing generative expression engines for robotic musicians is often achieved by observing human performers. Solis et al. [59] describe such a methodology where the expression of a professional flute player is modeled using artificial neural networks (ANNs), extracting various musical features (pitch, tempo, volume, etc.) from the performance. The networks learn the human's expressive mapping and output parameters describing a note's duration, vibrato, attack time, and tonguing. These parameters are then sent to the robot's control system.

Modeling human performers can also be used for improvisational systems. In one such example, Shimon Robot generates notes using Markov decision processes trained on performances of jazz greats such as John Coltrane and Thelonious Monk [51]. The note pitch and duration of the Jazz masters transcribed improvisation are modeled in 2nd order Markov chains. Through an iPhone application a listener can control the robot's playing by choosing different artists to model and even mix their styles. A different example for this approach is shown by Kapur et al., who developed a robotic harmonium called *Kritaanjli*, The robot extracts information from a human musician's style of harmonium pumping and attempts to emulate it [54]. The robot's motors use information from a linear displacement sensor that measures the human's pump displacement.

1.5 Embodiment

By integrating musical mechatronics with machine musicians, researchers can bring together the computational, auditory, visual, and physical worlds, creating a sense of embodiment—the representation or expression of objects in a tangible and visible form. It has been shown that the notion of embodiment can lead to richer musical interactions as co-musicians and audience members demonstrate increased attention, an ability to distinguish between the robot's pragmatic actions (physical movements performed to bring the robot closer to its goal such as positioning its limbs to play a note) and epistemic actions (physical movements performed to gather information or facilitate computational processing), and better musical coordination and synchronization as a result of the projection of physical presence.

Several non-musical human-robotic interaction studies have demonstrated desirable interaction effects as a result of physical presence, showing that physical embodiment leads to increased levels of engagement, trust, perceived credibility, and more enjoyable interactions [60, 61]. Robotic musicians yield similar benefits for those directly involved with the interaction (other musicians) as well those merely witnessing it (audience members). Moreover, Hoffman and Weinberg examined the effects of physical presence and acoustic sound generation on rhythmic coordination and synchronization [62]. The natural coupling between the robot's spatial movements and sound generation afforded by embodiment allows for interacting musicians to anticipate and coordinate their playing with the robot, leading to richer musical experiences.

In addition to the cues provided by a robotic musician's sound producing movements, physical ancillary gestures can create better informed social interactions by communicating cues related to a task, an interaction, or the system state. An important emphasis in our own research has been the design and development of robotic physical gestures that are not directly related to sound production, but can help convey useful musical information for human musicians and observing audiences. These gestures and physical behaviors are used to express the underlying computational processes allowing observers to better understand how the robot perceives and responds to the music and the collaborating human musicians. A robot's gaze, posture, velocity, and motion trajectories can be manipulated for communicative function. For example, simple behaviors designed for Shimon Robot, such as nodding to the beat, looking at its marimba during improvisation, and looking at interacting musicians when listening help enhance synchronization among the ensemble members and reinforce the tempo and groove of the music.

In addition to their communicative function, physical gestures have been shown to lessen cognitive load for human subjects in human-human interactions [63]. Unfortunately, the same is not yet true for robots, since using current computation methods, adding gestural components tend to increase computational and design complexity. For Shimon we utilize a large body of work in animation and robotics relevant to expressive physical behaviors. Other gestural robotic systems use their own unique physical designs and DoFs, which makes it difficult to draw from ideas or configurations of other systems. Therefore, user studies and surveys are essential for evaluating and developing physical behavior systems. For example, by drawing from research in psychology, animation, and sociology we designed and evaluated a number of communicative gestures for Shimon such as slowly closing its eyelids in a squinting fashion to convey confusion or intrigue and opening them wide to convey surprise or enjoyment.

Another important embodiment application for robotic musicians is path planning. The term 'path planning' in general robotics applications refers to the process of developing a plan of discrete movements that satisfies a set of constraints to reach a desired location. In a musical context, robots may not be able to perform specific instructions in a timely, coordinated, and expressive manner, which requires employing musical considerations in designing a path-planning algorithm. Such considerations can shed light on conscious as well as subconscious path planning decisions that human musicians employ while coordinating their gestures (such as finger order in piano playing).

Musical decision-making that is influenced by the physical body has great implications for robotic musicians. Similarly to humans, who are limited by the physical constraints of their physiology, robots are limited by their mechanics and design. While robotic musicians can be designed to outperform their human counterparts in terms of speed, precision, and dynamic range, their mechanical constraints often impose limitations that directly hinder the musical outcome. For example, Shimon's four-arm design limits its ability to play certain kinds of chords and musical passages, which could be easy for humans to play. In robotic musicianship path planning and note generative and rhythmic decisions become an integrated process.

1.6 Integrating Robotic Musicianship into New Interfaces

As the field of robotic musicianship expands, researchers look for new ways in which humans can interact with robots through music. At the Georgia Tech Center for Music Technology, we are exploring two types of robotic interfaces that incorporate musicianship: robotic musical companions and wearable robotics.

1.6.1 Musical Companion Robots

Shimi is a robotic musical companion that has five DoFs, three speakers, and uses a smart phone to perform all the higher-level computation related to motion and musical intelligence (Fig. 1.3). The robot was initially designed to function as an interactive speaker dock enabling users to interact with their music libraries through a number

Fig. 1.3 *Shimi*, the robotic music companion has five DoFs and uses a smart phone to perform higher level movement control and perception capabilities

of human-robotic interaction modalities. It has since evolved to include applications that require musicianship traits including perceptual and generative functions. For example, the robot can convey music-based social and emotional cues, provide accompaniment to other musicians, and synchronize and dance to a pre-recorded score [64].

Unlike other robotic musicians described in this paper, Shimi is not designed to generate acoustic sound, rather it produces sound by digital means. While this approach does not convey a causal link between movement and sound generation, the robot does possess gestural affordances that connect its movements to the synthesized sounds it produces in an indirect manner. Two basic methods were designed for linking gestures and sound, informed by aspects of music perception and human-movement science.

1. *impulsive*—This connection corresponds to the action of striking a percussive instrument and generating a shorter burst of acoustic energy. Sound is produced at a single point during the gesture, though characteristics of the sound are influenced by the motion leading up to and following production.
2. *sustained*—This connection corresponds to the action of bowing a stringed instrument or blowing a wind instrument. The motion and sound production are connected continuously throughout the entirety of the gesture.

The framework for implementing these two methods consists of a motion controller that allows for variable velocity contours of motion for each DoF. For impulsive sound production, one or several DoFs accelerate toward a defined location, culminating with the generation of sound once the location is reached. This method is used to create sounds with onsets that exhibit strong and sharp attack transients. In sustained sound production, the duration of a sound is temporally aligned with the duration of a movement. To create consistency and to maintain sonic and acoustic contingency specific sounds and musical parameters (such as pitch, timbre, or volume) are mapped to specific gestures and DoFs. This gives the illusion that Shimi is playing an instrument and provides visual cues in a similar fashion to that of robotic acoustic instrumentalists.

1.6.2 Wearable Robotic Musicians

Another research direction we have explored is wearable robotic musicians such as a musical robotic prosthetic for an amputee drummer (Fig. 1.4). The prosthetic is fitted with two DC motors, each controlling the motion of a single drumstick. The first motor is controlled by the user and is designed to recreate the functionality available to a biological arm, notably the palm and the fingers. The motor controls the rebound of the stick allowing the amputee drummer to control a wide range of expressive drum strokes, including single, double, buzz rolls, and others. The second drumstick functions as a robotic musician that responds to the music and improvises based on computational music analysis.

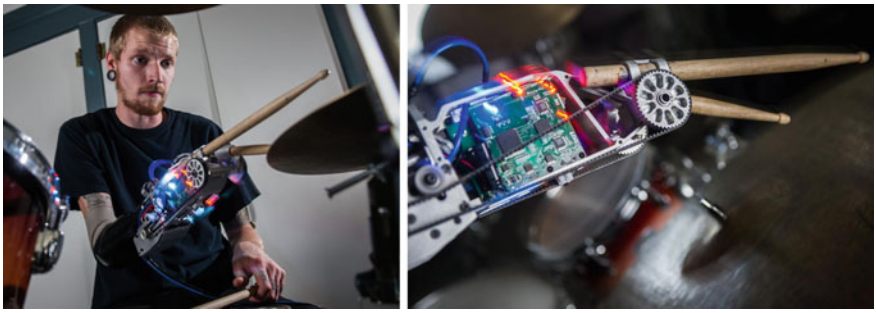


Fig. 1.4 A robotic drumming prosthetic equipped with two sticks, one of which has a "mind of its own" and responds to the amputee's playing and movements

While the amputee drummer does not have full control over the second stick, he can react and collaborate with it, creating novel interactive human-robot collaboration. In designing the algorithms controlling both drum sticks, we have taken into consideration the shared control aspects that the human and the machine have over the final musical result. We also took into consideration the physical interdependencies that may occur due to the fact the robotic musician is embedded in the human's own body. While the artificial intelligence is designed to generate different stroke types and rhythmic patterns, the human drummer can influence the final musical result through his own motion and manipulation of the distance between the autonomous stick and the drum. Concurrently, the autonomous stick has the opportunity to complement, contrast, or disrupt the drummer's own patterns.

Such a wearable system yields several research questions: What type of interaction between the human and robot is optimal for achieving fluid and comfortable playability that encourages musical cooperation, yet remains non-restrictive? What type of sensing by the robot is necessary to support creative musicianship? Is it possible for other musicians in the group to interact with and influence the robotic musician?

1.7 Discussion

In this chapter we presented several representative works in the field of robotic musicianship and discussed some of the different methodologies and challenges inherent to designing mechanical systems that can understand, reason and generate music and musical interactivity. The projects discussed in this chapter demonstrates the interdisciplinary nature of the field, from mechanics, through perception, to artificial creativity. Each system features a unique balance between these disciplines, as each is motivated by a different mixture of goals and challenges, from artistic, through scientific, to ones that are engineering-driven. This interdisciplinary set of goals and motivations makes it difficult to develop a standard for evaluating robotic musicians.

For example, when the goal of a project is to design a robot that can improvise like a jazz master, a musical Turing test can be used for evaluation; if the goal is artistic and experimental in nature, such as engaging listeners with an innovative performance, a focus group assessing audiences reviews of the performance can evaluate success; and if the project's motivation is to study and model humans' sound generation mechanism, anatomy- and cognitive-based evaluation methods can be employed.

Many of the projects discussed here include evaluation methods that address the project's specific goals and motivations. Some projects, notably those with artistic and musical motivations, do not include an evaluation process. Others address multiple goals, and therefore include a number of different evaluation methods for each aspect of the project. For example, in computational-cognition based projects, robots' perceptual capabilities can be measured by comparing the algorithm against a ground truth and calculating accuracy. However, establishing ground truths in such projects has proven to be a challenging task as many parameters in music are abstract and subjective. HRI motivated experiments, on the other hand, are designed to evaluate interaction parameters such as the robots' abilities to communicate and convey relevant information through their music generating functions or physical gesture. We believe that as research and innovation in robotic musicianship continues to grow and mature, more unified and complete evaluation methods will be developed.

Though robotic musicianship has seen many great developments in the last decade, there are still many white spaces to be explored and challenges to be addressed—from developing more compelling generative algorithms to designing better sensing methods enabling the robot's interpretations of music to correlate more closely with human perceptions. In addition to developing new interfaces, sensing methods, and generative algorithms we are also excited to showcase our developments through performances as part of the *Shimon Robot and Friends Tour*.[8] We look forward to continue to observe and contribute to new developments in this field, as robotic musicians continue to increase their presence in everyday life, paving the way to the creation of novel musical interactions and outcomes.

References

1. Kapur, Ajay. 2005. A history of robotic musical instruments. In *Proceedings of the International Computer Music Conference*, 21–28. Citeseer.
2. Rowe, Robert. 2004. *Machine musicianship*. The MIT press.
3. Weinberg, Gil, and Driscoll Scott. 2006. Toward robotic musicianship. *Computer Music Journal* 30 (4): 28–45.
4. Williamson, Matthew M. 1999. Robot arm control exploiting natural dynamics. PhD thesis, Massachusetts Institute of Technology.
5. Williamson, Matthew M. 1998. Rhythmic robot arm control using oscillators. In *Proceedings of IEEE/RSJ international conference on intelligent robots and systems, 1998*, vol. 1, 77–83. IEEE.

[8] http://www.shimonrobot.com.

6. Atkeson, Christopher G., Joshua G. Hale, Frank Pollick, Marcia Riley, S. Shinya Kotosaka, Tomohiro Shibata Schaul, Gaurav Tevatia, Ales Ude, Sethu Vijayakumar, et al. 2000. Using humanoid robots to study human behavior. *IEEE Intelligent Systems and their Applications* 15 (4): 46–56.
7. Ichiro, Kato, Ohteru Sadamu, Shirai Katsuhiko, Matsushima Toshiaki, Narita Seinosuke, Sugano Shigeki, Kobayashi Tetsunori, and Fujisawa Eizo. 1987. The robot musician 'wabot-2' (waseda robot-2). *Robotics* 3 (2): 143–155.
8. Zhang, Ada, Mark Malhotra, and Yoky Matsuoka. Musical piano performance by the act hand. In *2011 IEEE international conference on robotics and automation (ICRA)*, 3536–3541. IEEE.
9. Singer, Eric, Jeff Feddersen, Chad Redmon, and Bil Bowen. 2004. Lemur's musical robots. In *Proceedings of the 2004 conference on New interfaces for musical expression*, 181–184. National University of Singapore.
10. Ajay, Kapur, E. Singer Trimpin, Afzal Suleman, and George Tzanetakis. 2007. *A comparison of solenoid-based strategies for robotic drumming*. Copenhagen, Denmark: ICMC.
11. Weinberg, Gil, and Scott Driscoll. 2007. The design of a perceptual and improvisational robotic marimba player. In *The 16th IEEE international symposium on robot and human interactive communication, 2007. RO-MAN 2007*, 769–774. IEEE.
12. Laura, Maes, Raes Godfried-Willem, and Rogers Troy. 2011. The man and machine robot orchestra at logos. *Computer Music Journal* 35 (4): 28–48.
13. Raes, G.W. 2014. Robots and automatons catalogue. http://logosfoundation.org/instrumgwr/automatons.html. Accessed 2 Oct 2014.
14. Kemper, Steven, Troy Rogers, and Scott Barton. 2014. EMMI: Expressive machines musical instruments. http://www.expressivemachines.com/. Accessed 6 Oct 2014.
15. Trimpin. Portfolio. In *Seattle*, Washington.
16. Kapur, Ajay, Michael Darling, Dimitri Diakopoulos, Jim W. Murphy, Jordan Hochenbaum, Owen Vallis, and Curtis Bahn. 2011. The machine orchestra: An ensemble of human laptop performers and robotic musical instruments. *Computer Music Journal* 35 (4): 49–63.
17. Jordà, Sergi. 2002. Afasia: The ultimate homeric one-man-multimedia-band. In *Proceedings of the 2002 conference on new interfaces for musical expression*, 1–6. National University of Singapore.
18. Baginsky, N.A. 2004. The three sirens: A self learning robotic rock band. http://www.the-three-sirens.info/binfo.html. Accessed 2 Oct 2014.
19. McVay, J., D.A. Carnegie, J.W. Murphy, and A. Kapur. 2012. Mechbass: A systems overview of a new four-stringed robotic bass guitar. In *Proceedings of the 2012 electronics New Zealand conference*, New Zealand, Dunedin
20. Chadefaux, Delphine, Jean-Loic Le Carrou, Marie-Aude Vitrani, Sylvere Billout, and Laurent Quartier. Harp plucking robotic finger. In *2012 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 4886–4891. IEEE.
21. Koji, Shibuya, Ideguchi Hironori, and Ikushima Katsunari. 2012. Volume control by adjusting wrist moment of violin-playing robot. *International Journal of Synthetic Emotions (IJSE)* 3 (2): 31–47.
22. Shibuya, Koji, Shoji Matsuda, and Akira Takahara. Toward developing a violin playing robot-bowing by anthropomorphic robot arm and sound analysis. In *The 16th IEEE international symposium on robot and human interactive communication, 2007. RO-MAN 2007*, 763–768. IEEE.
23. Almeida, Andre, George David, Smith John, and Wolfe Joe. 2013. Theclarinet: How blowing pressure, lip force, lip position and reed "hardness" affect pitch, sound level, and spectrum. *The Journal of the Acoustical Society of America* 134 (3): 2247–2255.
24. Dannenberg, Roger B., H. Ben Brown, and Ron Lupish. 2011. Mcblare: A robotic bagpipe player. In *Musical Robots and Interactive Multimodal Systems*, 165–178. Springer.
25. Ferrand, Didier, and Christophe Vergez. 2008. Blowing machine for wind musical instrument: Toward a real-time control of the blowing pressure. In *2008 16th mediterranean conference on control and automation*, 1562–1567. IEEE.

26. Solis, Jorge, Atsuo Takanishi, and Kunimatsu Hashimoto. 2010. Development of an anthropomorphic saxophone-playing robot. In *Brain, body and machine*, 175–186. Springer.
27. Solis, Jorge, Klaus Petersen, Takeshi Ninomiya, Masaki Takeuchi, and Atsuo Takanishi. Development of anthropomorphic musical performance robots: From understanding the nature of music performance to its application to entertainment robotics. In *IEEE/RSJ international conference on intelligent robots and systems, 2009. IROS 2009*, 2309–2314. IEEE.
28. McPherson, Andrew. 2010. The magnetic resonator piano: Electronic augmentation of an acoustic grand piano. *Journal of New Music Research* 39 (3): 189–202.
29. Menzies, Duncan, Andrew McPherson. 2012. An electronic bagpipe chanter for automatic recognition of highland piping ornamentation. In *Proceedings of NIME*, Ann Arbor, MI, USA
30. Kapur, Ajay, Ariel J. Lazier, Philip Davidson, R. Scott Wilson, and Perry R. Cook. 2004. The electronic sitar controller. In *Proceedings of the 2004 conference on new interfaces for musical expression*, 7–12. National University of Singapore.
31. Logan-Greene, Richard. 2011. *Submersions I*. University of Washington.
32. Bello, Juan Pablo, Laurent Daudet, Samer Abdallah, Chris Duxbury, Mike Davies, and Mark B. Sandler. 2005. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13 (5): 1035–1047
33. Collins, Nicholas M. 2006. Towards autonomous agents for live computer music: Realtime machine listening and interactive music systems. PhD thesis, Citeseer.
34. Klapuri, Anssi. 1999. Sound onset detection by applying psychoacoustic knowledge. In *Proceedings of 1999 IEEE international conference on acoustics, speech, and signal processing*, vol. 6, 3089–3092. IEEE.
35. Davies, Matthew E.P., and Mark D. Plumbley. 2004. Causal tempo tracking of audio. In *ISMIR*.
36. Barton, Scott. 2013. The human, the mechanical, and the spaces in between: Explorations in human-robotic musical improvisation. In *Ninth artificial intelligence and interactive digital entertainment conference*.
37. Lerch, Alexander. 2012. *An introduction to audio content analysis: Applications in signal processing and music informatics*. Wiley.
38. Paiva, Rui Pedro, Teresa Mendes, and Amílcar Cardoso. 2005. On the detection of melody notes in polyphonic audio. In *ISMIR*, 175–182.
39. Chen, Ruofeng, Shen Weibin, Srinivasamurthy Ajay, and Chordia Parag. 2012. Chord recognition using duration-explicit hidden Markov models. *ISMIR* 445–450
40. Alexander, Sheh, and Daniel P.W. Ellis. 2003. Chord segmentation and recognition using em-trained hidden Markov models. *ISMIR* 2003: 185–191.
41. Solis, Jorge, Keisuke Chida, Shuzo Isoda, Kei Suefuji, Chiaki Arino, and Atsuo Takanishi. The anthropomorphic flutist robot wf-4r: From mechanical to perceptual improvements. In *2005 IEEE/RSJ international conference on intelligent robots and systems (IROS 2005)*, 64–69. IEEE.
42. Mizumoto, Takeshi, Angelica Lim, Takuma Otsuka, Kazuhiro Nakadai, Toru Takahashi, Tetsuya Ogata, and Hiroshi G. Okuno. 2010. Integration of flutist gesture recognition and beat tracking for human-robot ensemble. In *Proceedings of IEEE/RSJ-2010 workshop on robots and musical expression*, 159–171.
43. Otsuka, Takuma, Kazuhiro Nakadai, Toru Takahashi, Tetsuya Ogata, and Hiroshi G. Okuno. 2011. Real-time audio-to-score alignment using particle filter for coplayer music robots. *EURASIP Journal on Advances in Signal Processing* 2011: 2.
44. Matthew, Wright. 2005. Open sound control: An enabling technology for musical networking. *Organised Sound* 10 (3): 193.
45. Pan, Ye, Min-Gyu Kim, and Kenji Suzuki. A robot musician interacting with a human partner through initiative exchange. In *Proceedings of conference on new interfaces for musical expression*, 166–169.
46. Bretan, M., M. Cicconet, R. Nikolaidis, and G. Weinberg. Developing and composing for a robotic musician. In *Proceedings of international computer music conference (ICMC'12)*, Ljubljana, Slovenia, Sept 2012.

47. Lim, Angelica, Takeshi Mizumoto, Louis-Kenzo Cahier, Takuma Otsuka, Toru Takahashi, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Robot musical accompaniment: integrating audio and visual cues for real-time synchronization with a human flutist. In *2010 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 1964–1969. IEEE.
48. Kapur, Ajay. 2011. Multimodal techniques for human/robot interaction. In *Musical robots and interactive multimodal systems*, 215–232. Springer.
49. Hochenbaum, Jordan, and Ajay Kapur. 2012. Drum stroke computing: Multimodal signal processing for drum stroke identification and performance metrics. In *International conference on new interfaces for musical expression*.
50. Weinberg, Gil, Andrew Beck, and Godfrey Mark. 2009. Zoozbeat: A gesture-based mobile music studio.
51. Nikolaidis, Ryan, and Gil Weinberg. 2010. Playing with the masters: A model for improvisatory musical interaction between robots and humans. In *2010 IEEE, RO-MAN*, 712–717. IEEE.
52. Cicconet, Marcelo, Bretan Mason, and Weinberg Gil (2013) Human-robot percussion ensemble: Anticipation on the basis of visual cues. *Robotics & Automation Magazine*. 20 (4): 105–110 (IEEE).
53. Metheny, Pat. Orchestrion. http://www.theorchestrionproject.com/. Accessed 5 Jan 2014.
54. Kapur, Ajay, Jim Murphy, and Dale Carnegie. Kritaanjli: A robotic harmonium for performance, pedogogy and research.
55. Guy, Hoffman, and Weinberg Gil. 2011. Interactive improvisation with a robotic marimba player. *Autonomous Robots* 31 (2–3): 133–153.
56. Miranda, Eduardo R., and Vadim Tikhanoff. 2005. Musical composition by autonomous robots: A case study with aibo. *Proceedings of TAROS 2005 (Towards autonomous robotic systems)*.
57. Weinberg, Gil, Mark Godfrey, Alex Rae, and John Rhoads. 2008. A real-time genetic algorithm in human-robot musical improvisation. In *Computer music modeling and retrieval. Sense of sounds*, 351–359. Springer.
58. Albin, Aaron, Gil Weinberg, and Magnus Egerstedt. Musical abstractions in distributed multi-robot systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 451–458. IEEE.
59. Solis, Jorge, Kei Suefuji, Koichi Taniguchi, Takeshi Ninomiya, Maki Maeda, and Atsuo Takanishi. Implementation of expressive performance rules on the wf-4riii by modeling a professional flutist performance using NN. In *2007 IEEE international conference on robotics and automation*, 2552–2557. IEEE.
60. Kidd, Cory D., and Cynthia Breazeal. 2004. Effect of a robot on user perceptions. In *Proceedings of 2004 IEEE/RSJ international conference on intelligent robots and systems, 2004 (IROS 2004)*, vol. 4, 3559–3564. IEEE.
61. Kidd, C., and Cynthia Breazeal. 2005. Comparison of social presence in robots and animated characters. *Proceedings of human-computer interaction (CHI)*.
62. Hoffman, Guy, and Gil Weinberg. Synchronization in human-robot musicianship. In *2010 IEEE, RO-MAN*, 718–724. IEEE.
63. Goldin-Meadow, Susan, Howard Nusbaum, Spencer D. Kelly, and Susan Wagner. 2001. Explaining math: Gesturing lightens the load. *Psychological Science* 12 (6): 516–522.
64. Bretan, Mason, and Gil Weinberg. 2014. Chronicles of a robotic musical companion. In *Proceedings of the 2014 conference on new interfaces for musical expression*. University of London, Goldsmiths.

Chapter 2
Platforms—Georgia Tech's Robotic Musicians

2.1 Abstract

In Chap. 1 we surveyed a wide range of sound production mechanisms that could allow musical robots to play percussive, stringed, wind, non-traditional and augmented musical instruments. We discussed embedding musical intelligence—both perceptual and generative—in such robots, which could transform instruction-following musical robots into responsive and creative *Robotic Musicians*. We also addressed the notion of embodiment in robotic musicianship, which includes not only sound producing physical gestures but ancillary motions that are not involved directly in sound production. These embodied gestures can provide co-players and audiences with expressive and communicative visual cues that can help coordination, synchronization, interaction as well as appreciation of the musical human-robot interaction.

In this chapter, we will present in detail the four robotic musicianship platforms developed at Georgia Tech Center for Music Technology (GTCMT). These platforms, mainly geared at percussive instruments, provided the basis for the research presented in the remainder of the book. Although we did not explore sound generation techniques for string or wind-based instruments, we believe that the approaches we took for our percussive robotic musicianship platforms could provide useful insights for future researchers into the rationale, design, engineering and evaluation of a wide range of robotic musicians.

The first robot we present is **Haile**—a drum playing robot that utilizes multiple actuators to generate a variety of percussive sounds. Haile can listen to human percussionists and respond using two percussive arms that are designed to strike and move across the drum-head. Haile, however, was not designed to produce social ancillary gestures. GTCMT's second robotic musicianship platform, **Shimon**, is a marimba playing robot that was designed not only to play rhythmic patterns but melodies and harmonies as well. Shimon also features a gestural head that exhibits social ancillary motions. The third robot that will be described in this chapter is **Shimi**, which focuses on music analysis driven movements. Rather than generating acoustic sound, Shimi

G. Weinberg et al., *Robotic Musicianship*, Automation, Collaboration,
& E-Services 8, https://doi.org/10.1007/978-3-030-38930-7_2

plays digital music from his ear-like speakers. In the last section of this chapter we describe the **Robotic Drumming Prosthetic (RDP)**—a wearable robotic platform designed to augment human drummers by modeling human percussive expression as well as extending human virtuosity and drumming abilities to uncharted domains.

2.2 Haile—A Robotic Percussionist

2.2.1 Motivation

GTCMT's first robotic musician, Haile, is a robotic drummer designed to make music collaboratively with human players [1]. Haile can listen to rhythmic input and respond using improvisatory algorithms. The main goal behind the development of Haile was to provide a test-bed for novel forms of musical human-machine interaction, bringing together human creativity and robotic capabilities to create unique and inspiring musical outcomes. Unlike musical human-computer interaction, this project focused on bringing human-machine interaction into the physical world, providing rich acoustic sounds and visual queues that could enhance the musical experience for both players and audiences. Haile's physical motions help accompanying humans to anticipate and coordinate their playing, and give audience a physical and visual connection to the sound generation process. We propose that a collaborative and physical platform like Haile could lead to novel human-machine interaction and new and engaging musical experiences unlikely to be conceived by traditional means.

2.2.2 Design

Haile was designed to play a Native American powwow drum, a multi-player instrument used in ceremonial events. We chose an anthropomorphic design to provide Haile with an inviting and familiar look, but also left many of the electronic components exposed to reflect the technological nature of the robot, enabling co-players and audiences to get a sense of the underlying mechanism of the robot's operation. We uncovered mechanical apparatuses and LEDs embedded in Haile's body to provide an additional representation of its physical a actions (see Fig. 2.1). To match the natural aesthetics of the Native American powwow ritual, we chose to construct the robot from wood. A design made by Clint Cope was used as a basis for Haile's appearance. The wooden parts were made using a CNC wood-cutting machine and constructed from several layers of plywood. Metal joints were designed to allow shoulder and elbow movement as well as leg adjustability for different drum heights.

Haile's anthropomorphic design employs two percussive arms that can move across different locations along the drum's radius and strike with varying velocities. The right arm was designed to play fast notes, and the left arm was designed to

Fig. 2.1 Haile's anthropomorphic design

produce larger and more visible motions that produce louder sounds in comparison to the right arm. Both arms can adjust the sound of the strikes in two manners: 1. Different pitches are achieved by striking the drum-head in different locations along its radius, and 2. Volume is adjusted by hitting with varying velocities. To move to different positions over the drum-head, each arm employs a linear slide, a belt, a pulley system, and a potentiometer that provides feedback (see Fig. 2.2). Unlike robotic drumming systems that allow hits at only a few discrete locations [2, 3]; Haile's arms can strike anywhere on a line between the center and the rim of the drum, moving the 25 cm between these two points in about 250 ms.

Haile's right arm's striking mechanism consists of a solenoid-driven device and a return spring (see Fig. 2.3). The arm strikes at a maximum speed of 15 Hz, faster than the left arm's maximum speed of 11 Hz. However, the right arm cannot generate a wide dynamic range or provide clearly noticeable visual cues, which limits Haile's expression and visual cues. The left arm was designed to address these shortcom-

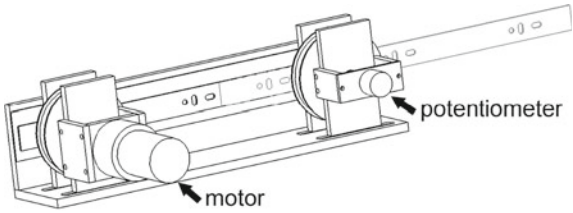


Fig. 2.2 The right arm slider mechanism

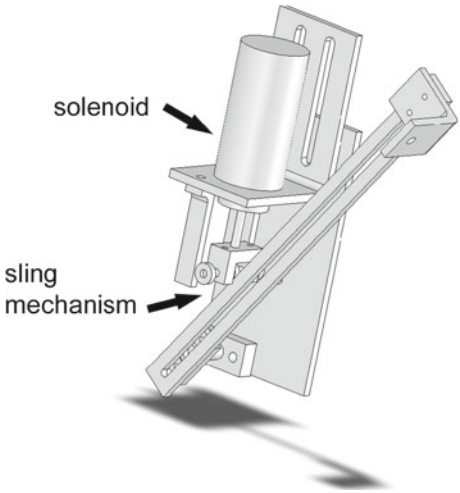


Fig. 2.3 The right arm striker mechanism

ings using larger visual movements and a more powerful and sophisticated hitting mechanism. Whereas the striking component of the right arm can move only 6 cm vertically, the entire left forearm takes part in the striking motion and can move up and down at about 20 cm. A linear motor and an encoder located at the elbow are used to provide sufficient force and control for the larger mass and motions. (Additional images showing the mechanical construction of the arms are provided in Figs. 2.4 and 2.5).

In an effort to provide an easy-to-program environment for Haile, we used Max/MSP, an intuitive graphical programming environment that could make the project accessible to composers, performers, and students. Our first one-armed prototype incorporated the USB-based Teleo System from MakingThings as the main interface between Max/MSP and Haile's sensors and motors. The low-level control of the solenoid-based right arm's position required a continuous feed of position updates. This consumed a significant part of the communication bandwidth and processor power on the main computer. Therefore, for the second arm we used multiple on-board microprocessors for local low-level control and Ethernet communication to connect to the main computer. Each arm was locally controlled by an 18F452 PIC microprocessor, both of which received RS232 communications from a Modtronix Ethernet board (SBC68EC). The Ethernet board received 3-byte packets from the

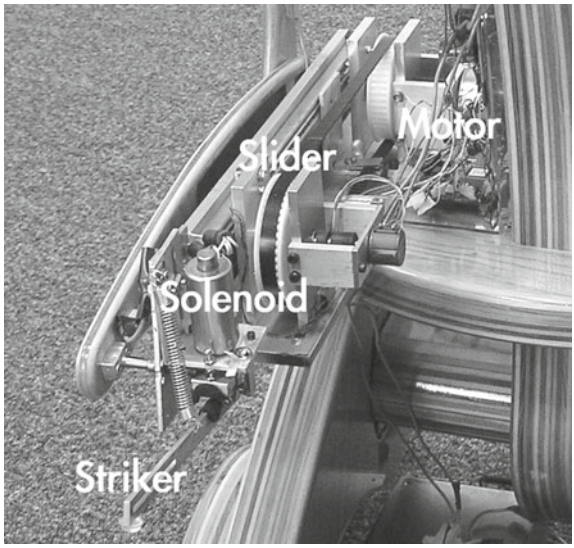


Fig. 2.4 Haile's right arm design

computer: a control byte and two data bytes. The protocol used an address bit in the control byte to send the information onto the appropriate arm processor. The two data bytes contained position and velocity set points for each hit, but could also be used to update the control parameters.

Two on-board PIC microprocessors were responsible for controlling the arms' sliding and hitting mechanisms, ensuring that the impact occurs at the desired positions and velocities. To allow enough time for the arms to slide to the correct locations and execute strokes, a 300-ms delay line was implemented between signal reception and impact. It has been shown that rhythmic errors of only 5 ms can be detectable by average listeners [4]; therefore, it was important to ensure that this delay remained accurate and constant regardless of different hit velocities. Both arms stored incoming hit commands in a first-in-first-out queue, moving toward the location of a new note immediately after each hit. Owing to its short vertical hitting range, the solenoid-driven right arm had a fairly consistent stroke time for both soft and loud hits. We therefore implemented an additional 300-ms delay as a constant for this arm. The left arm, on the other hand, underwent much larger movements, which required complex feedback control to ensure that impacts occur at the correct time regardless of hit velocity. While waiting for incoming notes, the left arm remained about one inch (about 2.5 cm) above the surface of the drum. When a new note was received, the arm was raised to a height proportional to the loudness of the hit. After a delay determined by the desired velocity and elevation, the arm started descending toward the drum-head under velocity control. After impact, the arm returned back to its standby position above the drum-head. Extremely fast notes employed a slightly different control mode that made use of the bounce of the arm in preparation for the next hit. The left arm, therefore, controlled a wide dynamic range and provided performers

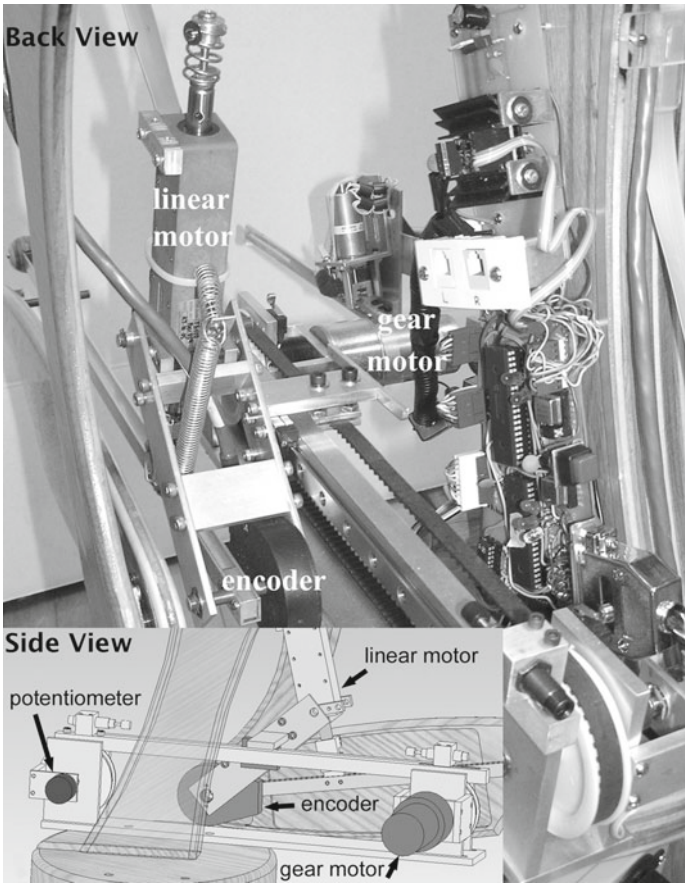


Fig. 2.5 The linear motor and encoder provide closed-loop position and velocity over the left arm height while the gear motor and potentiometer control distance from the center of the drum

and viewers with anticipatory and real-time visual cues, enhancing expression and enriching the interaction representation.

From an acoustic point of view, Haile could generate a wide range of sounds by hitting a drum with variety of hit intensities, speeds, locations, different striker materials, contact areas, contact duration, and drum-head pressure levels. From a communicative point of view, Haile's left arm could provide clear visual cues that allowed co-player to anticipate and synchronize their gestures to Haile's. While its right arm could played faster than its left arm, it did not provide clear visual cues, and could not support the richer sonic possibilities afforded by the left arm.

2.3 Shimon—A Robotic Marimba Player

The next robotic musician designed at GTCMT was *Shimon*, a marimba playing robot including eight actuated mallets and a socially expressive head (Fig. 2.6). Shimon was designed to expand our research in robotic musicianship beyond the Haile platform by introducing two main additional abilities: 1. The ability to play a melodic and harmonic instrument including chords with high note density; 2. The ability to communicate with co-players using visible socio-musical ancillary gestures.

2.3.1 Striker Design

To allow for high density polyphonic melodic generation, Shimon was designed to have four arms, each carrying two solenoid-driven marimba mallets (Fig. 2.7). Using linear actuators, the arms were designed to move horizontally along a shared rail in parallel to the marimba's 4 octave range. Figure 2.8 shows a top-down diagram

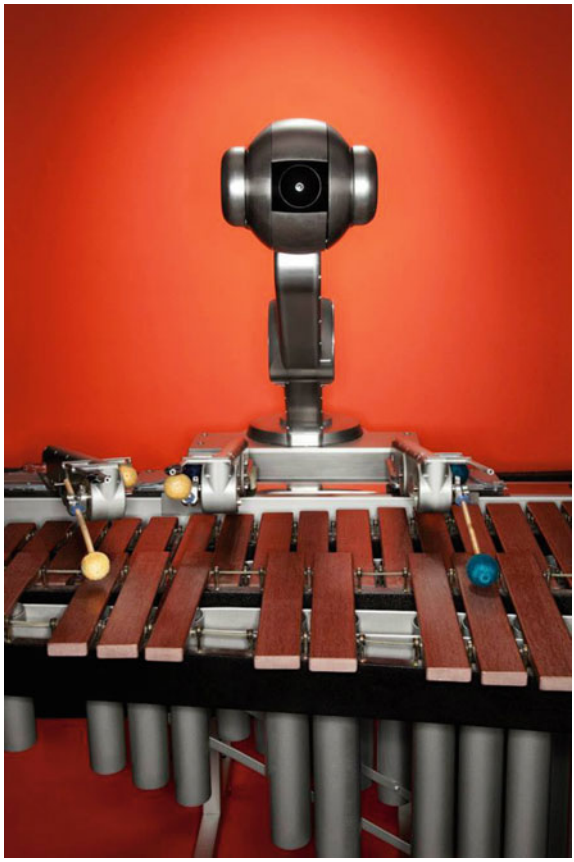


Fig. 2.6 Shimon is a robotic marimab player with eight actuated mallets mounted on four sliding arms, and a socially expressive head

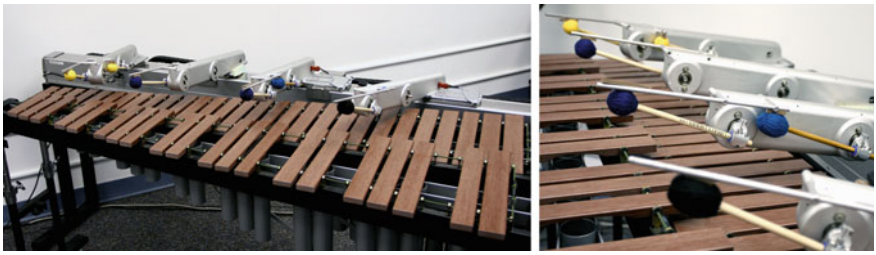


Fig. 2.7 Overall view (left) and detail view (right) of the robotic marimba player *Shimon*. Four arms share a voice-coil actuated rail. Two rotational solenoids per arm activated mallets of varying firmness

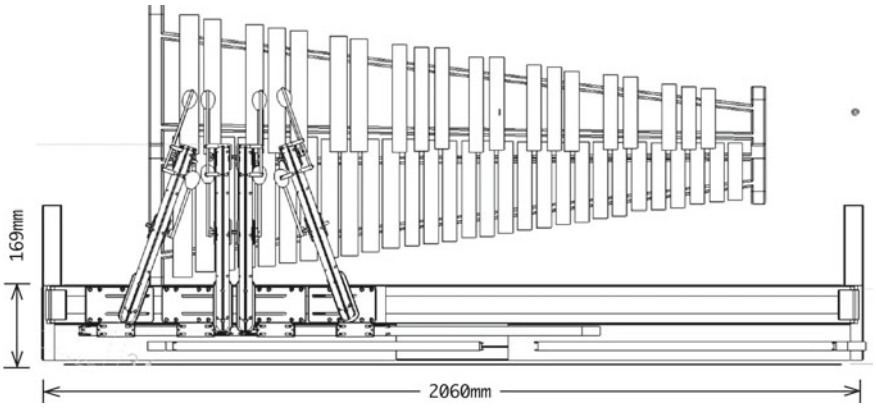


Fig. 2.8 Overall diagram of the relationship between the linear actuator, arms, and instrument of the robotic marimba player *Shimon*

depicting the relationship between the linear actuator, the mallet-carrying arms, and the marimba.

The linear actuators were based on a commercial product by Intelligent Actuator, Inc. (IAI), controlled by a SCON position and trajectory controller. Each arm could reach an acceleration of 3 *g* (gravitational force equivalent), and—at top speed—move at approximately one octave per 0.25 s.

The arms were custom-made aluminum shells, each carrying two marimba mallets. For fast striking action, the mallets are mounted on rapid-acting rotational solenoid motors. One solenoid was mounted approximately at the arm's midpoint, positioned to reach the bottom-row ("white") keys, and the other at the edge of the arm, positioned to reach the top-row ("black") keys (Fig. 2.9). The mallets were chosen with an appropriate softness to fit the area of the marimba that they are most likely to play. *Shimon* was designed and built in collaboration with Roberto Aimi of *Alium Labs*.

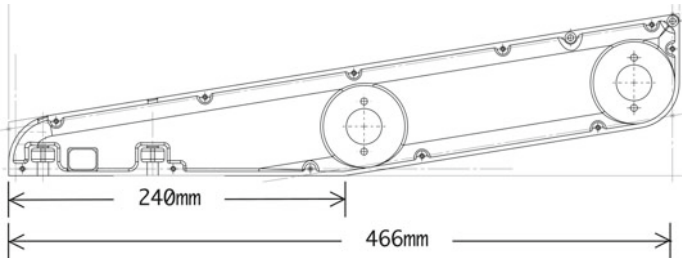


Fig. 2.9 Diagram showing a single mallet control arm, including the locations of the solenoid actuators (crosshairs)

2.3.1.1 Control Architecture

Figure 2.10 shows the overall communication and control structure of the robot. The computer ("PC") separately controls the mallets through the Mallet Motor Control module (MMC), and the Slider Motor Control module (SMC). The MMC uses the Musical Instrument Digital Interface (MIDI) protocol to trigger the solenoid motors. Specifically, the MMC generates MIDI `NOTE_ON` and `NOTE_OFF` commands addressed to each of the 8 mallet rotational solenoids. These commands are demultiplexed by the MIDI Solenoid controller to actuator currents. The SMC uses IAI's proprietary SCON/ASCII serial protocol to specify slider positions and motion trajectories for each of the four linear actuators (sliders).

In its first software version, Shimon did not include perceptual audio processing. Instead, the robot listened to and analysed the music played by humans via the MIDI protocol, which allows for high accuracy processing of both monophonic and polyphonic musical input. Shimon also produced a MIDI score, which then had to be translated to MMC and SMC commands. When a MIDI note comes into play, the software system needs to decide which arm should reach the note and then send the arm into position before commanding the mallet to strike the chosen note. This process is referred to as 'path planning.' In an early version of the software, the marimba's octaves were split into four regions, with each arm responsible to play one of the quadrants.

Two drawbacks of this method were (a) an inevitable delay between activation and note production, hampering synchronous joint musicianship, and (b) this approach doesn't allow for expressive control of gesture-choreography, including tonal and silent gestures. In a second iteration of the control software, we have therefore separated the control for the mallets and the sliders to enable more artistic freedom in the generation of musical and choreographic gestures, without compromising immediacy and safety. We also have enabled more flexibility with regards to which arm can play which notes, and increased the dynamic range of the mallets beyond the binary strike/no strike approach. For more details involving various algorithmic approaches to path planning see Chap. 4.

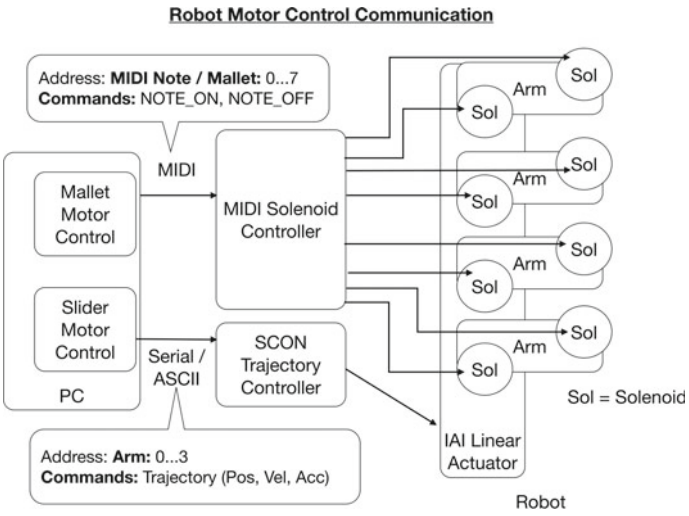


Fig. 2.10 Overall motor control communication diagram of the robot *Shimon*

2.3.2 Mallet Motor Control

As mentioned above, the mallets are struck using rotational solenoids responding to on/off control through a MIDI Solenoid Controller. Eight arbitrary MIDI notes were mapped to the eight mallets, and the MIDI `NOTE_ON` and `NOTE_OFF` messages were used to activate and deactivate the solenoid.

Despite this binary electro-mechanical setup, we still want to be able to achieve a large dynamic range of striking intensities (i.e. soft and loud notes). We also want to be able to strike repeatedly at a high note rate. This is achieved by expanding the Mallet Motor Control module (Fig. 2.11) to include dynamic information. Its new structure enables the translation of a dynamic range of desired intensities for each mallet strike into a specific timing pattern of the MIDI `NOTE_ON` and `NOTE_OFF` commands sent to the MIDI Solenoid controller. We denote a specific sound intensity with the letter p and the duration of solenoid actuation with the letter i. Note that this figure corresponds to the boxes labeled "Mallet Motor Control" and "MIDI Solenoid Controller" in Fig. 2.10.

Solenoids are binary-controlled motors that are either activated or deactivated. In the mechanical setup of a mallet mounted as a weighted lever on a rational solenoid, the mallet head accelerates toward the key as long as the solenoid is activated. It decelerates when the solenoid is deactivated. In addition, if the mallet stays on the key, a muting effect occurs, since the mallet prevents the key from vibrating.

In summary, the striking intensity of a note can be thought of as a function of two parameters: (a) the velocity gained from the acceleration and deceleration toward the key; and (b) the length of time the mallet is held on the marimba key.

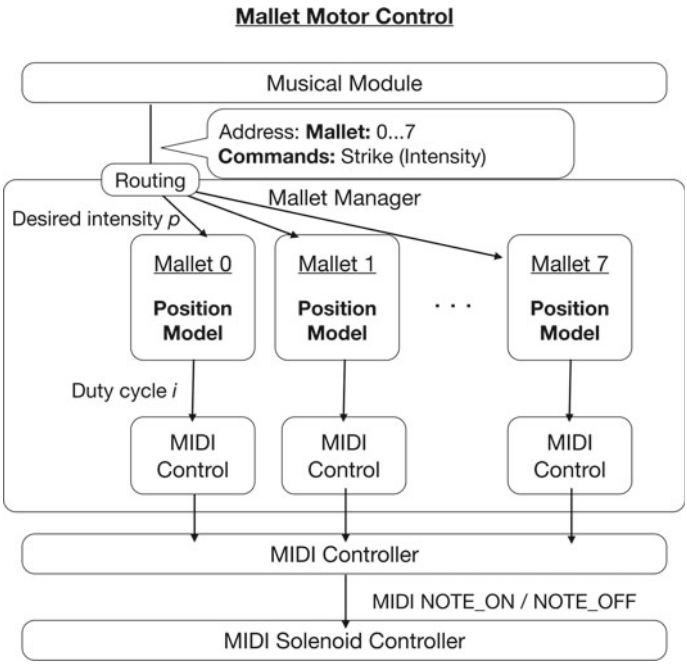


Fig. 2.11 Mallet Motor Control module diagram of the robot *Shimon*

To calculate the appropriate duty cycle for solenoid activation, we therefore need to maintain a model of the mallet dynamics for each striker. Given the nonlinearity of the system and the variability of the mallet heads and solenoid construction, we chose to take a data-driven approach to determine the correct duty cycle per position and mallet. We started by using a microphone to empirically sample sound intensity profiles for different solenoid activation lengths (duty cycle) i, and used those to build a simplified mechano-musical dynamics model for each of the eight mallets (Fig. 2.12). This model includes four parameters:

- i_m—the duty cycle that results in the highest intensity note for that mallet, when it starts from the resting position.
- $d_{\downarrow}$—the mean travel time from the rest position to contact with the key when the solenoid is activated for i_m,
- $d_{\uparrow}$—the mean travel time from the down position back to the rest position when the solenoid is deactivated,
- $d_{\rightarrow}$—the contact duration that results in the highest intensity note for a particular mallet, and

Using this model, each of the eight mallet control modules maintains an estimated position x based on the triggered solenoid commands and the empirical mallet model (Fig. 2.13). During up-travel, $x(t)$, with t being the time since the last mallet activation start, is estimated as

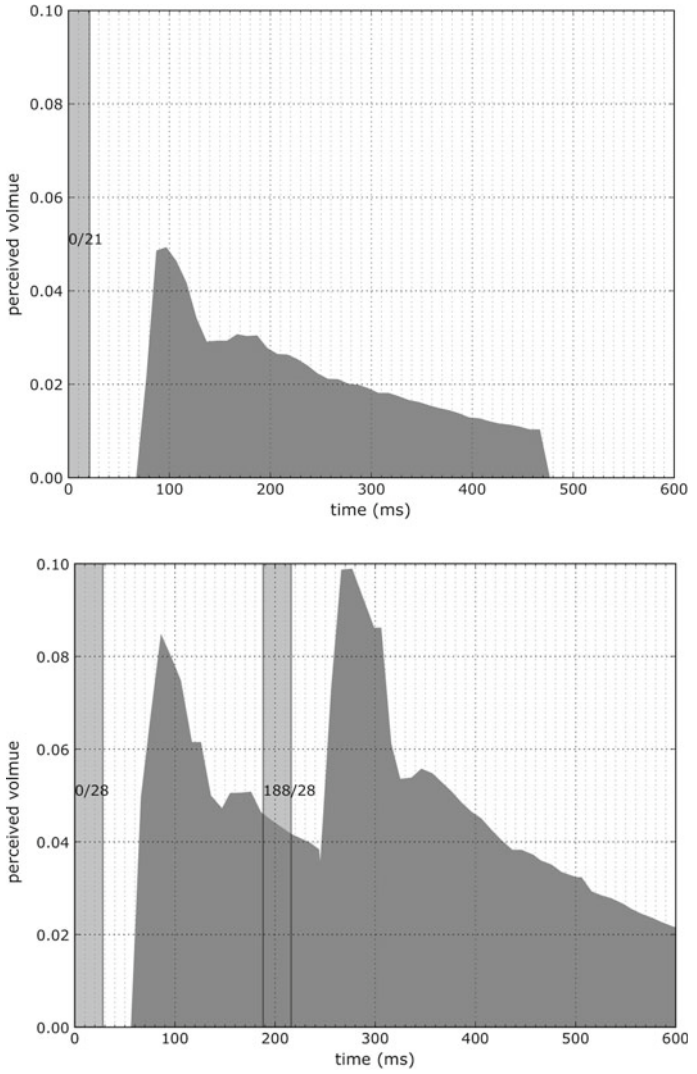


Fig. 2.12 Empirical strike/sound measurements used to build mallet models. We show one example each for single strike measurement to estimate $d_{\downarrow}$, $d_{\rightarrow}$, and i_m (top), and dual strike measurements used to estimate $d_{\uparrow}$ (bottom)

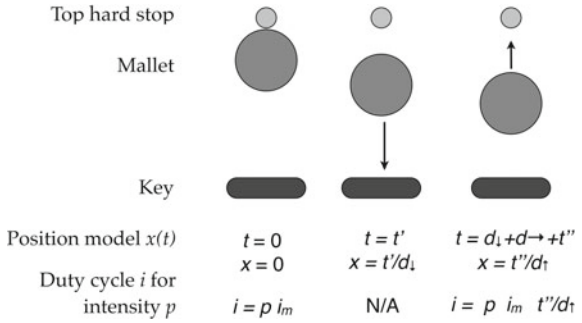


Fig. 2.13 Duty-cycle computation based on mallet position model

$$x(t) = \frac{t - d_{\downarrow} - d_{\rightarrow}}{d_{\uparrow}} \tag{2.1}$$

The MMC translates a combination of desired strike intensity and time of impact into a solenoid duty cycle and trigger time. If the mallet is modeled as being in the upper resting position, the requested duty cycle is a simple linear fraction of the maximum intensity duty cycle. The lower a mallet is at request time, the shorter the duty cycle needs to be to reach impact, and to prevent muting of the key through a prolonged holding time.

As a result, the updated duty cycle i of mallet m as a function of the desired intensity p, is then

$$i = p \times i_m \times x(t) \tag{2.2}$$

In the above equation, we approximate the mallet position as a linear function of travel time. Obviously, a more realistic model would be to take into account the acceleration of the mallet from the resting position to the key impact. Also, bounce-back should be accounted for, for short hold times.

Still, even with this simple linear approximation, the described system results a high level of musical expression, since it (a) maintains a finely adjustable dynamic striking range from soft to loud key strokes, and (b) allows for high-frequency repetitions for the same mallet, during which the mallet does not travel all the way up to the resting position before being re-triggered.

2.3.3 Slider Motor Control

As described above, the horizontally moving sliders were four linear carriages sharing a rail and actuated through voice coil actuators. The control scheme provided by the SCON controller included acceleration- and velocity-limited trapezoid control. This was done by the component labelled "SCON Trajectory Controller" in Fig. 2.10.

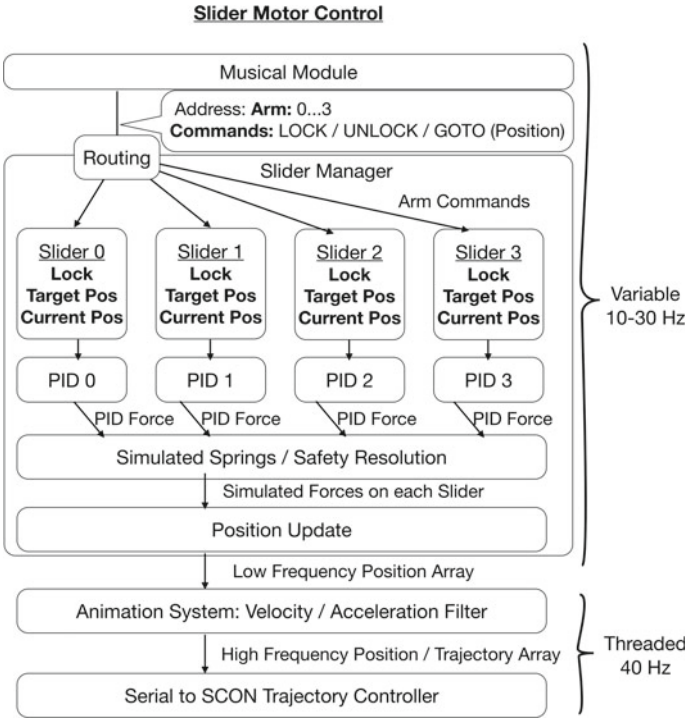


Fig. 2.14 Slider Motor Control module diagram of the robot *Shimon*

In the first version of this control system, as detailed earlier in this section, we found two issues with this control approach. The first was a mechanical ("robotic") quality associated with the standard fire-and-forget motion control approach. The second was collision-avoidance, since all four arms shared one rail.

To tackle these issues, we chose to take an *animation* approach to the gesture control (bottom of Fig. 2.14). Based on our experience with previous robots, e.g. [5–7], we use a high-frequency controller running on a separate program thread, and updating the absolute position of each slider at a given frame rate (in most of our performances we use 40 Hz). This controller was fed position data for all four arms at a lower frequency, based on higher-level movement considerations. It also applied animation-specific velocity and acceleration filters to support expressive movement.

This approach had three main advantages: (a) for each of the robotic arms, we were able to generate a more expressive spatio-temporal trajectory than just a trapezoid, as well as add animation principles such as ease-in, ease-out, anticipation, and follow-through [8]; (b) since the position of the sliders was continuously controlled, collisions could be avoided at the position request level; and (c) movements were smooth at a fixed frequency, freeing higher-level to vary in update frequency due to musical or behavior control considerations, or processing bottlenecks.

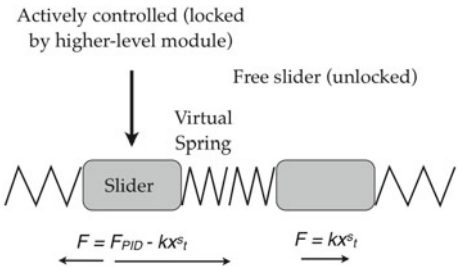


Fig. 2.15 Interaction between PID control and simulated spring model

Feeding the animation subsystem was a layer that handled the musical system's slider position requests and generated the positions for each of the four sliders. This *Slider Manager* also maintained collision safety. It operated by allowing higher-level modules to "lock" a slider, and thus control it for the duration of the locking period.

The Slider Manager then used a combination of proportional-integral-derivative (PID) control for each slider with a simulated spring system between sliders to update the position of all four sliders during each update cycle (Fig. 2.15). The simulated springs were maintained primarily for safety but also resolved collision conflicts with a natural-looking resolution rather than a "robotic" hard stop.

For each position request x_t^r of a locked slider at position x_t at time t, the Slider Manager first calculate the required PID force using the discrete PID formula:

$$F_{PID} = K_p e_t + K_i \sum_{0}^{k} e_{t-i} d\tau + K_d \frac{e_t - e_{t-1}}{d\tau} \tag{2.3}$$

where $d\tau$ is the inverse sampling frequency, and $e_t = x_t^r - x_t$. For sliders that are not locked, the PID force is 0.

In addition to this calculated force, the Slider Manager modeled "virtual springs" on each side of each slider, which help prevent collisions and move unlocked sliders out of the way in a naturally-seeming fashion. Based on the current position of the carriages, the heuristically determined spring constant k, the length of the virtual springs, and thus their current simulated compression x_t^s at time t, we added the spring component kx_t^s to the force. The force update for each carriage was then

$$F_{PID} - kx_t^s \tag{2.4}$$

The result of this control approach is a system that is both safe—carriages will never collide and push each other out of the way—and expressive.

Figure 2.14 shows an summary of the Slider Motor Control module discussed in this section. This diagram corresponds to the boxes labeled "Slider Motor Control" and "SCON Trajectory Controller" in Fig. 2.10.

In sum, higher-level musical modules can "lock" and "unlock" each slider, and can request target positions for each locked slider. These target positions are translated through the PID controller for each slider into virtual forces, which are then

combined for safety in the simulated spring resolver. The combined forces are used to update the target position of each arm. As the temporal resolution of this process is unpredictable and variable in frequency (in our applications, usually between 10–30 Hz, depending on the musical application), these positions are transferred to the animation system, which runs on a separate thread at a fixed frequency (we normally use 40 Hz) and updates the final motor position for each actuator using interpolation with velocity and acceleration limiting. The output positions from the animation controller are transmitted through the SCON ASCII serial protocol to the SCON Trajectory Controller.

While it can normally be assumed that higher-level modules will not cross over carriages, and be generally coordinated, adding this middle-layer control system allows more freedom of expressivity on a higher-level, while still keeping the system safe and expressive at all times.

2.3.4 Shimon's Socially Expressive Head

In order to address the second goal for the project—communicating with co-players using visible socio-musical ancillary gesture—we designed a socially expressive head to provide Shimon with an additional channel of embodied gesture-based communication. Shimon could use the head to communicate internal states, such as rhythm or emotional content and to manage turn-taking and attention between the robot and human musicians.

In specifying the overall design direction of Shimon's head, we decided early on against a humanoid head. In addition, we felt that it would make more sense to match the aesthetics of the arm mechanism by an equally mechanical-looking appearance.

The design process of Shimon's head included five stages: (a) abstract 3D animation exploration to specify expressive degrees-of-freedom; (b) freehand appearance sketches; (c) detailed DOF placement animation explorations; (d) scale models; and (e) final robot solid design and construction.

To evaluate the number of degrees of freedom necessary for the intended musical social expression, as well as their spatial relationship, the design process started with animation explorations using abstract 3D volumes. Using cut-off cylinders and pyramids in different configurations, we attempted a set of expressive gestures, such as responding to the musical beat or shifting attention from one band member to another. For each set of volumes and configurations we evaluated intuitively how readable and expressive the specific configuration played out to be. Our exploration suggested four degrees of freedom: two in the neck (pan/tilt), one tilt below the neck, and one pan at the base of the whole structure.

In parallel to the animation tests, we developed the appearance of the robot through freehand sketch studies. The aim of the sketches was to explore the appearance possibilities unconstrained by mechanical considerations. Most of the designs steered clear of explicit facial features. To compensate for this lack and restore expressive capabilities, we opted for an abstract emotional display in the head. Inspired by the

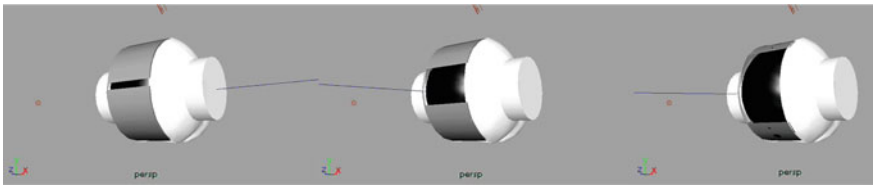


Fig. 2.16 Still frames from animation tests of Shimon head opening mechanism, intended to steer clear from an explicit anthropomorphic face model, but still enabling expressive "facial" expressions

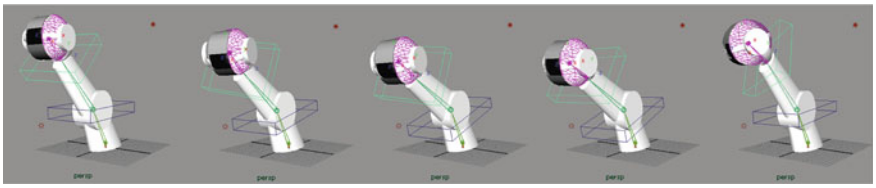


Fig. 2.17 Still frames from detailed animation tests of Shimon head design, exploring DoF placement, hierarchy, and orientation

expressive capabilities of the iris design of AUR [9], an opening mechanism was introduced into the head's design. The idea was that opening and closing a central "space" in the robot could be attached to a range of emotional meanings. We tied this design to the opening and narrowing of eyes and mouth in human faces, associated with a variety of emotional states [10]. It was intentionally left ambiguous whether this opening is more akin to a single eye with eyelids, or a large mouth with opening jaws, in order to not connect a direct anthropomorphic interpretation to the design and leave the interpretation up to the audience.

After the general form was determined, the design process shifted to the next, more involved, step of 3D animation studies. A rough model of the robot was built in a 3D animation program, based on the freehand sketches. Figures 2.16 and 2.17 show still frames from tests of the opening and head motor placement conducted in this stage of the design process.

A notable outcome of this design stage was to use a non-orthogonal angle between the pan and the tilt motors, in combination with a seemingly right angle relationship between the joints reflected in the shell (Fig. 2.18 left and middle). As the pan DoF rotates, the straight neck appears to "break" and create an illusion of a fully articulated 3-DoF joint. This is due to the fact that in the off-right-angle placement, the pan creates both a horizontal and a vertical movement, creating a "sideways-and-up" effect (Fig. 2.18 right). Using 3D animation tests parameterizing the spatial relationships between robot parts, we found that the choice of angle had a significant effect on the character expressed by the robot's movement. After several angle studies, the design settled on a 40° offset between the two joints for a somewhat mischievous personality, befitting a jazz musician. Figure 2.19 shows the fully assembled robot.

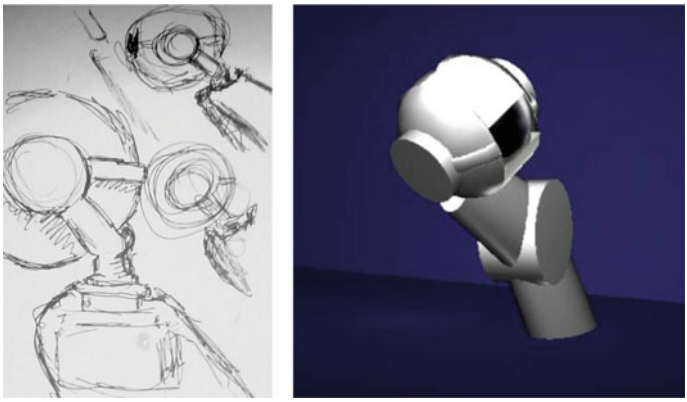


Fig. 2.18 Deceptive placement of pan-tilt mechanism creates a surprising and organically moving joint simulating a fully articulated neck show the resulting characteristic "sideways-and-up" gesture suggesting Shimon's mischief

Fig. 2.19 The socially expressive head of the robot "Shimon", fully assembled and installed by the marimba-playing arms

Shimon's head is placed next to the music-playing part of the robot and is used to signal the robot's internal beat, allowing human musicians to cue their playing to the robot's beat. It also allows making and breaing approximate eye contact, based on fixed band member positions. For example, when the robot takes the lead in an improvisation session, it will turn towards the instrument, and then it will turn back to the human musician to signal that it expects the musician to play next. The head also tracks the currently playing arms by employing a clustering algorithm in

conjunction with a temporal decay of active or striking arms. And finally, two animation mechanisms—an occasional blinking of the shutter and a slow "breathing"-like behavior convey a continuous liveliness of the robot.

2.4 Shimi—A Music Driven Robotic Dancing Companion

Shimi is a dancing social robot that can analyze music and respond with gestures that represent its musical understanding (Fig. 2.20). A smart-phone enabled five degree-of-freedom (DoF) robotic platform, Shimi is a robotic musical companion designed to enhance musical experiences in day-to-day scenarios. Shimi was developed in collaboration with the Media Innovation Lab (miLAB) at the Interdisciplinary Center Herzliya. Unlike Haile and Shimon, Shimi was not designed to generate acoustic sound, but rather respond to digital music played through two speakers embedded in its head. This allowed us to focus on designing music-driven gestures for social communication and interaction.

Fig. 2.20 Shimi is a robotic musical companion that employs the "dumb robot, smart phone" approach and engages users with a variety of musical applications

2.4.1 Robotic Musical Companionship

Haile and Shimon, described in the previous chapters, were first and foremost acoustic instrumentalists. Their interactive experiences were constructed to address the performative aspects of music. A Robotic Musical Companion (RMC) like Shimi, on the other hand, maps onto other scenarios in which people commonly interact with or experience.

For example, music is frequently consumed in the form of general listening through the radio or desktop and mobile applications such as iTunes, Pandora, and Spotify [11–13]. Within and outside of these systems, software exists to recommend music including iGenius, last.FM, and tasteKid [14, 15]. These applications utilize methods involving artificial intelligence or provide interfaces encouraging users to suggest music to friends.

Other musical experiences related to playing instruments outside of the context of performance includes instrumental practice and composition. Musicians often practice with metronomes or in front of peers and educators to receive feedback. Composers often play instruments during the writing process. Additionally, compositional aids in the forms of computer interfaces are becoming more pervasive and even allow novices to develop creative arrangements and compositions [16, 17].

An additional aim of a RMC is to enhance the very experience of listening to music. In this sense, Shimi is part of a larger segment of HRI we can call robotic experience companionship (REC) [18, 19]. This use suggests that people's experience of events can be positively affected by watching a robot co-experience these events jointly with them.

Music can be leveraged to serve additional purposes for non-musical tasks in robotics. For example, a common research area in the field of HRI includes methods for robots to clearly communicate underlying computational and mechanical processes, system performance, and even higher level affective states. Sonifying these pieces of information with music can lead to the development of more expressive and engaging robots. Later we will describe how Shimi uses music to exhibit and communicate emotion, empathy, and comprehension.

In summary, we define an RMC as a physical companion agent which encourages social and personable communications to achieve entertaining musical interactions, while also utilizing music as a tool for typically non-musical tasks. In essence the robot should generate, consume, and demonstrate a knowledge of music.

We acknowledge that existing technologies that may have the same goals as a RMC are easy to use and their interfaces have been optimized for the best user experience. We cannot guarantee that implementing these applications through an HRI will produce a better or worse user experience. Rather, we hope the new types of interaction made available by robotics will pique interest, inspire human creativity, and facilitate tasks that would otherwise be dull, tedious, or difficult through a non-sociable non-embodied computer interface.

2.4.2 Design

At the heart of Shimi's design is an Android smartphone. It thus falls under the umbrella of the "dumb robot, smart phone" paradigm [20]. According to this paradigm, all computation, most sensing, and all the high-level motion planning and control are performed on the mobile device. The rest of the robot's parts deal only with mechanics, additional sensors, and low-level actuator control.

This design approach stems from the observation that personal computing is shifting towards handheld devices characterized by many features of interest to HRI: (a) high-end reliable sensors previously unavailable to lay userscameras, microphones, GPS receivers, accelerometers, gyroscopes, magnetometers, light, and touch sensors; (b) high processing power, comparable to recent notebook computers; (c) a growing number of advanced software libraries, including signal processing modules; (d) continuous internet connectivity through wireless and mobile data networks; and (e) high mobility, due to small weight, small size, and battery power.

Under the constraint of this core design decision, the robot's physical appearance was conceived with a number of guidelines in mind: First, the robot's main application is to deliver music and to move expressively to the music. Its morphology therefore emphasizes audio amplification, and supports expressive movement to musical content. The speakers feature prominently and explicitly in the robot's design and Shimi's head and limb DoFs are placed and shaped for prominent musical gestures. Second, the robot needs to be capable of basic nonverbal communicative behavior, such as turn-taking, attention, and affect display. The robot's head, when placed on a desk, is roughly in line with a person's head when they are seated in front of it.

Finally, the robot's appearance should evoke social presence and empathy with the human user. Its body is sized and shaped to evoke a pet-like relation, with size comparable to a small animal, and a generally organic, but not humanoid form.

When designing a smartphone-based robot, an inevitable design decision is the integration of the mobile device within the overall morphology of the robot. Past projects have opted to integrate the device as either the head or the face of the robot, placing it on a pan-tilt neck [21] or behind a shell with an opening cut for the screen [22, 23].

In contrast, Shimi "holds" the device, and is connected to it through a headphone cable running to its head. This is intended to make the robot be more similar to how a human uses a smartphone, and possibly create empathy with the robot. Moreover, this setup allows for the device to serve as an object of common ground and joint attention between the human and the robot. The robot can turn the phone's front screen towards its head and towards the human discussion partner. In our current application, for example, we use a gaze gesture as a nonverbal grounding acknowledgment that the device was correctly docked.

Overall, we used an iterative industrial/animation/mechanical design process, similar to the one used for the design of Shimon. This process integrates requirements for appearance (industrial design), motion expressiveness (animation), and physical constraints (mechanical design).

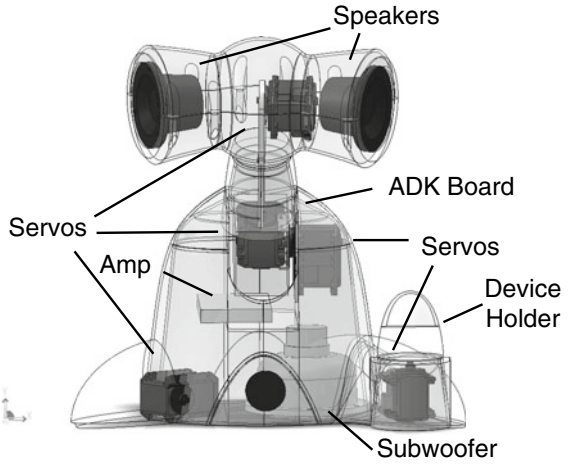


Fig. 2.21 *Shimi* mechanical structure

The resulting design consists of a five degree-of-freedom robot with one DoF driving the device-holding hand pan, one driving the foot tap, and three degrees of freedom in the neck, set up as a tilt-pan-tilt chain. Each DoF is controlled via direct-drive with motors daisy-chained through a TTL network. The robot has two speakers, acting as a stereo pair, in the sides of its head, and one subwoofer speaker pointing downwards in the base. In addition, the robot contains an ADK/Arduino control board, and a digital amplifier with an audio crossover circuit (Fig. 2.21).

2.4.3 Software Architecture

The robot's system can be divided into two parts (Fig. 2.22): all software, including high-level motor control is performed on the smartphone, in the form of a single mobile application. This application communicates over USB using the Android Debug Bridge (ADB) protocol with the ADK board. The device also transmits analog audio to the amplifier in the robot's body. The mobile device software's interface to the ADK board is the *Motor Controller* module, using a low-latency position-velocity packet protocol, with packets sent at variable intervals. The board runs a simple firmware acting as a bridge between the ADB interface and the motor network protocol. It forwards the position-velocity commands coming in on the USB port to the TTL bus. Each motor maintains its own feedback, position control, and velocity limit through the servo firmware of the motor unit.

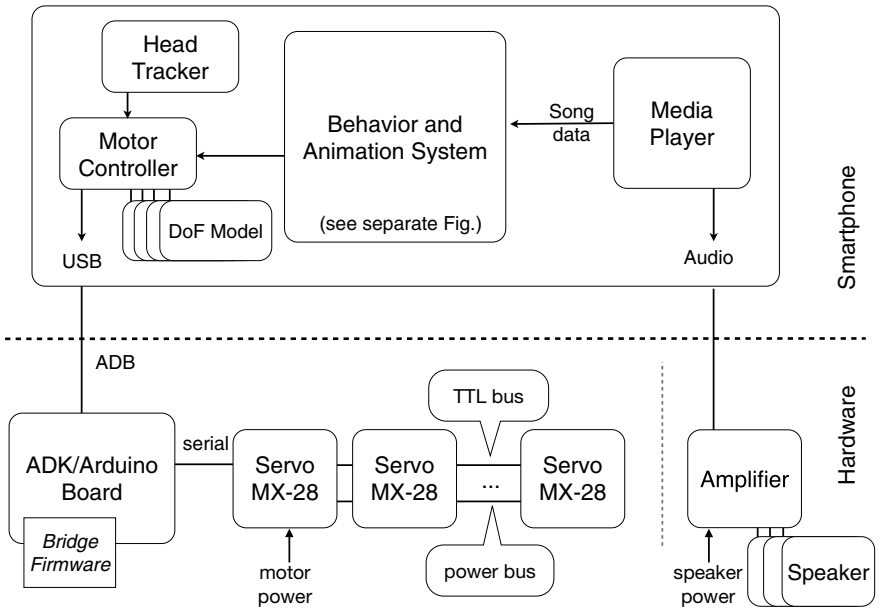


Fig. 2.22 *Shimi* system diagram

2.4.4 Core Capabilities

Shimi's core capabilities, useful for any application include being able to generate expressive gestures, syncing movements to specific beat events, and making eye contact. Figure 2.23 shows an overview of the robot's core system software that enables these behaviors. This core architecture was then adapted for each specific application.

The building blocks of the expressive behavior system are *Behavior* modeled as movement responses to real-time song beats. The *Behavior Controller* receives the current song's metadata—its genre, tempo, and duration—from the device's media player, and manages the launching and aborting of the robot's various Behaviors. When no song is playing, a default "breathing" Behavior indicates that the robot is active and awaiting input from the user (see: [5]). As the song is playing, a *Beat and Segment Tracker* module follows the progress of the song by the Media Player, and triggers callback events to the behavior subsystems of the robot. In case of a segment change, the Tracker calls back the Behavior Controller, causing it to select the next appropriate Behavior based on the genre and segment. For beats, we have currently implemented two kinds of Tracker modules, one fixed-interval module that detects the first beat, and then triggers beats at fixed intervals. This is usually appropriate for electronically generated music files. The second module uses variable intervals read from a beat data file generated by prior beat analysis.

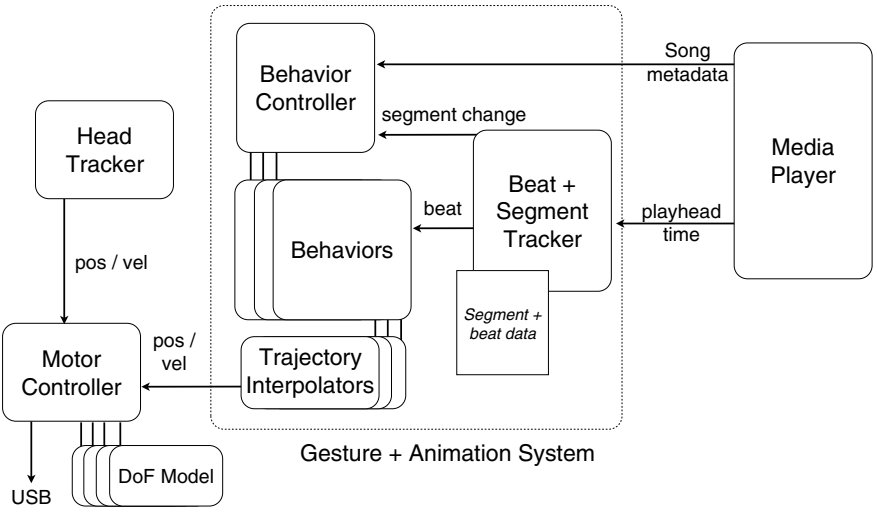


Fig. 2.23 *Shimi* core capabilities software library

In case of a beat trigger, the Tracker calls the currently running Behavior to execute one of two beat responses: (a) a repetitive beat gesture involving one or more DoFs; or (b) a probabilistic adjustment gesture, adding variability to the repetitive motion. Each motion is then split by DoF and sent to the Trajectory Interpolator associated with the DoF described below.

2.4.4.1 Syncing to a Beat

Shimi often responds to a sync signal or beat by doing a repetitive back-and-forth gesture (e.g. "head banging", "foot tapping", etc). For this gesture to appear on beat, the robot has to perform the direction change very close to the audible occurrence of the beat, as human observers are extremely sensitive to the timing of the trajectory reversal. This planning challenge is exacerbated when beats are not at perfectly regular intervals.

We address this challenge with an *overshoot and interrupt* approach, scheduling each segment of the repetitive movement for a longer time period than expected, and ending the motion not with a zero velocity, but with a slow continued trajectory to a point beyond the target. The following beat then interrupts the outgoing trajectory on sync with the returning trajectory command. Since the exact spatial position of the beat event is not crucial, "overshoot-and-interrupt" allows for a continuous and on-beat repetitive gesture. The robot seemingly reaches the end of its motion precisely on beat, simply by reversing course at that moment.[1]

[1]Thanks to Marek Michaelowski for pointing out this last insight.

2.4.4.2 Smoothing the Motion Trajectory

Within each gesture segment, we aim to achieve life-like, expressive motion. Traditional and computer animation uses trajectory edge-damping to achieve less mechanical seeming movement, a technique called ease-in and ease-out. While easily accomplished through acceleration-limited motor control, many lower-end servo motors, such as the ones used in the design of Shimi, specify movement only in terms of goal position and velocity. In addition, to optimize bandwidth on the servo's half duplex architecture, we also rely on dead-reckoning, without polling the motors for their accurate position.

To simulate ease-in/ease-out given these constraints, we use a high-frequency interpolation system, inspired by the animation arbitration system used in [24], and similar to the one used in Shimon [25]. A *Trajectory Interpolator* per DoF receives target positions and maximal velocities from the Behavior layer, and renders the motion through a high-frequency (50 Hz) interpolator. The closer the motion is to the edge of the movement, the slower the commanded velocity of the motor. Periodic velocity v' is expressed as a positive fraction of goal velocity v:

$$v' = v \times (2 \times (1 - \frac{|t - d/2|}{d}) - 1)$$

where t is the time that passed since the start of the movement and d is the planned duration of the movement.

An opportune side-effect of this approach is that the duration compensation from the original linear motion trajectory causes the movement to take slightly longer than the single or half beat of the gesture. This enables the use of the overshoot-and-interrupt approach described above, resulting in precise beat timing. The combination of both methods results in continuous, life-like, beat-synchronized gestures.

2.4.4.3 Eye Contact

Gaze behavior is central to interaction [26]. Shimi makes eye-contact by using the built-in camera of the mobile device to capture the scene in front of it. We then make use of existing face detection software on the phone to track and follow the user's head.

Since the phone is mounted on a pan DoF, linear compensation feedback will keep the head centered in the camera view. Given a high enough face detection frame rate, and continuous user motion, we move the device-holding hand according to

$$p' = p + \lambda(x - \frac{w}{2})$$

with p being the current motor position, x being the face detection center of mass, w the image width, and λ the tracking factor. A higher value for λ results in more responsive, but also more jittery, tracking.

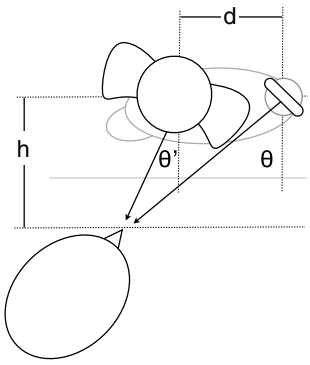


Fig. 2.24 Active perception tracking with the head following the camera-holding hand, compensating for parallax

As the mobile device, and thus the camera, is coupled to the robot's hand, gaze behavior requires an additional transformation of the hand rotation to the head pan coordinates. Coupling the neck pan DoF angle θ' to the active perception result angle θ, the robot compensates for parallax induced by the disparity d between the two DoF centers (Fig. 2.24). h is the estimated frontal distance of the human's head:

$$\theta' = arctan(tan\theta - \frac{d}{h})$$

2.5 The Robotic Drumming Prosthetic

GTCMT's Robotic Drumming Prosthetic Arm (RDPA), which was originally developed as a wearable device for amputees, uses modeling of human drumming techniques to support higher levels of dexterity and music expression than previously possible. Before developing the RDPA, research on robotic percussionists at GTCMT focused on the development of musical analysis and improvisational algorithms and the design of robots that respond in a contextually appropriate way to music and methods to utilize gestures in collaborative music making. These research directions addressed the question of "what" is being played—what are the notes, dynamics, inter-onset times and the rhythmic parameters that a robot can play in response to the musical analysis. The RDPA allowed us for the first time to focus more on "how" each of these notes are played, how to enrich robotic timbral expression and how to create human-like musical expression, which has been a long standing goal for research in robotic musicianship.

To achieve this goal we developed a generative model for stroke generation based on physical modeling of the interaction between a human hand and a drumstick. By varying one parameter—the time-varying torque applied by the thumb on the stick—our model can generated a wide palette of strokes such as multiple bounce strokes, double, and triple strokes. We have also developed a number of post processing tools

to further enrich the strokes produced by the physical model. This allowed us to simulate and capture the acoustic variety and richness of natural human drumming.

2.5.1 Motivation

The RDPA was designed to enhance the drumming capabilities and musical expression in robotic percussionists. There are different manners one can control musical expression in drumming. Variations in timbre, stroke types, and activity rates all contribute in a significant way to the richness, variability, and expression of rhythmic phrases. When human drummers learn the language of their instrument and the idiosyncrasies of musical styles, the question of "how" to play each note becomes critical. In an effort to approach the question of "how" a stroke is execute, we studied how humans control the stick motion, and implemented the model in the RDPA robotic platform. Our approach was to develop a generative model that captures the physics of a drum stroke (focusing on multiple bounce strokes) as performed by human drummers and develop the appropriate control design to implement the model in a robotic drummer. It is important to note that, the model was not aimed at creating an exact replication of multiple bounce strokes played by a human, but to create a similar perceptual effect for the listener.

There are two main types of drumming technique rudiments: 1. rudiments that control the exact number of bounces after a primary stroke (for example, single strokes, double strokes, triple strokes) and 2. rudiments that control the statistical aspects of the stroke such as density of the bounces (for example, closed roll) [27]. More sophisticated bounce behavior such as generating multiple bounce strokes in which the frequency of bounces change over time, is often for finer musical expressions and ornamentation in soloing and jazz accompaniment.

The first rudiment type can be implemented in a robotic drummer by specifying the number of "single" hits that need to be put together to produce the rudiment, which does not require sophisticated modeling. But in order to achieve less specific and more free flow playing experience that is captures by the second rudiment, a generative model, such as the one developed in the work, is required. Moreover, our model developed can also be used to produce atypical drumming behaviors in robotic drummers by harnessing the speed and computational capabilities of a computer. For example, due to the effect of gravity, a typical multiple bounce stroke by a human drummer has an exponentially decaying amplitude envelope. Using our proposed model, it cam be possible to model a system in which the gravity is changing over time, creating humanly impossible decay envelopes for the strokes, and pushing the percussive sound palette to uncharted territories.

2.5.2 Related Work

One of the most common actuator mechanisms for robotic percussionists is a motor/solenoid system, mounted on the rim of a drum, which strikes a drum membrane using a stick or mallet much like Shimon [28]. In such solenoid based systems the force of the drum strike is proportional to the voltage that is applied to the solenoid. In a MIDI triggered solenoid system it is common for every drum strike to correspond to a single MIDI message, meaning, a multiple bounce stroke can only be accomplished by sending repeated strike messages. In comparison, humans execute a multiple bounce stroke by manipulating other parameters such as the initial velocity and position of the stick and the grip pressure. Therefore, a solenoid-based system is not an effective approach if we are to simulate the natural manner in which human drummers operate.

To address this challenge, studying the literature about the mechanics of human drumming from an engineering and bio-physical standpoint can be helpful. Hajian et al. have studied the relationship between grasp force and the stick bounce and have implemented a mass-spring model which explains the dynamics of a double stroke drum roll in a simple, single joint robot which uses pneumatic actuators [29]. Another approach is to model the bounce as a restoring force that acts on the drumstick upon contact. Berdahl et al. followed this approach, using a lumped mass spring damper model to model the effect of fingers [30]. However, the physical modeling approaches adopted by Hajian and Berdahl only addresses closed double stroke roll, and cannot be generalized to produce different kinds of strokes.

A different approach for modeling the dynamics of the interaction between the drumstick, drumhead and the fingers was explored by Kim et al. [31]. Here, a coupled mass-spring damper model led to the design of a variable stiffness actuator that executed robotic drum strokes, allowing for different ranges of stiffness that correspond to single and double strokes. However, this project did not address multiple bounce strokes and led to results that did not resemble human drumming where the second hit in the double stroke was much lower than the first hit. In our work we attempted to overcome these issues by modeling the force exerted by the finger on the stick in such a manner that would allow for the amplitude of hits to remain consistent.

Other studies focused on modeling but did not attempt implement the results in a robotic percussionist. For example, For Wagner analyzed the interaction between the human hand, drum stick and the membrane, coming to the conclusion that the dominant factors that shape the interaction process between a drumstick and the drumhead are the deflection and the tension of the drumhead and the vibration of the drumstick [32]. Wanger's model assumes that the human drummer determine the initial onset energy (affecting the dynamic level) and the placement of the stroke (affecting the modes excited in the drum), but does not address the influence of changing grip.

Physics based concepts have also been utilized in robotic musicianship systems. For example, in [33] a dynamical systems approach enabling switching between discrete and rhythmic modes was adopted for online generation of drumming motions

for humanoid robots. The system was generated trajectories for an arbitrary drum score, although it was not evaluated by listeners.

Williamson et al. designed a compliant robot arm with six degrees of freedom and uses simple non-linear oscillators to control the arm [34]. The arms used series elastic actuators which incorporate physical springs at each joint. This project differed from traditional robot control in the manner in which robot dynamics play a crucial role in the execution of the task. Willamson's approach is limited to the trajectories that are generated by the oscillators in comparison to the more general traditional approach.

2.5.3 Platform

Informed by related work and our own previous work on Haile and Shimon, with the help of Meka Robotics we designed a robotic drumming arm that used small brushless gimbal motors, exhibiting a low kV rating. The motors provided higher torques at low speeds compared to standard brushless motors which require significant gear reduction for usable output torque and speeds. The gimbal motors also provided rapid acceleration, allowing for hitting rate of up to 20 Hz. A single stage timing belt drive was chosen for the device, serving compliant element between the output and the motors and capable of withstand the repeated shock loading common in drumming. The belt drive allowed the motors—the heaviest components of the system—to be placed close to the elbow, thereby, reducing fatigue and serving as a mechanical fuse. The final gear ratio of our device is less than 3:1.

In an effort to reduce the weight of the device (the main concern for usability), the primary structure of the RDPA was a unibody frame machined from a single billet of aluminum. Diagonal supports were cut into the frame, removing as much material as possible without sacrificing structural rigidity. The frame had an i-beam profile that is closed on each end for stiffness. The control electronics were integrated into the main structure, while cutouts along the top and bottom gave access to connectors for sensors, power, and control bus. The motors were located in a custom sheet metal enclosure at the back of the main structure, tucked underneath the carbon fiber socket and slightly offset downwards. High resolution optical encoders mounted to the rear of the motors allowed for advanced high precision motor control. The boards were equipped with analog inputs for taking input from various sensors such as EMG and potentiometers. For modeling the PI controller and position tracking we used *MATLAB/Simulink*. Optical encoders mounted on the rear of the motors were used in an effort to achieve high precision motor control. The proprietary main microprocessor boards, designed by Meka, were mounted directly in the main structure and the communication between the host computer and the board was conducted through a high speed *EtherCat* control bus (Fig. 2.25).

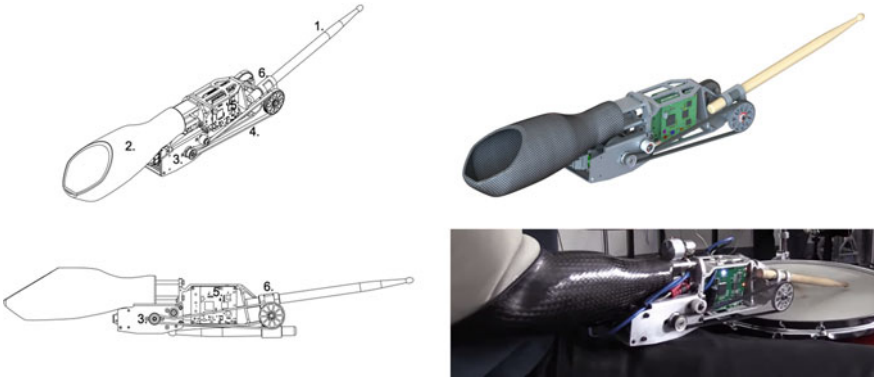


Fig. 2.25 **1.** Drum stick, **2.** socket, **3.** casing for motors, **4.** belt connecting motor to drum stick mount, **5.** processor, **6.** drum stick mount

2.5.4 Generative Physical Model for Stroke Generation

In order to generate a variety of drum strokes such as single hit, double hit and multiple bounces, our system is designed to control the time-varying torque provided by the thumb as a tunable parameter in a physical model. This allows the device to hit the drum in a variety of manners in an attempt to recreate the expression and richness of human drumming. Since the variations in multiple bounce strokes are typically controlled by using smaller fingers and wrist muscle groups, we focused on modeling palm and finger control, rather than addressing other degrees of freedom such as the elbow and the wrist.

2.5.4.1 Simplified Model of Drumming Mechanics

For simplification, we treat the drumstick as a uniform rigid rod rotating about a fixed pivot point. When the stick is in rotational equilibrium about the fulcrum, the torque is balanced by a counter torque that provided by the backside of the thumb (Fig. 2.26). In order to produce a multiple bounce stroke, drummers typically release the stick by opening up the thumb and allowing the stick drop. Upon hitting the drumhead, the thumb and other fingers are used to provide a time varying torque to control the after bounce timbre.

The following time varying torque model provides a physics-based framework to support multiple bounce strokes generation, identifying a number of stroke categories based on the general shape of the torque profiles:

- No External Torque (Only Gravity)—In which there is no torque due to the thumb and the bounce behavior is similar to that of a bouncing ball.
- Constant External Torque—In which the behavior is similar to that of a bouncing ball, but as if in an environment with stronger gravitational force.

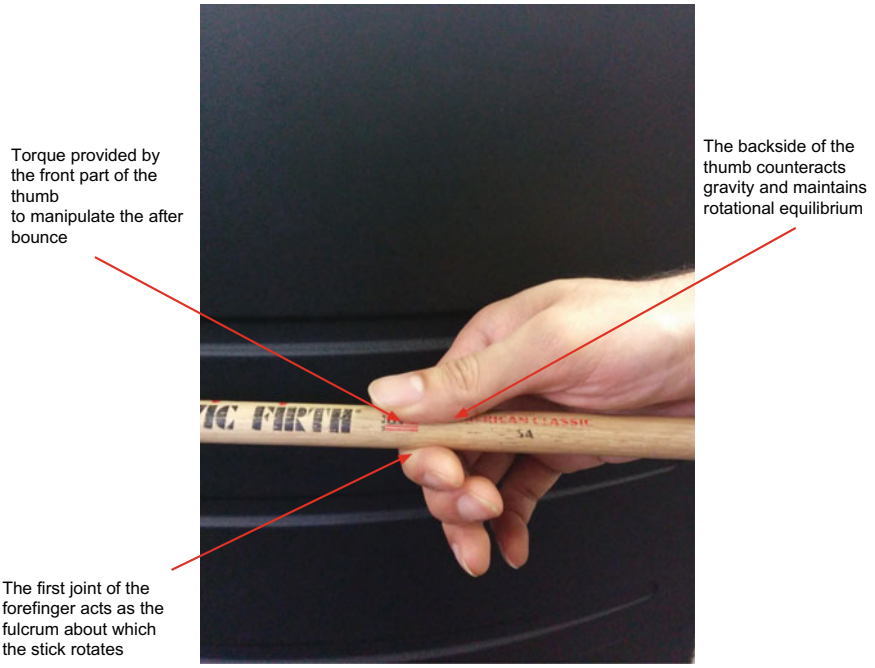


Fig. 2.26 Stick in rotational equilibrium

- Low to High variation in torque—In which the initial onsets are widely spaced with a sudden increase in the frequency of onsets.
- High to Low variation in torque—In which the initial onsets are closer together followed by a sudden decrease in the frequency of onsets.
- Un-natural variations in torque—In which the profiles are extremely hard to produce by human drummers, but can be possible in a physics simulation, thereby giving rise to strokes that can go beyond human capabilities.

2.5.4.2 Derivation of the Dynamical Equation

In Fig. 2.27, let L be the length of the drumstick, x be the distance of the fulcrum from the tip and m be the mass of the drumstick. θ is the angular displacement of the stick and $\theta = 0$ when the stick is parallel to the drumhead. As the stick moves downwards towards the drum, θ decreases (downward direction is taken as negative). The moment of inertia I is given by

$$I = \frac{m}{3L}(x^3 - (x - L)^3) = \frac{m}{3}(3x^2 - 3xL + L^2) \tag{2.5}$$

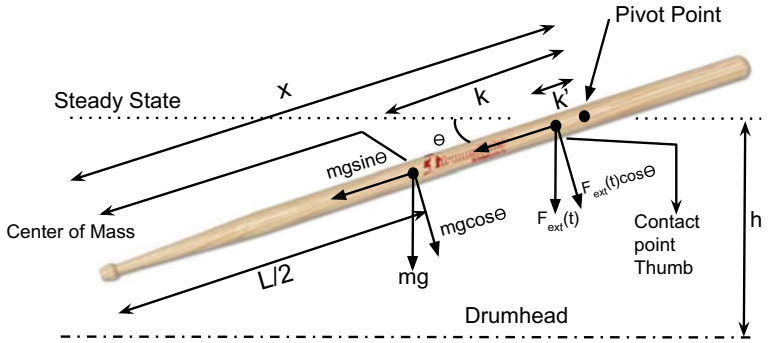


Fig. 2.27 Detailed torque diagram

We treat the drumstick as a freely rotating rod about a pivot point that is at a distance of x from the tip of the stick. Gravity acts at the center of mass which is located at the midpoint of the stick and is at a distance of k from the fulcrum. The thumb is in contact with the stick at a distance of k' from the fulcrum.

The total torque(τ_{total}) acting on the drumstick can be described as

$$\tau_{total} = \tau_{gravity} + \tau_{ext}(t)$$

where $\tau_{gravity}$ is the torque due to gravity and the external torque τ_{ext} is a function of time t and can have different varieties of profiles. Furthermore,

where, $F_{ext}(t) = ma_{ext}(t)$ and $a_{ext}(t)$ is the linear acceleration provided by the thumb. From Newton's laws we have

$$\tau_{total} = I\ddot{\theta}$$

where $\ddot{\theta}$ is the angular acceleration of the stick. Then,

$$I\ddot{\theta} = \tau_{gravity} + \tau_{ext}(t) \tag{2.6}$$

$$I\ddot{\theta} = -mgcos(\theta)k - ma_{ext}(t)cos(\theta)k' \tag{2.7}$$

Substituting Eq. 2.5 in Eq. 2.7 and rearranging the terms we get

$$\ddot{\theta} = -\frac{3cos(\theta)(gk + a_{ext}(t)k')}{3x^2 - 3xL + L^2} \tag{2.8}$$

Since the geometrical relationship between the pivot point, the stick and the drumhead is fixed, the vertical distance of the pivot point from the drumhead (h) is constant. The maximum angular displacement possible from h and x from steady state position can be calculated as

$$\theta_{max} = -sin^{-1}\left(\frac{h}{x}\right) \tag{2.9}$$

In order to incorporate collisions of the stick with the drum, two approaches can be adopted.

$$\tau_{gravity} = -mgcos(\theta)k$$
$$\tau_{ext}(t) = -F_{ext}(t)cos(\theta)k'$$

In the first approach, we approximate the forces acting upon the stick at the moment of contact and modify dynamical equation for a short duration the. We can summarize the effect of all the contact forces treat the result as an inelastic collision. Due to the damping of the drumhead and loss of energy, the sound produced, and the air resistance, the collision is inelastic. Therefore, when $\theta = \theta_{max}$ the velocity undergoes an instantaneous change in direction and magnitude given by,

$$\dot{\theta}_{after} = -c\dot{\theta}_{before} \tag{2.10}$$

where c is the coefficient of restitution (COR) between the drum stick and the drumhead. The parameter c can be computed explicitly if the mass of the drum stick, damping and spring constants of the drum head are known as in [30].

For this project we chose a simpler approach where we determine the coefficient of restitution experimentally. By modeling the drumstick as a bouncing ball falling due to gravity, we can compute the coefficient of restitution (COR) by comparing the maximum height the stick reaches after the first bounce to its initial height when released (disregarding friction at the fulcrum and air resistance). Typically, this value will be high for taut drum heads with high pitch, whereas for low pitched drums, the value will be lower.

The stick used for the study was a *Zildjian Maple Mini-ball*.[2] The approximate values of its variables are presented in Table 2.1. Substituting the parameter values from Table 2.1 in Eq. 2.8 with acceleration due to gravity $g = 9.8$ m/s^2 will lead to the following equation:

$$\ddot{\theta} = -\frac{cos(\theta) \times (0.735 + (0.01 \times a_{ext}(t)))}{0.0714} \tag{2.11}$$

This is a second order differential equation (with $a_{ext}(t)$ as the time varying input to the system) which can be solved using standard numerical methods. The external linear acceleration $a_{ext}(t)$ is in the range 0–250 m/s^2. For a stick with $m = 0.0373$ kg this corresponds to a linear force of approximately in the range of 0–10 N. This is the characteristic force range that fingers can produce during the interaction with a drumstick while performing a drum stroke [35].

[2] http://zildjian.com/Products/Drumsticks-and-Mallets/Maple-Series/Maple-Mini-Ball.

Table 2.1 Parameters values for Zildjian Maple Miniball stick

Parameter	Value
L	0.39 m
x	0.27 m
k	0.075 m
k'	0.01 m
h	0.07 m
θ_{max}	−0.2623 rad
vol	6.23×10^{-5} m^3
ρ	600 kg/m^3
m	0.0373 kg

2.5.4.3 Implementation of Stroke Profiles in the RDPA

The RDPA uses a PI controller to perform reference trajectory tracking, using motor angle trajectories computed by generative physical model described in the previous subsection. *MATLAB/Simulink* is used to simulate the various multiple bounce stroke profiles based on our physical model of the different stroke profiles. The profiles are stored as text files on the host computer which communicates with the RDP. At runtime, the stroke files are loaded into memory and accessed depending on the musical application. Musical compositions and time based sequences are generated using custom C++ code which utilize the stroke profiles as building blocks. Our system implementation is overviewed in Fig. 2.28.

To achieve minimum error between actual trajectory and the reference, the PI controller has to be well tuned. Since the diversity of simulated strokes is very high, we did not use formal methods of tuning the controller rather determined the exact values of the gains manually by trial and error. We did fix the controller gains w for all the different stroke profiles to provide some level of automation.

2.5.4.4 Creativity in Model Parameter Tuning

Since there is no one "correct" way to play a series of multiple bounce strokes, we used the system to explore different strokes in an effort to generate novel creative outcome. By changing the input acceleration profiles in a continuous manner, infinitely many stroke profiles can be generated. Moreover, by setting the model parameters to non-standard values, unnatural stroke profiles can be generated, which will then enhance the stroke palette available for composers. Furthermore, the stroke profiles can be post processed to create artificial variations that cannot be achieved in a natural way. For example, through time stretching, compression (changes the dynamic level) and time reversal, novel humanly impossible rhythmic outcomes can be created. A more detailed explanation of the musical applications of the work can be found in [36].

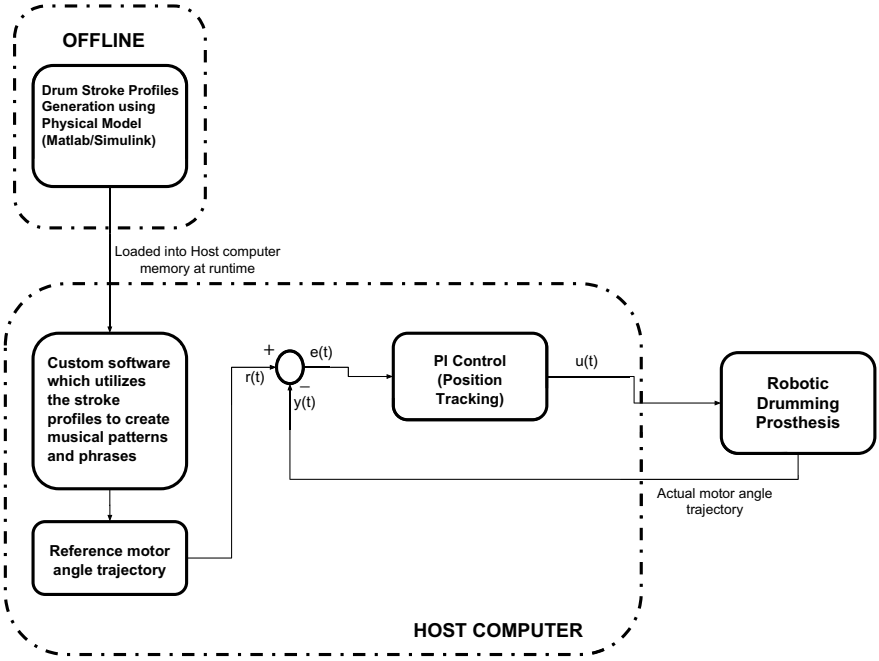


Fig. 2.28 Implementation framework

2.5.5 Conclusions

Here, we presented a generalized physical model for multiple bounce drum stroke generation with the goal of expanding the stroke palette for robotic drummers. This marks an important step towards achieving human-like musical expression for robotic drummer, as well as expanding the pallet towards humanly impossible drumming. In Chap. 7 we describe the RDPA in more detail and describe a user study in which an amputee actually uses the device.

References

1. Weinberg, Gil, Scott Driscoll, and R. Mitchell Parry. 2005. Haile-an interactive robotic percussionist. In *ICMC*.
2. Jordà, Sergi. 2002. Fmol: Toward user-friendly, sophisticated new musical instruments. *Computer Music Journal* 26 (3): 23–39.
3. Maes, Laura, Godfried-Willem Raes, and Troy Rogers. 2011. The man and machine robot orchestra at logos. *Computer Music Journal* 35 (4): 28–48.
4. Stanley, Coren, Lawrence M. Ward, and Clare Porac. 1989. *Sensation & perception*. Harcourt Brace Jovanovich.

5. Hoffman, Guy, Roni Rony Kubat, and Cynthia Breazeal. 2008. A hybrid control system for puppeterring a live robotic stage actor. In *Proceedings of the 17th IEEE international symposium on robot and human interactive communication (RO-MAN 2008)*.
6. Hoffman, Guy, and Cynthia Breazeal. Collaboration in human-robot teams. In *Proceedings of the AIAA 1st intelligent systems technical conference*, Chicago, IL, USA, Sept 2004. AIAA.
7. Hoffman, Guy, and Ju Wendy. 2014. Designing robots with movement in mind. *Journal of Human-Robot Interaction* 3 (1): 89.
8. Lasseter, John. 1987. Principles of traditional animation applied to 3D computer animation. *Computer Graphics* 21 (4): 35–44.
9. Hoffman, Guy, and Cynthia Breazeal. 2009. Effects of anticipatory perceptual simulation on practiced human-robot tasks. *Autonomous Robots* 28 (4): 403–423.
10. Ekman, P., and W.V. Friesen. 1969. The repertoire of nonverbal behavior: Categories, origins, usage, and coding. *Semiotica* 1 (1): 49–98.
11. Apple. 2014. http://www.apple.com/itunes/.
12. Pandora. 2014. http://www.pandora.com/.
13. Spotify. 2014. https://www.spotify.com/us/.
14. Last.fm. 2014. http://www.last.fm/.
15. Tastekid. 2014. http://www.tastekid.com/.
16. Dahlstedt, Palle, and Peter McBurney. 2006. Musical agents: Toward computer-aided music composition using autonomous software agents. *Leonardo* 39 (5): 469–470.
17. Rao, Visarapul. 2013. Columbia: Music composition aid (version 1.2) [mobile application software]. https://itunes.apple.com/us/app/columbia-music-composition/id676546538?mt=8.
18. Hoffman, Guy, and Keinan Vanunu. 2013. Effects of robotic companionship on music enjoyment and agent perception. In *2013 8th ACM/IEEE international conference on human-robot interaction (HRI)*, 317–324. IEEE.
19. Hoffman, Guy, Shira Bauman, and Keinan Vanunu. 2016. Robotic experience companionship in music listening and video watching. *Personal and Ubiquitous Computing* 20 (1): 51–63.
20. Hoffman, Guy. 2012. Dumb robots, smart phones: A case study of music listening companionship. In *2012 IEEE, RO-MAN*, 358–363. IEEE.
21. Adalgeirsson, Sigurdur O., and Cynthia Breazeal. Mebot: A robotic platform for socially embodied presence. In *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, 15–22. IEEE Press.
22. Short, Elaine, Katelyn Swift-Spong, Jillian Greczek, Aditi Ramachandran, Alexandru Litoiu, Elena Corina Grigore, David Feil-Seifer, Samuel Shuster, Jin Joo Lee, Shaobo Huang, et al. 2014. How to train your dragonbot: Socially assistive robots for teaching children about nutrition through play. In *The 23rd IEEE international symposium on robot and human interactive communication*, 924–929. IEEE.
23. Wistort, Ryan, and Cynthia Breazeal. 2009. Tofu: A socially expressive robot character for child interaction. In *Proceedings of the 8th international conference on interaction design and children*, 292–293. ACM.
24. Gray, Jesse, Guy Hoffman, Sigurdur Orn Adalgeirsson, Matt Berlin, and Cynthia Breazeal. 2010. Expressive, interactive robots: Tools, techniques, and insights based on collaborations. In *HRI 2010 Workshop: What do collaborations with the arts have to say about HRI*, 21–28.
25. Hoffman, Guy, and Gil Weinberg. 2011. Interactive improvisation with a robotic marimba player. *Autonomous Robots* 31 (2–3): 133–153.
26. Argyle, Michael, Roger Ingham, Florisse Alkema, and Margaret McCallin. 1973. The different functions of gaze. *Semiotica* 7 (1): 19–32.
27. Rich, Buddy, and Henry Adler. 2005. *Buddy Rich's modern interpretation of snare drum rudiments*. Amsco Music.
28. Kapur, Ajay, E. Singer Trimpin, Afzal Suleman, and George Tzanetakis. 2007. A comparison of solenoid-based strategies for robotic drumming. In *ICMC*, Copenhagen, Denmark
29. Hajian, Aram Z., Daniel S. Sanchez, and Robert D. Howe. 1997. Drum roll: Increasing bandwidth through passive impedance modulation. In *Proceedings of IEEE international conference on robotics and automation*, vol. 3.

30. Berdahl, Edgar, Bill Verplank, Julius O. Smith, and Günter Niemeyer. A physically-intuitive haptic drumstick. In *Proceedings of the international computer music conference*.
31. Kim, Young Gil, Manolo Garabini, Jaeheung Park, and Antonio Bicchi. Drum stroke variation using variable stiffness actuators. In *IEEE international conference on intelligent robots and systems (IROS)*.
32. Wagner, Andreas. 2006. Analysis of drumbeats–interaction between drummer, drumstick and instrument. Master's Thesis, KTH Computer Science and Communication.
33. Degallier, Sarah, Cristina P. Santos, Ludovic Righetti, and Auke Ijspeert. 2006. Movement generation using dynamical systems: A humanoid robot performing a drumming task. In *6th IEEE-RAS international conference on humanoid robots*.
34. Brooks, Rodney A., Cynthia Breazeal, Matthew Marjanović, Brian Scassellati, and Matthew M. Williamson. 1999. The cog project: Building a humanoid robot. In *Computation for metaphors, analogy, and agents*, 52–87. Springer.
35. Hajian, Aram Zaven. 1997. A characterization of the mechanical impedance of human hands, PhD Thesis, Harvard University.
36. Gopinath, Deepak. 2015. Enhancing stroke generation and expressivity in robotic drummers-a generative physics model approach. Master's Thesis, Georgia Institute of Technology.

Chapter 3
"Listen Like A Human"—Human-Informed Music Perception Models

3.1 Abstract

One of the main principle guidelines our Robotic Musicianship research is to develop robots that can "**listen like a human and play like a machine**." We would like for our robots to be able to understand music as humans so they can connect with their co-players ("listen like a human") but also surprise and inspire humans with novel music ideas and capabilities ("play like a machine"). The first aspect of this guideline calls for the utilization of computational modeling of music perception that would allow robots to perceive and process music similarly to how humans do. Our assumption is that if a robotic musician could recognize musical concepts meaningful to humans such as beat, stability, similarity, tension and release etc., it would be able to respond musically in a relatable manner that would be understood and appreciated by humans. We believe that such perceptual modeling is crucial for creating a meaningful connection between humans and robotic musicians. The second aspect of the guideline (described in detail in Chap. 4) focuses on the design and development of robots that generate humanly-impossible musical ideas and possess humanly-impossible mechanical abilities. Such ideas and abilities could lead to surprising and inspiring musical outcomes that would push the musical experience to uncharted territories.

In this chapter we describe in detail some of the approaches we explored to allow robots to address the first part of our guideline—"listen like a human." The chapter addresses the design and implementation of both symbolic and audio-based human-perception driven analysis of musical aspects such as rhythm, melody, harmony and timbre, aimed at creating meaningful, expressive, and emotional human-robot interactions.

G. Weinberg et al., *Robotic Musicianship*, Automation, Collaboration, & E-Services 8, https://doi.org/10.1007/978-3-030-38930-7_3

3.2 Rhythmic Analysis of Live Drumming

Our first effort at modeling musical human perception focused on rhythm and was designed for and implemented in our first robotic percussionist—Haile. As part of this project, Haile was designed to analyze rhythmic input from human percussionists in real-time and modify its perceptually-driven responses using stochastic manipulations. The analyzed features included both fundamental musical concepts such note onset timings, pitches, and amplitude, as well as higher level musical percepts such as beat, tempo, rhythmic density, stability, and similarity.

3.2.1 Onset Detection

Haile analyzed audio input from accompanying drummers, recorded from microphones inserted inside hand drums or located above the North American powwow drum. To detect drum strike onsets and amplitude from human drummers we used the Max/MSP bonk~ object [1]. Although the bonk~ object provided effective onset attack detection, its frequency band output was insufficient for accurate pitch detection of the powwow's low and long-reverberating sound. Because bonk~ is hard-coded with a 256-point analysis window, the lowest frequency it can analyze is 172 Hz–too high for the powwow drum, which has a natural fundamental frequency of about 60 Hz. Moreover, pitch becomes even more difficult to detect when high-frequency hits are masked by the long decay of previous low-pitched strikes. To address these issues, we developed a Max/MSP external object that uses 2,048-point analysis windows to determine the magnitude and derivative of lower-frequency bins. By taking into account the spectral changes in addition to magnitudes, we could better determine whether energy in a particular frequency band came from the current hit or from previous ones (see Fig. 3.1).

3.2.2 Beat Detection

Other relatively low-level perceptual modeling modules provide beat and tempo detection using Tristan Jehan's beat~ Max/MSP (based on Scheirer [2]). The object is based on a bank of comb filters, which resonate at a given tempo. The highest peak represents the current tempo, although some bias is needed to prevent sharp changes in tempo. We adjusted the timing parameters in order to provide suitable beat detection in dynamic collaborative playing sessions, allowing humans to play expressively with the robot. One of our initial findings was that lag in the beat detection could cause an unsteady "chase" between the human and the machine, each continually adjusting their playing to match the other. To avoid this effect we programmed Haile to stop adjusting to the tempo in some sections, and added a few restrictive adjustment rules after first acquiring a tempo.

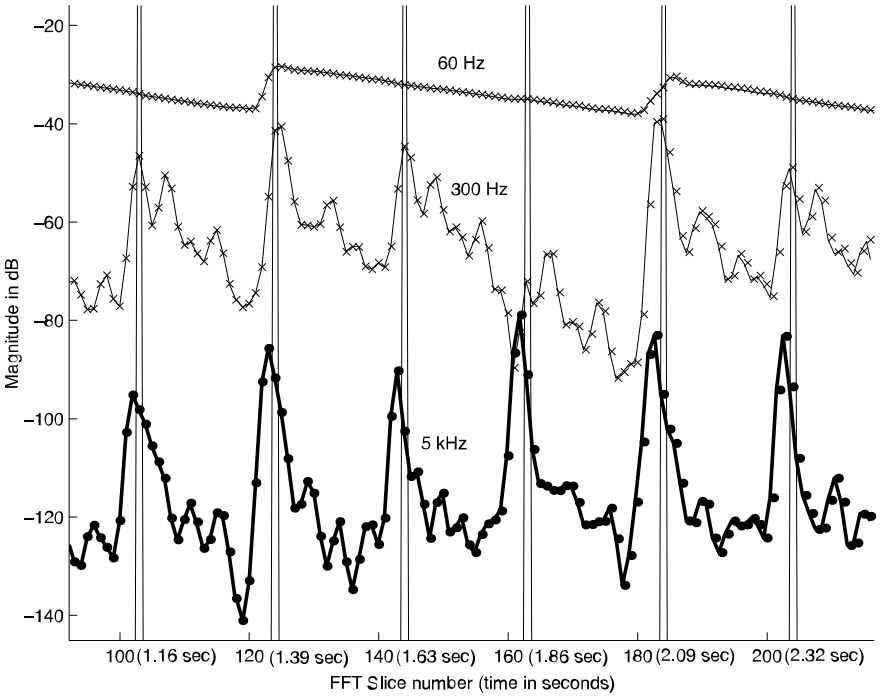


Fig. 3.1 Magnitude plots from 60 Hz, 300 Hz, and 5kHz frequency bands over several low- and high-pitched hits showing the relatively slow decay of the low-pitched hits (A 2,048 point FFT and a 512 point hop size were used.)

3.2.3 Rhythmic Stability and Similarity

Building on the low-level analysis described above, we developed a higher-level rhythmic analysis application in a Max/MSP External format to obtain higher level rhythmic percepts, namely similarity and stability.

Our stability model was based on Desain and Honing's computational model [3], which examines the relationship between pairs of adjacent note durations (or note groupings, see Fig. 3.2), that are rated according to their perceptual expectancy. Stability can be understood as the "predictability of" or ease of tapping one's foot along with a particular rhythm. For instance, the most stable rhythms are metronomic, while the least stable are chaotic. According to Desain and Honing, ryhtmic stability depends on three main criteria: Perfect-integer relationships are favored, ratios have inherent expectancies (i.e., 1:2 is favored to 1:3, and 3:1 is favored to 1:3), and durations of 600 ms are preferred. Desain and Honing define this expectancy as polynomials and as the sum of Gaussians. The expectancy function may be computed as:

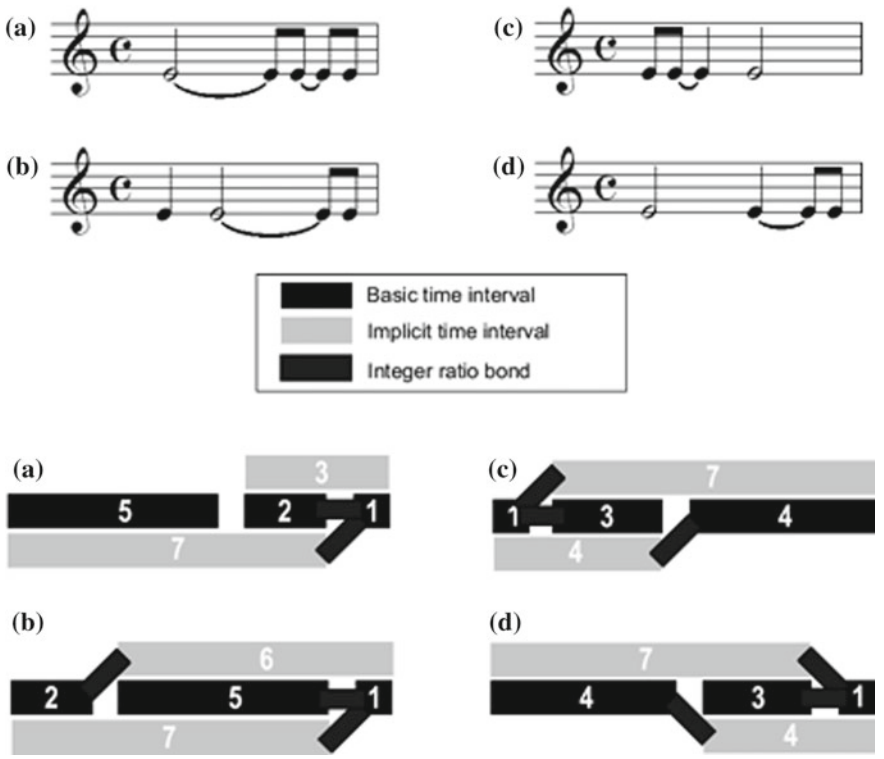


Fig. 3.2 Music notation (top) and corresponding interval representation with bonds connecting intervals with large expectancy (bottom), reproduced from Desain

$$E_b(A, B) = \int_0^r (round(r) - r)x \times |2(r - floor(r) - 0.5)|^p \times round(r)^d \mathrm{d}r \tag{3.1}$$

where r = max(*A*/*B*, *B*/*A*) represents the (near) integer relationship between note duration values. Generally, the expectancy function favors small near-integer ratios and becomes asymmetric when the total duration varies, exhibiting bias towards the 0.6 s tempo interval (see Fig. 3.4) (Fig. 3.3).

To compute the stability rating for an entire rhythmic pattern, we considered each onset in turn. The stability of an onset was computed as the sum of the stability values of all duration pairs it divided. The stability score for the pattern is the geometric mean of the onset stability values.

Our similarity rating was derived from the binary representation described by Tanguiane [4]. We quantized all measure-length rhythms to 48 steps and shifted them so that the first onset is on the first beat of the measure. We then counted the number of overlapping onsets, allowing for small deviations at a cost. For instance, if two onsets are one step away they count half as much. We can retrieve new rhythms from a database based on its similarity to an input rhythm and a desired stability level. Based

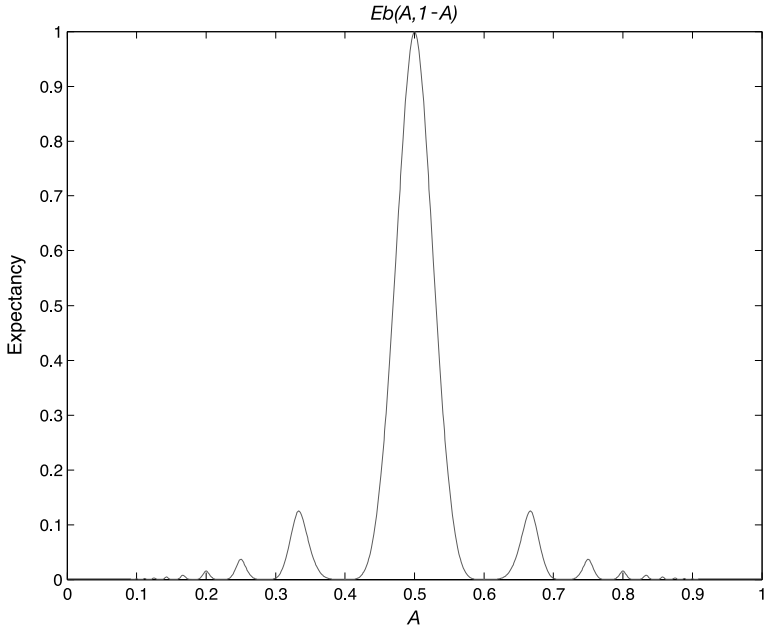


Fig. 3.3 Cognitive expectancy of neighboring time intervals: the expectancy of intervals A and 1-A is higher at integer ratios, 1:1 being the highest

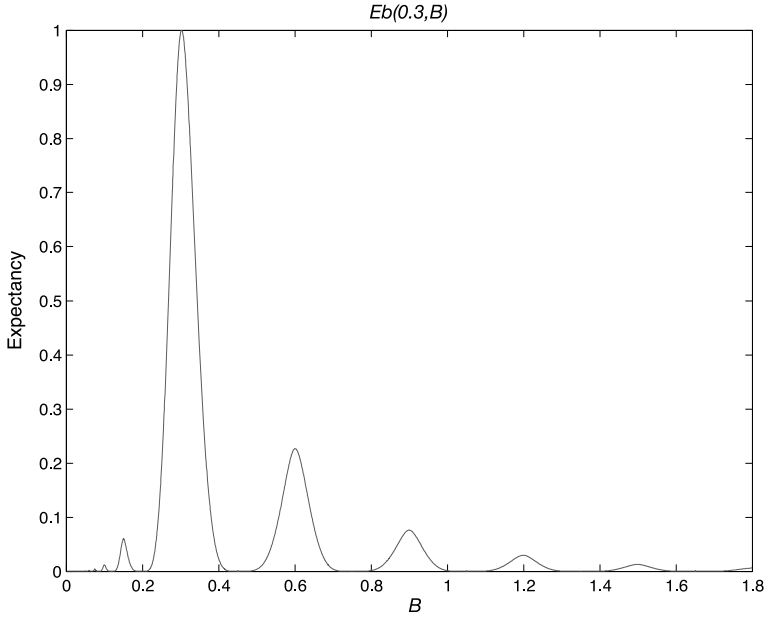


Fig. 3.4 Cognitive expectancy of neighboring time intervals: bias toward expecting intervals of 600 ms (after [3])

on the 48-step measure, we construct a database of over 100,000 rhythms. We do this by considering half, quarter, quarter-triplet, eighth, eighth-triplet, and sixteenth notes constructed by recursively dividing each half-measure into all combinations of twos and threes. Then, every combination of rests is considered (except on the first onset).

Similarity comparisons typically focus on how well two rhythms overlap. Our similarity rating was derived from Tanguiane's binary representation, where two rhythms are first quantized and then given a score based on the number of note-onset overlaps and near-overlaps.

In order to generate new rhythms, we combined the stability rating with a similarity metric. In an effort to support real-time and performance driven applications we implemented the stability and similarity algorithms as externals to the Max/MSP programming environment. The stability object accepted a list of onset times in seconds and outputed a floating-point stability score. The similarity object contained the database of rhythms (one 64-bit integer per rhythm) and the similarity algorithm. In our most recent implementation, it accepted a list of onset times as an input rhythm, the stability of the input, a stability coefficient, and a similarity coefficient. The stability coefficient indicated the desired output stability as a floating point number between zero and one; one indicated the most stable rhythm in the database, zero the least stable, and 1/2 equated to matching input stability. We linearly interpolated between the minimum, maximum, and input stability values to find intermediate values. The similarity coefficient determined the relative contribution of similarity and stability during retrieval. A similarity coefficient of one required that the output would be identical to the input. A similarity coefficient of zero required that the output has the desired stability. Intermediate values offered a compromise between the two.

3.2.4 User Study

Our implementation of similarity and stability was evaluated in a user study involving 14 undergraduate students. The subjects were enrolled in a percussion ensemble class and had at least eight years of experience playing percussion instruments. Each subject spent about 20 min experimenting with different interaction modes with Haile.

As part of the perceptual experiment on stability, subjects were asked to improvise a one-measure rhythmic phrase while Haile provided a 4/4-time beat at 90 beats per minute. Subjects were then randomly presented with three transformations of their phrase: a less stable version, a version with similar stability, and a more stable version. The transformed measures were generated by our Max/MSP stability external using stability ratings of 0.1, 0.5, and 0.9 for less, similar, and more stability, respectively.

The original phrase was played followed by three transformations using Haile's right arm while its left arm provided a metronomic beat. All phrases were played twice to familiarize subjects with the materials. Students were then asked to indicate

which phrase, in their opinion, was less stable, similarly stable, or more stable in comparison to the original input. Stability was explained as representing the predictability of or ease of tapping one's foot along with a particular rhythm. The goal of this experiment was to obtain a preliminary notion about the correlation between our algorithmic implementation and a number of human subjects' perceptions in an interactive setting.

In the perceptual rhythmic stability study, half of the respondents (7 out of 14) correctly identified the three transformations. By comparison, a random response would choose 2.3 out of 14 correctly, on average. The majority of confusions were between similar and more-stable transformations and between similar and less-stable transformations. Only three responses out of the total 42 decisions confused a more-stable version for a less-stable version, implying that larger differences in algorithmic stability ratings made differentiation easier. Only one subject labeled all three generated rhythms incorrectly.

3.3 Tonal Music Analysis Using Symbolic Rules

We developed a method for analyzing high-level musical percepts such as melodic and harmonic tension, which can allow robotic musicians such as Shimon to extract musical meaning from symbolic input played by humans. For example, using this method a robotic musician can detect the level of dissonance in an interval, and follow a rule-based system that would allow it to resolve or extend the dissonance. We based our method on Fred Lerdahl's theory of tonal pitch space [5]. Compared to similar work in this field such as [6, 7], Lerdahl's cognitive theory focuses on the concepts of melodic tension and stability. The theory received criticism claiming that it is not based on empirical findings and cannot be used as an analytical tool [8]. However, more recent research has shown that these formulas can be used effectively in computational systems [9, 10].

Our algorithmic perceptual analysis is based on Lerdahl's analysis of voice leading, which depends on two major components: anchoring strength and relative note distance. The concept of anchoring strength maintains that, given a certain pitch space value, there remain areas of greater and lesser attraction. Our approach utilizes the given pitch space to determine the perceptual anchoring-strength values based on values depicted in Fig. 3.5 (adapted from Lerdahl).

Strength	*Basic Pitch Space (0 = tonic, 11 = leading tone...)*											
4	0											
3	0				4			7				
2	0		2		4	5		7		9		11
1	0	1	2	3	4	5	6	7	8	9	10	11

Fig. 3.5 Anchoring-strength table for computing the attraction between pitches

G♭	G	A♭	A	B♭	B	C	C♯	D	D♯	E	F	F♯
7	6	5	4	3	2	1	2	3	4	5	6	7

Fig. 3.6 Relative note distance

The 0 value represents the root of any given key, 11 represents its leading tone, and values 1 through 10 correspond to the ten notes in between. The values 0, 4, and 7 have the strongest anchoring strength, as these pitch classes correspond to the tones of a major triad.

Another major component of Lerdahl's voice leading equation relies on relative note distance. In Fig. 3.6, the center represents the previous pitch—in this example, C. The relative note distance grows as notes move farther away from C. This distance inversely affects the probability of selection as a following note (C to C has a distance of 1 to avoid division by 0). Accordingly, there is a generative preference towards smaller melodic intervals.

In Lerdahl's stability equation for voice leading (Eq. 3.2), the effect of the next note's stability is inversely proportional to the previous note's anchoring strength:

$$S = \frac{a_2}{a_1}\frac{1}{n^2} \tag{3.2}$$

where a_1 and a_2 represent the previous and next note's anchoring strength, respectively, and n represents the relative step-size from the previous pitch to the next pitch.

3.3.1 Implementation

To evaluate our perceptual model we developed a generative musical system that controls for melodic tension and stability, and conducted a user study to assess the perception of these elements by subjects. We hypothesized that humans' perception of these high-level musical percepts would have high correlation to our parametric manipulation. Three main generative modules were developed for the model: melody, rhythm and harmony, as can be seen in Fig. 3.7.

Our melodic generative system calculated the probability for possible next generated notes provided the previous note, based on Lerdhal's melodic tension model. It also derived the probability for any given note to sound an octave above or below the previous note. Given a certain harmony, we aimed to create a unique anchoring-strength set within two octaves, extending Lerdahl's single octave anchoring strength set to 24 by adding columns left of 0, therefore providing an anchoring set one octave below any tone. This adjustment enhanced the opportunity for more precise manipulation of the equations. Our generative algorithm measured the distance between the most recent pitch value and all prospective pitch values based on Lerdhal's relative

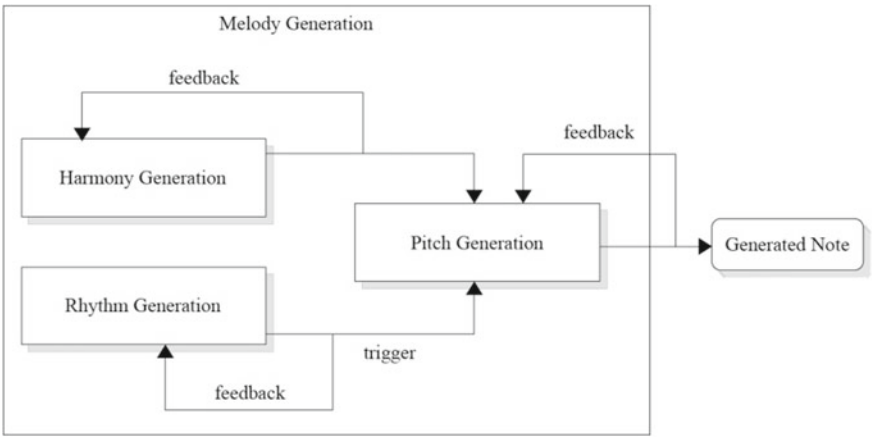


Fig. 3.7 The interaction between rhythm-, pitch-, and harmony-generation modules used to generate the next note

note distance table. This distance inversely affected the probability of selection as a following note (C to C has a distance of 1 to avoid division by 0). Accordingly, there was a generative preference towards smaller melodic intervals.

$$L(P) = \frac{a_2}{a_1}\frac{1}{n^Y} + X \tag{3.3}$$

Equation 3.3 is an altered form of Eq. 3.2, specialized for generative purposes: where $L(p)$ represents the likelihood that a given pitch will occur next, and where the variables' values lie in these ranges: a: 15–1; z: 2–0; n: 0–12; 10–100; and input tension parameter (not shown in equation): 0–1. Responding to critics of Lerdahl's work, such as [8], and in an effort to reach our own subjectively satisfactory musical results, we decided to experiment with and manipulate some of the parameters in the formula. We added variables x, y and z and mapped them to a single input, T (for tension), controlling whether stable or unstable pitches are more likely to be generated. The larger this input parameter, the more likely it is for an unstable pitch to be played. Changing z controls the influence of anchoring strength in determining the next pitch. As tension T increases, z decreases, reducing the likelihood that strong anchoring pitches will be generated. Similarly, y affects the impact of the relative step size. As discussed earlier, theorists have shown that smaller steps between pitches increase the perception of stability. As the tension input value approaches zero, a small pitch step size becomes more likely, and therefore the output becomes more stable. Variable x effectively adds noise to the equation. By raising x, anchoring strength and step size become relatively less significant in generating the next note. This makes unstable pitches more likely. We consider this extension of Lerdahl's formula to be a primary contribution of the present research.

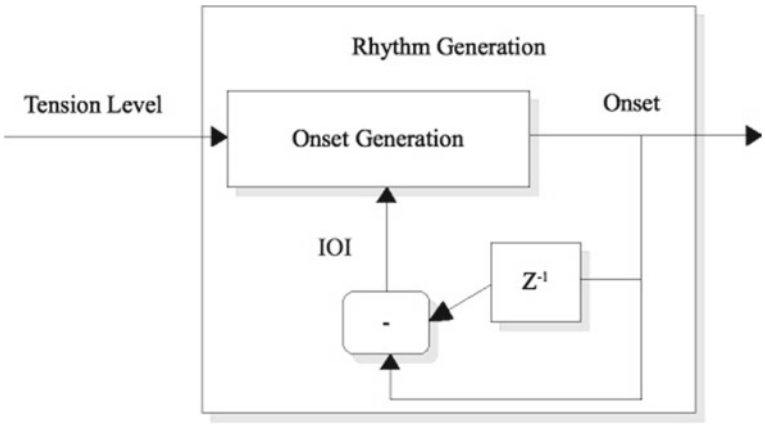


Fig. 3.8 Rhythm generation based on tension level and previous inter-onset interval

In order to generate melodic lines that could be evaluated by human listeners, we integrated a rhythm generator into the melodic pitch generator described above. We based the rhythm generator on a model proposed by Desain et al. [3] for analysis of rhythmic stability. Their work demonstrated the relationship between rhythmic stability and the bounds between contiguous Inter-onset Intervals (IOIs). In particular, they showed direct proportionality between the complexity of ratios between contiguous durations and relative rhythmic stability. Extending the concept for analyzing stability into a predictive model, we implemented a method for rhythmic generation. In our predictive implementation, the algorithm refers to previous IOIs to inform the generation of future onsets, as shown in Fig. 3.8.

Provided a high or low input tension level, the algorithm accordingly gives preference to future onsets that form either complex or simple ratios, respectively, with the previous IOI. The onset prediction relies on a lookup table in order to pseudo-randomly generate future onsets. Its lookup table includes a list of ratios arranged according to complexity, where ratios such as 1/2 and 2/1 occur low on the list, whereas 9/2 and 2/9 occur significantly higher. Influencing the pseudo- random generation, high input tension values give weight to ratios high on the list and, vice versa, low tension values give weight to lower ratios. The algorithm combines the note density mapping with the rhythmic stability prediction. To do so, it first considers the influence of the speed mapping. This determines the relative note density. The onset generation then pseudo-randomly generates the next onset with a more or less complex ratio between IOIs, but also weights the lookup table probabilities based on distance from the relative note density.

To provide harmonic context to our melodic and rhythm generator, we developed a harmonic generation module, which addresses pitch relationships between groups of notes that are simultaneous or close together in time, and governs the choice of pitches in simultaneous, independent melodies (polyphony). The harmonies generated by our algorithm influence the movement of each melody. As listeners, we have

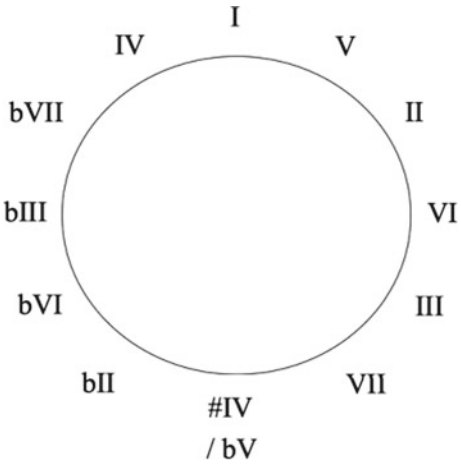


Fig. 3.9 Circle of fifths

expectations about the movement from one harmony to the next. Through subjective and physiological response studies, researchers have found a correlation between harmonic expectations and chords related by the circle of fifths (see Fig. 3.9), a theoretical model that orders pitches according to a regular interval shift of seven semitones, or five diatonic scale steps [11, 12].

Similar to the rhythm-generation module, harmony generation depends on a lookup table to generate the next harmony. We wanted to limit the scope of the harmonic possibilities in order to rely on a simple model of harmonic expectation. In doing so, we limited the lookup table to diatonic triads of a major scale. We ordered the table according to expectation. Based on the last harmony generated, we calculate expectation from movement on the circle of fifths. Low values on the table values that are more expected correspond to small movements on the circle of fifths. Higher values, relating to more unexpected and therefore tense harmonic shifts, correspond to large movements on the circle. A harmonic tension value influences the generation of the next harmony. As with rhythm generation, higher tension values weight the probability of generating a more unexpected harmony. Conversely, low tension values increase the chance of the algorithm generating a low table value, an expected harmony.

3.3.2 Evaluation

In an effort to evaluate the effectiveness of the algorithm in representing various degrees of tension in real time, we conducted a user study designed to assess the relationship between algorithmically generated tension and perceived tension. The user group included 100 volunteer students pooled from Georgia Tech. We presented to each subject 100 four-second excerpts of audio. To account for the relative effects imposed by the order of the excerpts, each trial employed a randomized sequence.

To evaluate the influence of these parameters on perceived melodic tension, we manipulated the register, density, and instrumentation of the musical excerpts generated by the algorithm. Knowing how these other features affect the perception of tension will allow us, in future revisions of the algorithm, to normalize across features. Pitch material was classified as either high or low register, as excerpts contained notes that are exclusively either higher or lower than C4. Note density was categorized using average IOI, as either longer or shorter than 750 ms. We subcategorized instrumentation by sustain and brightness levels. Two of the instruments were sine-tone generated, one with long sustain and the other with short sustain. Three other sampled instruments offered differences in sustain and brightness, classified as either bright or dark in timbre. For all combinations of these categories we generated excerpts at five different tension levels, with level 5 representing high tension and level 1 representing low tension.

After listening to each clip, listeners indicated tension using magnitude estimation where any number may be assigned to represent perceived tension. Magnitude estimation provided a solution to two major concerns. First, in an assignment system constrained by maximum and minimum values, the subject limits the range with the first assignment of either boundary. For instance, if the maximum permitted value was 10 and the subject indicated 10 for the previous excerpt yet found the next excerpt even more tense, they would have no additional range for expressing this relativity. In order to resolve this problem, the procedure could have first provided maximum and minimum examples of tension.

This would impose designer-interpreted conditions on the subjects, however. On the other hand, magnitude estimation, and in particular modulus- free magnitude estimation, is used to address these issues. In order to account for earlier inconsistencies due to initial ambiguity in the perceived range and resolution, the first five values of each trial were discarded.

Working with data from magnitude estimation that has no consistency in range and boundary across subjects, we used geometric means, rather than normal arithmetic means, to represent all of the available data within an equivalent context across categories and between subjects. Although the IOI compared to perceived tension showed only slight correlation, registration and instrumentation proved significantly influential towards affecting perceived tension. Post hoc Tukey-Kramer correction (α.05) was used to evaluate and verify significance across all of the results. As shown in Fig. 3.10, music generated with the same parameters but in a higher register proved, on average, 24% more tense than when compared to music in a lower register.

Comparing sine-tone instruments, we found, as expected, that sustaining notes are perceived as sounding more tense than shorter, resonating notes. We hypothesize that as the sustained notes overlap succeeding notes, they may cause beating, and therefore a more distinct sensory dissonance. Additionally, we found that brighter instruments, as shown in Fig. 3.6 (right), appeared more tense than darker instruments. This finding is supported by existing research in sensory dissonance, with brighter sounds having stronger high-frequency harmonics beating against each other [13–16]. We aim to consistently model tension across instrumentation. In order to normalize across these

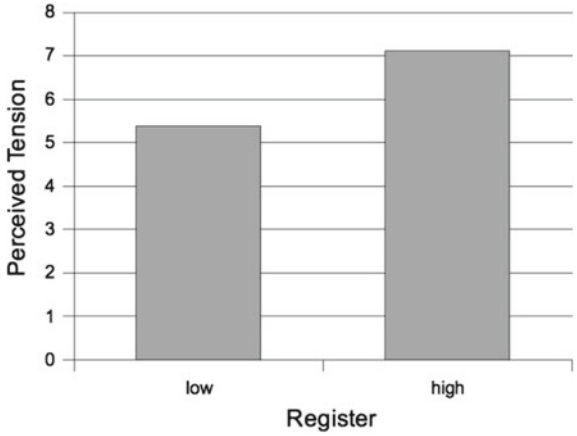


Fig. 3.10 Geometric mean response in perceived tension as compared to change in register

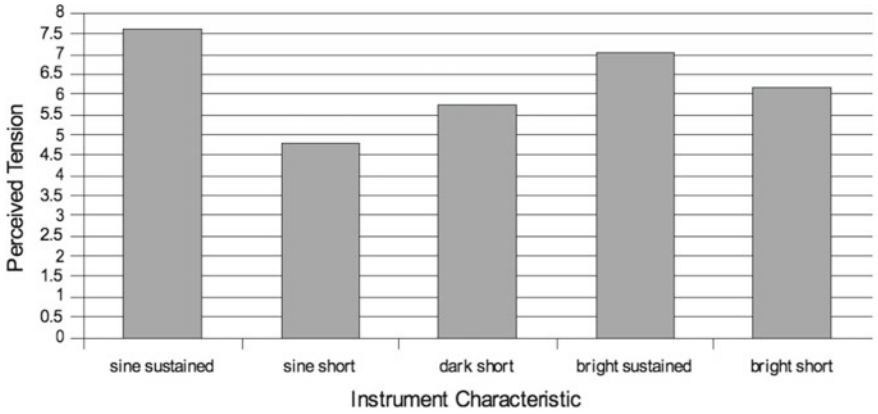


Fig. 3.11 Geometric mean response in perceived tension as compared to changes in instrumentation

different instrumentations, we must model the impact of each instrument on the perceived tension, as shown in Fig. 3.11.

In evaluation of the tension control of the algorithm, we compared perceived tension to the tension input level across all manipulated conditions. Figure 3.12 shows the results of this analysis, with a direct, linearly proportional correlation (r 0.98) between the input tension level and subjectively perceived tension. This correlation demonstrates a 1:1 relationship between the tension control of our generative system and the perceived tension. It also supports the melodic tension percepts laid out by Lerdahl and Krumhansl [10], and the effectiveness of our modifications of Lerdahl's formulas.

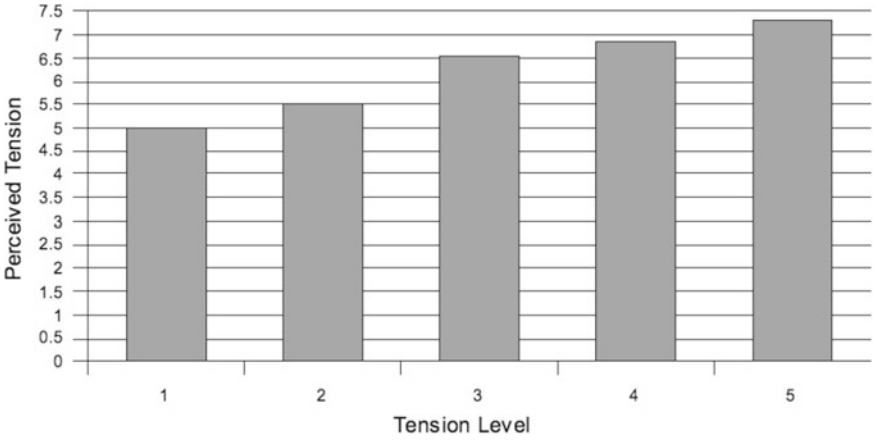


Fig. 3.12 Geometric mean response in perceived tension, as compared to change in the input tension parameter

3.4 Music Analysis Using Deep Neural Networks

This section describes our effort to move from rule based systems to data based systems where we aim to derive meaningful musical features by employing self-supervising training techniques for neural networks. The objective of data-driven modeling is to capture meaningful musical concepts that cannot easily be codified by a rule-based expert system.

3.4.1 Deep Musical Autoencoder

Self-supervised learning has proven to be useful in many domains ranging from music to language to computer vision. A common method of self-supervised training uses denoising autoencoders which are networks that learn to reconstruct noisy input representations of the original data. If trained effectively the bottleneck layer of an autoencoder can encode useful high level features about the domain.

In this work a deep autoencoder was trained to encode and decode a single measure of music. A dataset consisting of 4,235 lead sheets from the Wikifonia database containing melodies from genres including (but not limited to) jazz, folk, pop, and classical was used [17]. In addition, 120 publicly available jazz solo transcriptions from various websites were collected. The dataset contained roughly 170,000 unique measures. Of these, roughly 20,000 unique rhythms were seen in the measures. The dataset was augmented by manipulating pitches through linear shifts (transpositions), tempo doubling, and alterations of the intervals between notes resulting in roughly 80 million unique measures. The interval alterations were performed in order to expand

the dataset beyond what might be played by humans. For example, Shimon is better at playing fast sequences of notes that have large pitch intervals (greater than eight half steps) compared to small intervals. If these types of units are not present in the database then Shimon would never play them. By adjusting the intervals there is risk of sacrificing the musical validity because they are no longer human composed or confirmed. We attempt to maintain some validity by not modifying any of the rhythmic content. Furthermore, the number of units in the resulting library created by this process compose roughly 5% of the unit library. Therefore, the autoencoder remains significantly biased towards effectively learning the semantics of the human valid units.

The intervals are altered using two methods: (1) adding a constant value to the original intervals and (2) multiplying a constant value to the intervals. Many different constant values are used and the resulting pitches from the new interval values are superimposed on to the measure's original rhythms. The new unit is added to the dataset. We restrict the library to measures with pitches that fall into a five octave range (midi notes 36–92). Each measure is transposed up and down continuously by half steps so that all instances within the pitch range are covered. The only manipulation performed on the duration values of notes within a measure is the temporal compression of two consecutive measures into a single measure. This "double time" representation effectively increases the number of measures, while leaving the inherent rhythmic structure in tact. After this manipulation and augmentation there are roughly 80 million unique measures. We use 60% for training and 40% for testing the autoencoder.

The first step in the process is feature extraction and creating a vector representation of the unit. Unit selection allows for a lossy representation of the events within a measure. As long as it is possible to rank the units it is not necessary to be able to recreate the exact sequence of notes with the autoencoder. Therefore, we can represent each measure using a bag-of-words (BOW) like feature vector. The features include:

1. counts of note tuples $<\text{pitch}_1, \text{duration}_1>$
2. counts of pitches $<\text{pitch}_1>$
3. counts of durations $<\text{duration}_1>$
4. counts of pitch class $<\text{class}_1>$
5. counts of class and rhythm tuples $<\text{class}_1, \text{duration}_1>$
6. counts of pitch bigrams $<\text{pitch}_1, \text{pitch}_2>$
7. counts of duration bigrams $<\text{duration}_1, \text{duration}_2>$
8. counts of pitch class bigrams $<\text{class}_1, \text{class}_1>$
9. first note is tied previous measure (1 or 0)
10. last note is tied to next measure (1 or 0).

These features were chosen in an attempt to keep the feature vector size to less than 10,000 so that training and computation could be performed in a reasonable time frame using a laptop computer. The pitches are represented using midi pitch values. The pitch class of a note is the note's pitch reduced down to a single octave (12 possible values). Rests of each duration are also included and treated similarly

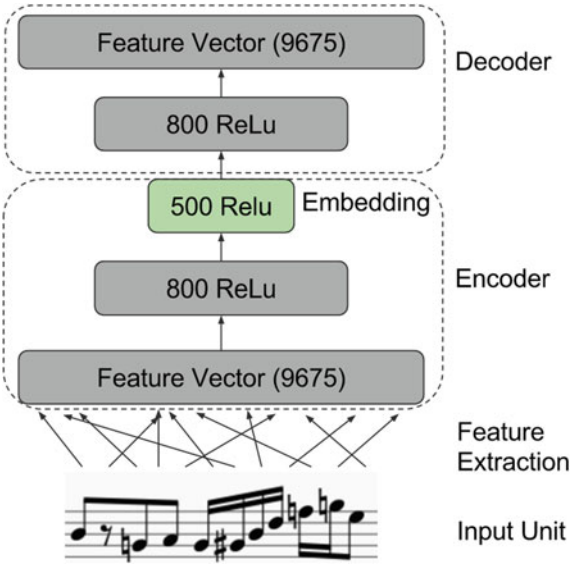


Fig. 3.13 Autoencoder architecture—the unit is vectorized using a BOW like feature extraction and the autoencoder learns to reconstruct this feature vector

to pitched notes (like a 13th pitch class). We also represent rests using a pitch value equal to negative one. Therefore, no feature vector will consist of only zeros. Instead, if the measure is empty the feature vector will have a value of one at the position representing a whole rest. Because we used data that came from symbolic notation (not performance) the durations can be represented using their rational form (numerator, denominator) where a quarter note would be '1/4.' Finally, we also include beginning and end symbols to indicate whether the note is a first or last note in a measure. The resulting feature vector includes the counts and is not normalized, however, the unit size of one measure is constant.

The architecture of the autoencoder is depicted in Fig. 3.13. The objective of the decoder is to reconstruct the feature vector and not the actual sequence of notes as depicted in the initial unit of music. Therefore, the entire process involves two types of reconstruction:

1. **feature vector reconstruction**—the reconstruction performed and learned by the *decoder*.
2. **music reconstruction**—the process of selecting a unit that best represents the initial input musical unit.

In order for the network to learn the parameters necessary for effective feature vector reconstruction by the decoder, the network uses leaky rectified linear units ($\alpha = 0.001$) on each layer and during training minimizes a loss function based on the cosine similarity function

$$sim(\mathbf{X}, \mathbf{Y}) = \frac{\mathbf{X}^T \cdot \mathbf{Y}}{|\mathbf{X}||\mathbf{Y}|} \tag{3.4}$$

where **X** and **Y** are two equal length vectors. This function serves as the basis for computing the distance between the input vector to the encoder and output vector of the decoder. Negative examples are included through a softmax function

$$P(\mathbf{R}|\mathbf{Q}) = \frac{\exp(sim(\mathbf{Q}, \mathbf{R}))}{\sum_{\mathbf{d}\epsilon D} \exp(sim(\mathbf{Q}, \mathbf{d}))} \tag{3.5}$$

where **Q** is the feature vector derived from the input musical unit, Q, and **R** represents the reconstructed feature vector of Q. D is the set of five reconstructed feature vectors that includes **R** and four candidate reconstructed feature vectors derived from four randomly selected units in the training set. The network then minimizes the following differentiable loss function using gradient descent

$$-log \prod_{(Q,R)} P(\mathbf{R}|\mathbf{Q}) \tag{3.6}$$

A learning rate of 0.005 was used and a dropout of 0.5 was applied to each hidden layer, but not applied to the feature vector. The network was developed using Google's *Tensorflow* framework [18].

3.4.2 Music Reconstruction Through Selection

The feature vector used as the input to the autoencoder is a BOW-like representation of the musical unit. This is not a loss-less representation and there is no effective means of converting this representation back into its original symbolic musical form. However, the nature of a unit selection method is such that it is not necessary to reconstruct the original sequence of notes. Instead, a candidate is selected from the library that best depicts the content of the original unit based on some distance metric.

In concatenative text-to-speech (TTS) methods, this distance metric is referred to as the *target cost* and describes the distance between a unit in the database and the target it's supposed to represent [19]. In this musical scenario, the targets are individual measures of music and the distance (or cost) is measured within the embedding space learned by the autoencoder. The unit whose embedding vector shares the highest cosine similarity with the query embedding is chosen as the top candidate to represent a query or target unit. We apply the function

$$\hat{y} = \arg\max_{y} sim(x, y) \tag{3.7}$$

where x is the embedding of the input unit and y is the embedding of a unit chosen from the library.

The encoding and selection can be objectively and qualitatively evaluated. For the purposes of this particular musical autoencoder, an effective embedding is one that captures perceptually significant semantic properties and is capable of distinguishing

Table 3.1 Autoencoder ranking and collision results

mean rank@50	1.003
accuracy@50	99.98
Collision rate per 100 k	91

the original unit in the library (low collision rate) despite the reduced dimensionality. In order to assess the second part we can complete a ranking (or sorting) task in which the selection rank (using Eq. 3.5) of the truth out of 49 randomly selected units (rank@50) is calculated for each unit in the test set. The collision rate can also be computed by counting the instances in which a particular embedding represents more than one unit. The results are reported Table 3.1.

Given the good performance we can make a strong assumption that if an identical unit to the one being encoded exists in the library then the reconstruction process will correctly select it as having the highest similarity. In practice, however, it is probable that such a unit will not exist in the library. The number of ways in which a measure can be filled with notes is insurmountably huge and the millions of measures in the current unit library represent only a tiny fraction of all possibilities. Therefore, in the instances in which an identical unit is unavailable an alternative, though perceptually similar, selection must be chosen.

Autoencoders and embeddings developed for image processing tasks are often qualitatively evaluated by examining the similarity between original and reconstructed images [20]. Likewise, we can assess the selection process by reconstructing never before seen music.

Figure 3.14 shows the reconstruction of an improvisation (see the related video for audio examples[1]). Through these types of reconstructions we are able to see and hear that the unit selection performs well. Also, note that this method of reconstruction utilizes only a target cost and does not include a concatenation cost between measures.

Another method of qualitative evaluation is to reconstruct from embeddings derived from linear interpolations between two input seeds. The premise is that the reconstruction from the vector representing the weighted sum of the two seed embeddings should result in samples that contain characteristics of both seed units. Figure 3.15 shows results of reconstruction from three different pairs of units.

3.5 Real-Time Audio Analysis of Prerecorded Music

3.5.1 Introduction

In the previous sections we presented a few approaches for symbolic analysis of rhythm, melody, and harmony. Shimon utilized these approaches to process real-time MIDI input and symbolic representation of notes detected from acoustic instruments,

[1] https://youtu.be/BbyvbO2F7ug.

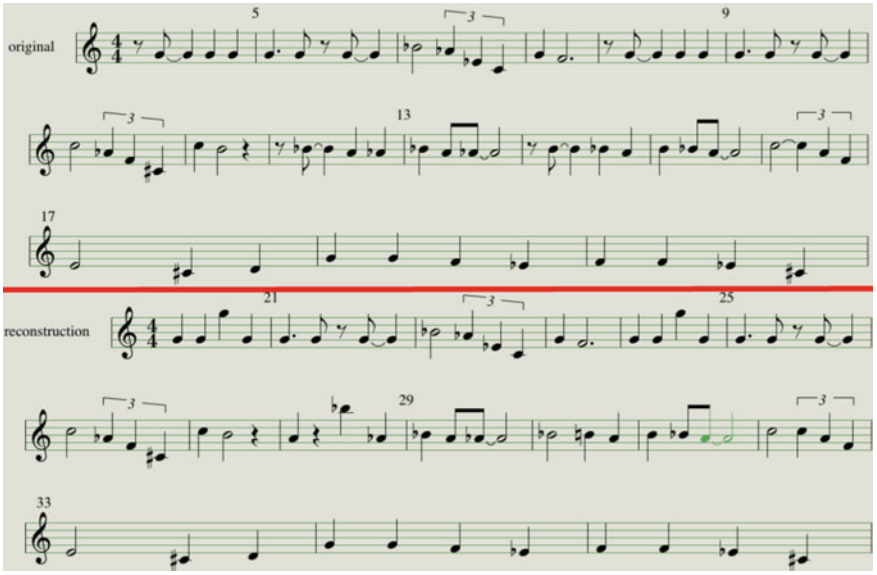


Fig. 3.14 The music on the stave labeled "reconstruction" (below the line) is the reconstruction (using the encoding and unit selection process) of the music on the stave labeled "original" (above the line)

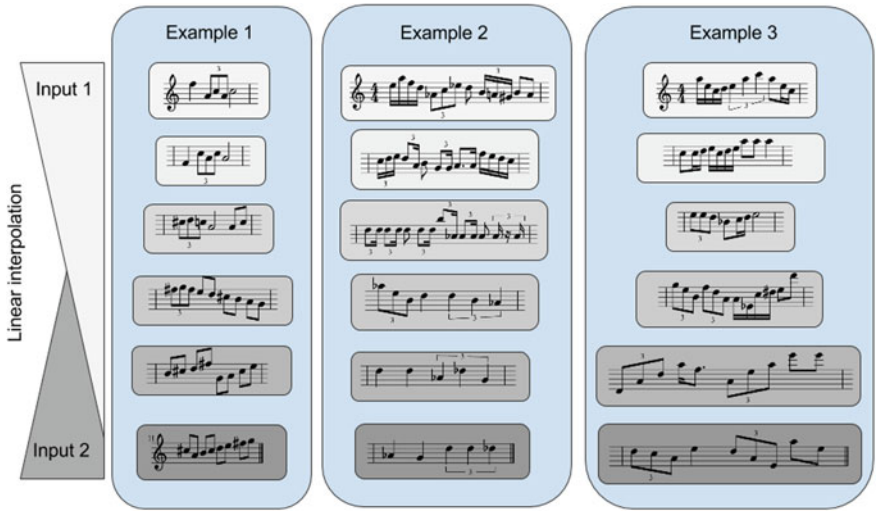


Fig. 3.15 Linear interpolation in the embedding space in which the top and bottom units are used as endpoints in the interpolation. Units are selected based on their cosine similarity to the interpolated embedding vector

as well as off line analysis of symbolic transcriptions of tonal musical data sets. In this section we describe an audio based approach for music analysis using real-time Musical Information Retrieval (MIR) techniques. The goal of this project, titled The Shimi Band, was for a group of three Shimi Robots to analyze pre-recorded songs and respond to the analysis with audio-driven synchronized dancing movements. To address this challenge we implemented a set of MIR techniques and designed a set of corresponding robotic gestures to represent the robots' musical understanding. The improvisatory gesture choreography was then evaluated by a user study.

3.5.2 Previous Work

In related projects developed for Shimi, humans could interact with the robot by tapping, head bobbing, playing a musical instrument, or speech. In these projects, based on human input, Shimi played relevant music through its built-in speakers and danced to the music in a manner that represented detected features in the music. In our Query by Tapping project, for example, Shimi estimated beat and tempo from human tapping, and responded by retrieving and playing a song in the same tempo. Similarly, in the Query by Movement Project, Shimi estimated the tempo and beat from a human head bobbing and retrieved songs from its database with similar tempo and beat. We also used speech detection using Google Voice to identify a few key words, for which Shimi responded by selecting songs by different artists, genres and moods. Rather than responding to human input, the goal of the Shimi Band Project was to allow for a of group of Shimis to listen to and analyze pre-recorded music, and to automatically respond with synchronized movements that match the detected musical features.

3.5.3 System Design

To synchronize three Shimi Robots and perform MIR tasks we used a Mac Pro laptop server, which reduced the computational requirement on the robot's limited phone processor.

The overall system structure is shown in Fig. 3.16. The central Mac server receives live audio input, extracts audio features and analyzes higher level musical parameters in real-time. It also runs a parallel process to plan the gestures structure for the Shimi Band movements, sending messages to the robots via UDP. A certain degree of randomization was implemented in the system, leading to diverse and constantly changing gestural response. Real-time analysis and gesture planning, combined with low network latency, led to instantaneous reaction of the robots to the analyzed music.

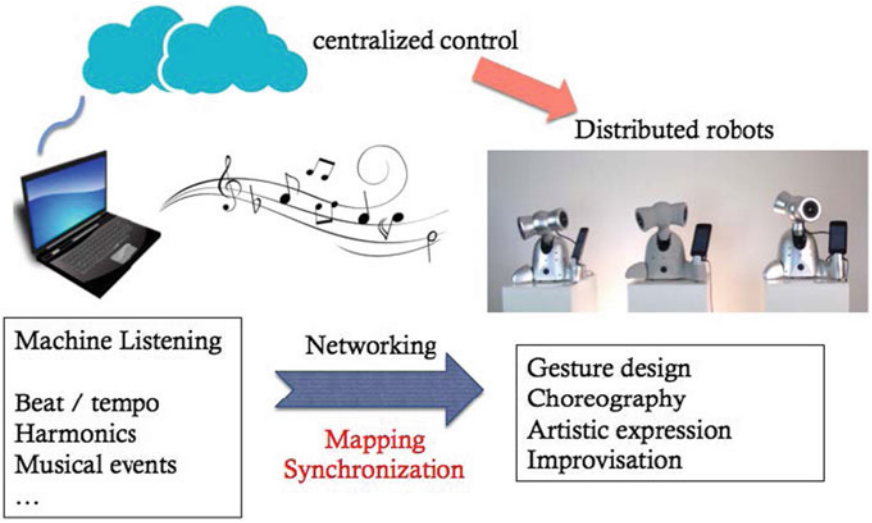


Fig. 3.16 Shimi band system structures

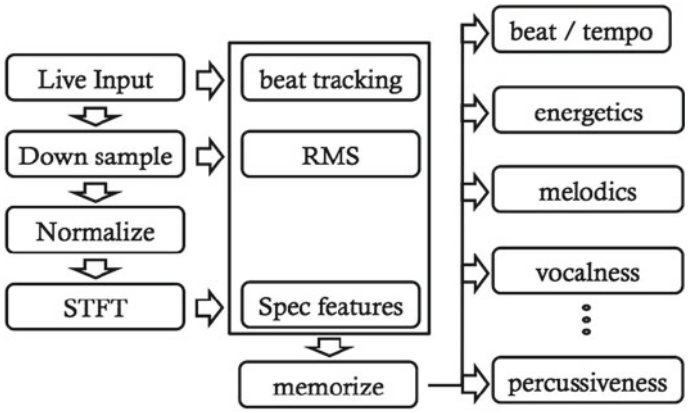


Fig. 3.17 Structure of real time audio analysis

3.5.4 Live Audio Analysis

To drive the robotic musical perception and dance procedures, the central computer received live audio input from the Mac microphone and extracted time domain and spectral domain features in real-time such as Energetics, Melodics, and Soloness [21]. Real-time beat tracking was implemented in PD using the Retibe object [22] as depicted in Fig. 3.18. Each sample block was downsampled to 4000 Hz to calculate Root Mean Square (RMS) value and other time domain features such as skewness, kurtosis. After performing normalization and Short Time Fourier Transform (STFT),

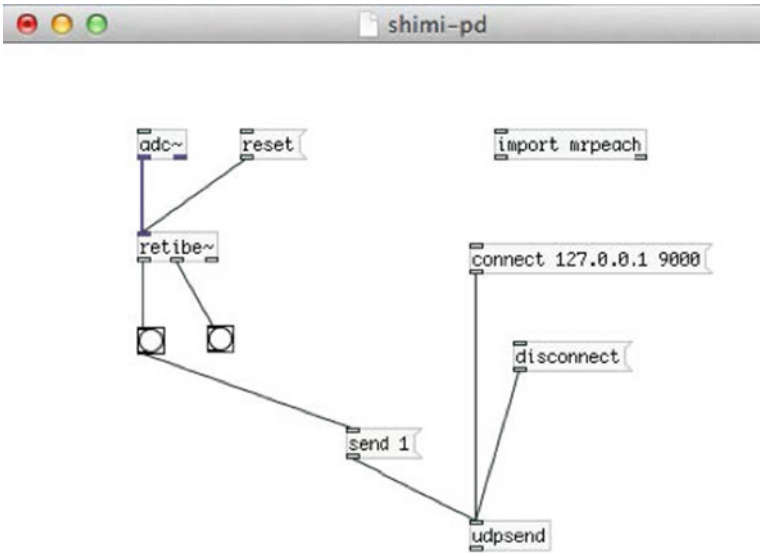


Fig. 3.18 PD Patch for beat tracking

spectral domain features were extracted. These lower-level features (indicated in the box in Fig. 3.17) were fed into a memory buffer to keep track of historical data. There were then used to compute the higher musical level features such as beat, energetics, melodics, vocalness and percussiveness. To receive a smooth feature output, block length was set to 11,290 with hop size of 512, with an FFT size of 1024, resulting in an overlap of about 95%.

For tempo estimation, we filtered out onset intervals smaller than 10 ms since: 1. Such small intervals are not likely to occur in pop songs, and 2. Shimi’s motors are not fast enough for the robot to respond to such fast events. Onset intervals above the threshold were used to calculate the song’s tempo. Every valid onset interval was sent into a sliding beat duration buffer, where its median was compared to the current onset duration. If the duration value were similar, the averaged tempo value sent into another stable buffer. Tempo was considered as stable only if the stable buffer had long enough concatenating blocks. A timer was added to keep track of the tempo estimation time. If a stable tempo was not detected in this time limit, a default tempo was set and sent to the Shimi Band. The process of tempo estimation is illustrated in Fig. 3.19.

To calculate the instantaneous energy level during the song we use the average square of the amplitude, or Root Mean Square (RMS). A sliding buffer was used to keep track of the RMS values across time. The RMS value of each block was fed into the buffer, followed by a smoothing step to avoid sudden change. The energy level was weighted by the estimated tempo. Faster tempo had positive influence on the output while slower tempo had a negative weight. The final output was normalized to a value between 0 (lowest) and 1 (highest). Figure 3.20 depicts the calculation of the energetics feature.

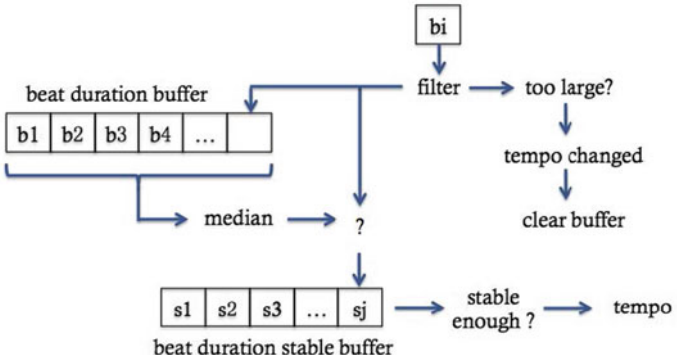


Fig. 3.19 Tempo estimation flowchart

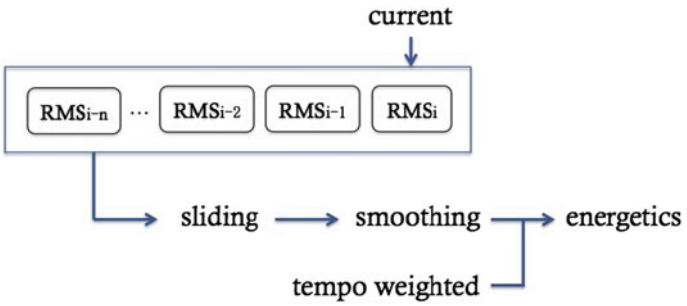


Fig. 3.20 Energy level calculation

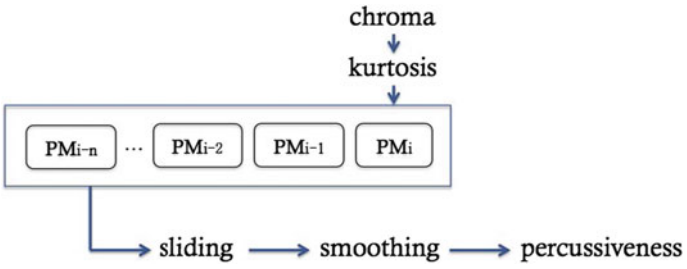


Fig. 3.21 Percurssiveness/melodics calculation

To define the percussiveness versus melodics nature of the music we used the chroma feature (see Fig. 3.21). Percussive sounds usually exhibit flat chroma, while clear melody presents high energy, focused on one of the chroma bins. The kurtosis of the chromagram was therefore calculated as an indicator of the sound percussiveness versus melodics. A window buffer was used to smooth the output, which was normalized to a value between 0 and 1. High percussiveness corresponded to a low melodics value and vice versa.

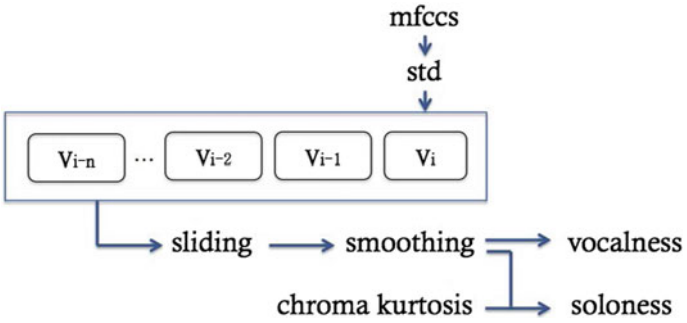


Fig. 3.22 Vocalness/soloness calculation

Mel-Frequency Cepsturm Coefficients (MFCCs) were used to calculate vocalness and soloness of the music due to its close connection to speech. When vocals appear in the music, the MFCCs exhibit distinguishing decreased variance. Therefore, standard deviation of the MFCCs was calculated for each block and a sliding window was used to smooth the output. The difference between vocalness and soloness was determined so that soloness also indicated a solo performance by any instruments rather than just vocals. The chroma kurtosis (melodic level) was combined with MFCC standard deviation to calculate the soloness, which fell between 0 and 1. Figure 3.22 shows the vocalness computation process.

Several other features including spectral centroid and pitch were extracted as well. These could be used for training classifiers and other mapping strategies in the future.

The application interface displayed the detected features in real-time as shown in Fig. 3.23. The color bars on the left indicates chroma features. MFCCs values are on the right. The current beat duration, pitch, and RMS level appears at the bottom. The final output features are displayed on the top right corner.

3.5.5 Gesture Design

We developed a gesture generation framework in an effort to allow each Shimi to move expressively with only 5 DoFs (Fig. 3.24). Our singular gesture model was designed based on Laban Movement Analysis [23]. The system took musical elements as input to create a set of parameterized gestures, using a set of different pre-designed gestures for different styles of music. The amplitude of the gesture movements was designed to reflect the musical dynamics in real-time. Singular gestures were used as basic components for a rule based choreography system that synchronised the movements of all 3 Shimis in The Shimi Band.

A number of performance modes were designed for the Shimi Band. In Solo Mode, only one Shimi moves, while the other two are inanimate. In Pairwise Mode two Shimis are synchronized and the third moves by itself, while in Triple Mode all Shimis

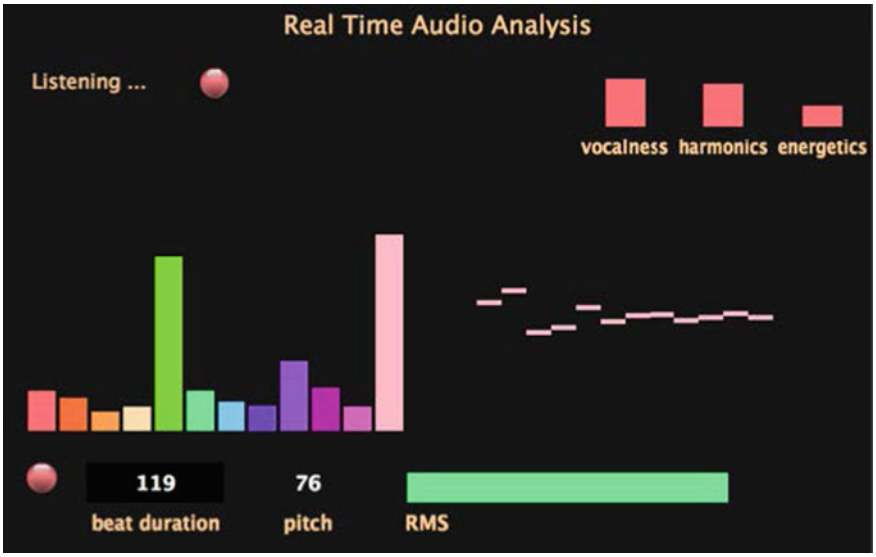


Fig. 3.23 Real time audio analysis interface

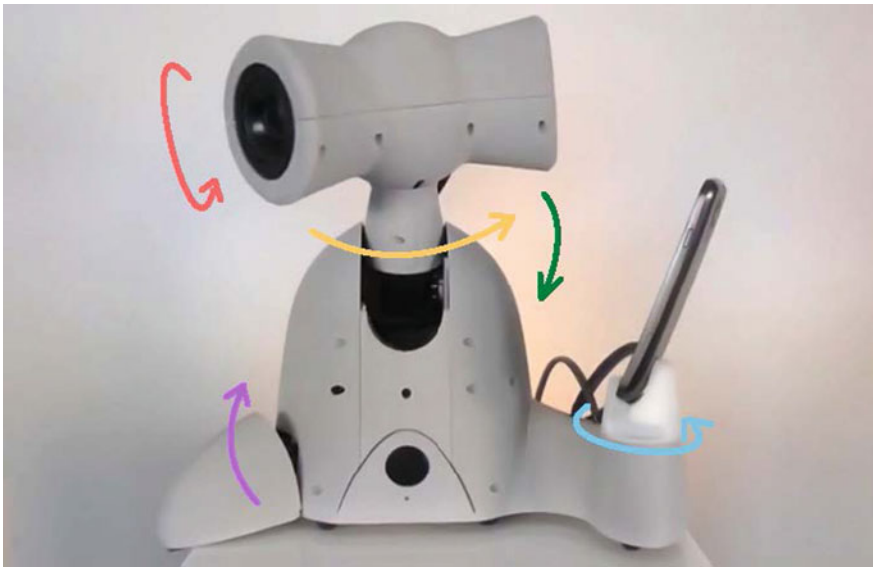


Fig. 3.24 Shimi's five degrees of freedom

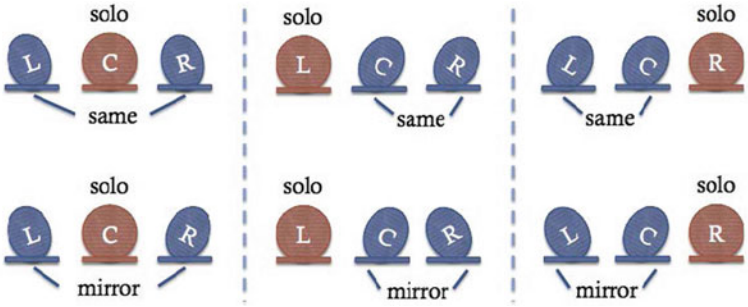


Fig. 3.25 Pairwise interaction

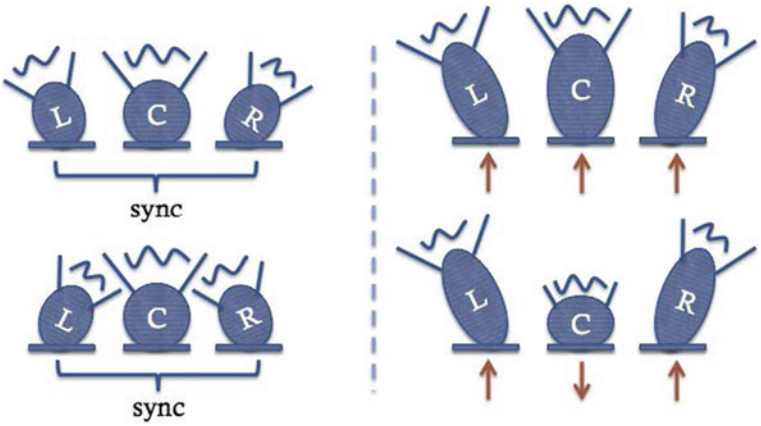


Fig. 3.26 Oriented interaction

move. Figure 3.25 shows different approaches designed for duo interactive modes, where one Shimi dances using idiosyncratic movements, the other two dance together using either the same or mirrored gestures. Figure 3.26 shows oriented interaction mode, where the three robots move synchronously facing different directions.

Sequential movements were designed to increase the perception of interaction among the robots. Figures 3.27 and 3.28 show two sequential choreography approaches: Single Sequential and Accumulative Sequential modes. In Sequential Mode—gestures start from one end of the band followed sequentially by the other two. In Accumulative Sequential modes—the robots join the dance sequentially until they all dance together. The movement direction can be from left to right, right to left, middle to both sides and vice versa. This choreography can be applied to any singular gesture in the library, leading to a wide range of combinations and interactive experiences. For example, in Wave Curve Mode, all three Shimis use the same gestures but in different heights. The choreography framework provides an extendable and flexible way for designing more gestures and interactions in the future.

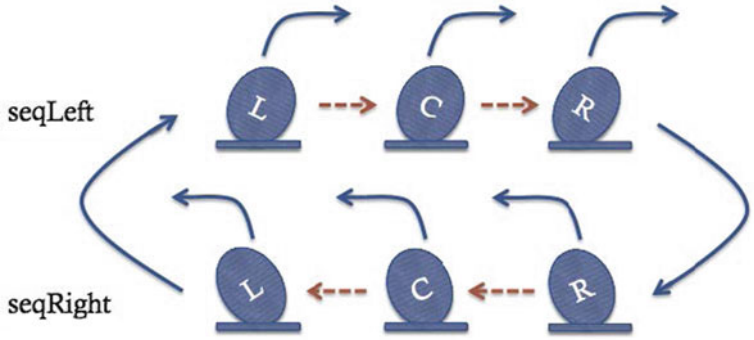


Fig. 3.27 Single sequential design

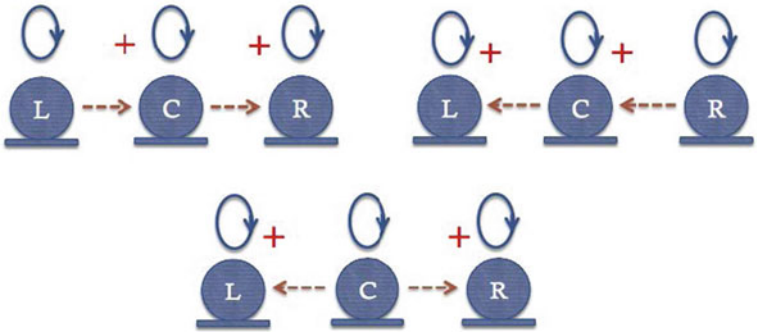


Fig. 3.28 Accumulative sequential design

In designing the mapping between the analyzed music and the robotic gestures we aimed to utilize Shimi's artificial musical intelligence to create expressive and musically relevant choreography. Our guideline for the mapping called for the gestures to reflect the musical nature of the song in a humanly intuitive manner, while also surprising the audience with unexpected improvisation from time to time, to increases audience engagement. Since the system was designed to perform in real-time for any given song, we set a hard limit of 12 ms for the computation time in each function block of the system to accommodate fast tempo songs. This prohibited the implementation time demanding operations such as SVM classifiers for musical event detection. Therefore, a semi-supervised learning method was implemented for the mapping. A set of 20 songs with clear beat were used for training. After extracting the musical features in real-time, manual calibration was performed on each feature to determine its boundaries. A threshold was set for each feature to quantize the original value and a set of mapping strategies were designed based on the quantized feature vectors. Figure 3.29 shows some of the mapping rules. For instance, high vocal and low energy were mapped to generate soft solo gestures, while high melodics and high energy music were mapped to generate interactive energetic movements.

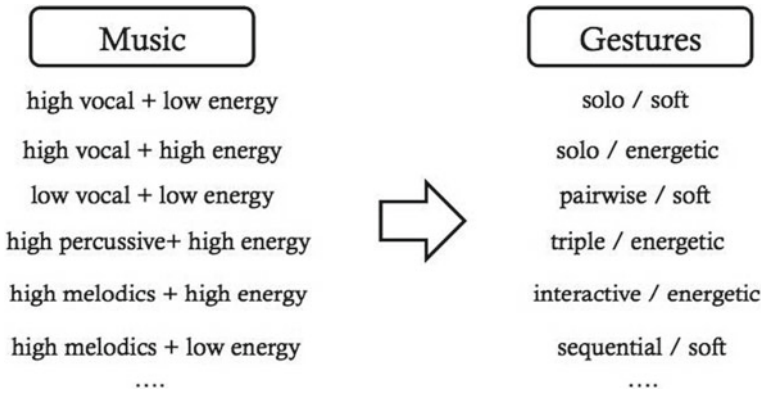


Fig. 3.29 Mapping strategies

The mapping design was guided by subjective aesthetic reasoning, rather than theory driven mapping.

An important characteristic of the Shimi Band is its capacity to improvise based on real-time listening. Each movement mode used an arsenal of gestures that could be chosen based on a set of stochastic rules. The degree of stochastic operation was determined by the style of the music. In some cases only one of the Shimis improvised while the other followed a preset sequence, and in other cases two or all three robots improvised. The level of improvisation was set by a centralized control framework, which led the band to generate a different performances every time, even for the same song.

3.5.6 Network Design

The instantaneous response of the Shimi Band to music stimuli required the networking communication to be precise, fast and computationally efficient. User Datagram Protocol (UDP) was used to connect the central computer with the Shimi Band via three separate lines to avoid interference or delays, although some level of message drop offs and latency was unavoidable. A set of strategies was implemented to ensure that the robots receive messages correctly without noticeable time delay, and that the system works properly and stably in different networking conditions.

In general, five types of commands were sent via UDP network to the Shimi Band, as can be seen in Fig. 3.30. "Tempo Change" notified Shimi that a new song was detected. The Shimi Band then stopped moving immediately and listened to the new song to detect a new tempo. To ensure that all of the three robots get the message at the same time, the command was sent twice with a slight time interval. When the tempo became stable or passed the time limit, a "Tempo Stable" message was sent to start Shimi's dance movement. This command was also sent twice to

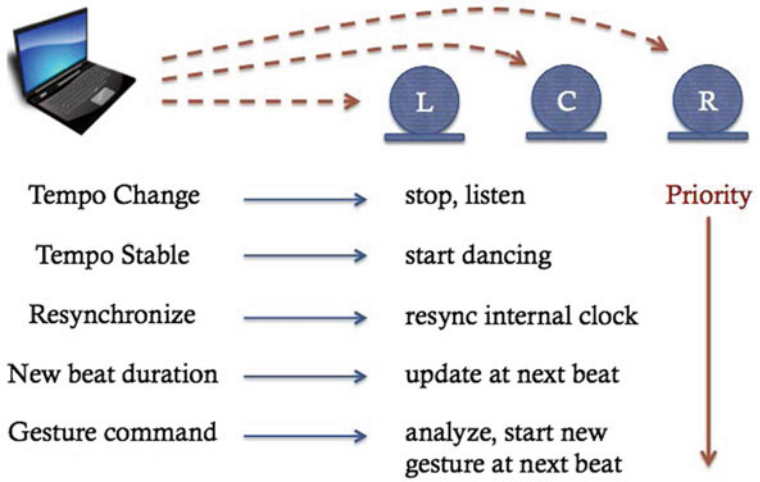


Fig. 3.30 Networking strategies

ensure the robots responded. Meanwhile, a new "beat duration" message was sent to the Shimi Band to update the robots internal clock at the next beat. A "Re-sync" message was sent to force the three robots to synchronize their movements during performance. This was necessary because the three Shimi phones internal clocks had slight differences, which led the band to go gradually offbeat when such slight differences accumulate. The "Resync" message was used to bring the three robots back to the same starting point every 16 beats. Gesture commands were generated every 1/4 of beat duration on the computer to guarantee that Shimi's gesture reflects the current musical characters. Each command was sent five times to make sure that the three robots received the instructions.

The centralized computer for the Shimi Band ran three main threads: 1. Play Thread, which was used for executing gestures and movements; 2. Listening Thread, which handled UDP messages and updated gestures accordingly; and 3. Update Thread, which was used for updating parameters like beat duration. To avoid sudden changes in the gestures, the beat duration and gesture commands were updated at the beginning of next beat after receiving the messages. The "Tempo Change" message received the highest priority, and could interrupt the current performance whenever it was received. Tempo Stable, New beat duration and "Resync" messages were sent on beat to keep the computer and the three robots at the same pace. Shimi also had a sleep mode to help saving energy, when no messages were received within a certain amount of time. The Shimi Band held performances at different places under various networking conditions. The networking strategies were effective in keeping the system stable and robust. In noisy environments, the Shimi Band could still detect the beat and dance properly.

3.5.7 User Study

Two comparative experiments were conducted to evaluate the system. In Experiment 1, 19 subjects, ages 5–30, were asked to watch a performances of the Shimi Band. Participants were told they could change songs and watch the robots dance as long as they wanted. After the performance, subjects were asked to answer 8 questions using a Likert scale from 1 (strongly disagree) to 7 (strongly agree). In Experiment 2, 16 Georgia Tech Music Technology graduate students were asked to watch the Shimi Band perform random gestures with no connection to the music. They were asked to fill out the same questionnaire after the performance. The questioner was designed to evaluate three aspects of the performance: synchronization and gesture expressiveness (Questions 1–3), music listening capacity (Questions 4–6), and general impression (Questions 7–9). Both between-subject and within-subject studies were conducted. Cronbach's alpha [24] measurement was used to check the internal consistency of the three target categories. The scores for Category 1 and 2 were all above 0.6, indicating that the questions in each category had high consistency and the test was valid and reliable. Category 3 had an alpha value below 0.6, so we treat the two questions (Q7 and Q8) separately.

The mean score for each category was calculated for each subject. The boxplot for Category 1 and 2 are shown in Fig. 3.31. The synchronization, music listening and general impression scores were higher in Experiment 1 than in Experiment 2.

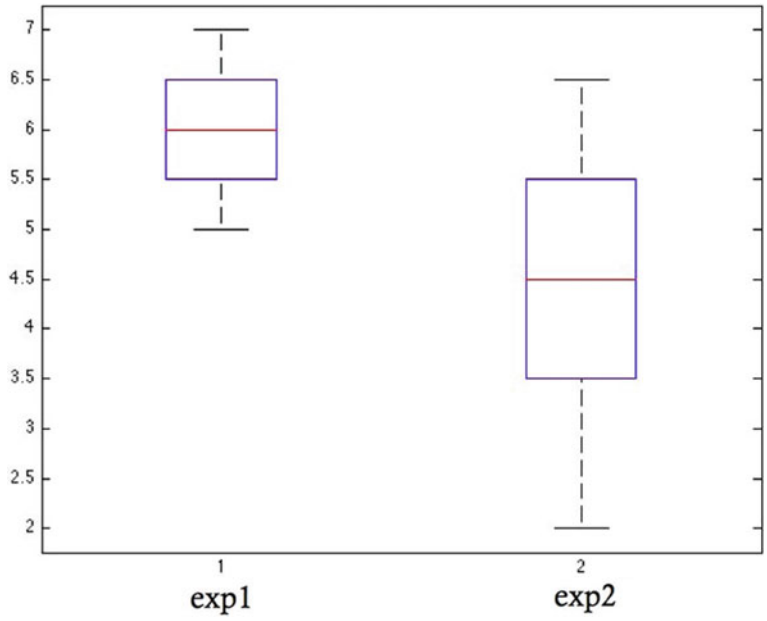


Fig. 3.31 Synchronization and gesture expressiveness

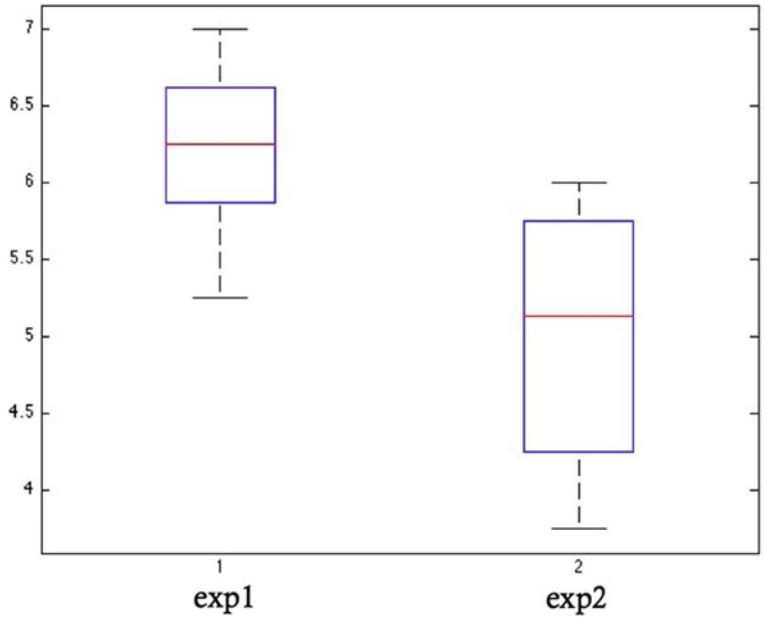


Fig. 3.32 Music listening

The variance of opinions in Experiment 1 was generally smaller. T-test was done to further study the difference level in the two experiments. The p values showed that the three categories in the two experiments were significantly different at 0.1% level, indicating that the new designed system had significantly improved the performance of the Shimi Band from synchronization, gesture expressiveness, and music listening perspectives (Fig. 3.32).

3.5.8 Summary

This project aimed to explore novel mapping and interactivity between music analysis and robotic movement by designing an intelligent dancing robot band that listens to music and responds with improvised synchronized gestures. A real-time MIR application was built for music listening and analysis. Several novel features were designed for robotic music interaction. New gestures and choreography design frameworks were implemented to increase the flexibility and audience engagement with the robotic band. The project has shown that through the use of music analysis, theory based and artistic gesture choreography, and efficient networking communication, the Shimi Band can perform in novel manners that are well received by audiences.

References

1. Puckette, Miller S., Miller S. Puckette Ucsd, Theodore Apel, et al. 1998. Real-time audio analysis tools for pd and msp.
2. Scheirer, Eric D. 1998. Tempo and beat analysis of acoustic musical signals. *The Journal of the Acoustical Society of America*, 103 (1): 588–601
3. Desain, Peter, H.J. Honing, et al. 2002. Rhythmic stability as explanation of category size. In *Proceedings of the international conference on music perception and cognition*, pages CD–rom. Sydney: UNSW
4. Tanguiane, Andranick S. 1993. *Artificial perception and music recognition*. Springer
5. Fred, Lerdahl. 2001. Tonal pitch space.
6. Eugene, Narmour. 1992. *The analysis and cognition of melodic complexity: The implication-realization model*. University of Chicago Press.
7. Elizabeth Hellmuth Margulis. 2005. A model of melodic expectation. *Music Perception: An Interdisciplinary Journal* 22 (4): 663–714.
8. Nattiez, Jean-Jacques. 1997. What is the pertinence of the Lerdahl-jackendoff theory? In *Perception and cognition of music*, 413–419.
9. Farbood, Morwaread Mary. 2006. *A quantitative, parametric model of musical tension*. PhD thesis, Massachusetts Institute of Technology.
10. Lerdahl, Fred, and Carol L. Krumhansl. 2007. Modeling tonal tension. *Music Perception: An Interdisciplinary Journal* 24 (4): 329–366.
11. Justus, Timothy C., and Jamshed J. Bharucha. 2001. Modularity in musical processing: The automaticity of harmonic priming. *Journal of Experimental Psychology: Human Perception and Performance* 27 (4): 1000.
12. Steinbeis, Nikolaus, Stefan Koelsch, and John A. Sloboda. 2006. The role of harmonic expectancy violations in musical emotions: Evidence from subjective, physiological, and neural responses. *Journal of Cognitive Neuroscience* 18 (8): 1380–1393.
13. von Helmholtz, H. 1954. On the sensations of tone (A.J. Ellis, trans.). Braunschweig: Vieweg & Son. (Original work published 1863).
14. Plomp, Reinier. 1964. Rate of decay of auditory sensation. *The Journal of the Acoustical Society of America* 36 (2): 277–282.
15. Hutchinson, William, and Leon Knopoff. 1978. The acoustic component of western consonance. *Journal of New Music Research* 7 (1): 1–29.
16. Vassilakis, Pantelis N., and K. Fitz. 2007. Sra: A web-based research tool for spectral and roughness analysis of sound signals. In *Proceedings of the 4th sound and music computing (SMC) conference*, 319–325.
17. Simon, Ian, Dan Morris, and Sumit Basu. 2008. Mysong: Automatic accompaniment generation for vocal melodies. In *Proceedings of the SIGCHI conference on human factors in computing systems*, 725–734. ACM
18. Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv:1603.04467.
19. Zen, Heiga, Keiichi Tokuda, and Alan W. Black. 2009. Statistical parametric speech synthesis. *Speech Communication* 51 (11): 1039–1064.
20. van den Oord, Aäron, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. 2016. Conditional image generation with pixelCNN decoders. arXiv:1606.05328.
21. Alexander, Lerch. 2012. *An introduction to audio content analysis: Applications in signal processing and music informatics*. Wiley.
22. Puckette, Miller, et al. 1996. Pure data: Another integrated computer music environment. In *Proceedings of the second intercollege computer music concerts*, 37–41.
23. Eden, Davies. 2007. *Beyond dance: Laban's legacy of movement analysis*. Routledge
24. Martin Bland, J., and Douglas G. Altman. 1997. Statistics notes: Cronbach's alpha. *BMJ* 314 (7080): 572.

Chapter 4
"Play Like A Machine"—Generative Musical Models for Robots

4.1 Abstract

In the previous chapter we discussed a number of approaches that allow Robotic Musicians to "Listen Like a Human" by modeling high level human musical percepts. In this chapter we focus on the second part of our Robotic Musicianship guideline—"Play Like a Machine". While implementing "Listen Like a Human" modules in Robotic Musicians allow robots to connect to humans in a relatable manner, "Play Like A Machine" modules are aimed at generating novel musical outcomes that are humanly impossible in an effort to surprise and inspire human collaborators. A well crafted balance between the two aspects of our guideline bears the promise of leading to musical outcomes and experiences that are both meaningful for humans and human aesthetics ("Listen Like a Human"), and novel, exciting, and inspiring ("Play Like a Machine"). We have explored two main approaches in our effort to develop robots that can play like machines. The first approach focuses on algorithmic generation of symbolic musical information and the second approach focuses on extended mechanical capabilities. In software, we looked at artificial processes that require algorithms that are foreign to human thinking, bearing the promise of creating novel aesthetic results. These include computational techniques such as genetic algorithms, mathematical construct such as fractals, and statistical modeling such as Markov processes and deep learning. In hardware, we developed mechanical abilities that are humanly impossible such as a robots that can control eight simultaneous arms, or strikers that can hit up to twenty hits per second, fast enough to create novel timbres and virtuosic polyrhythms. In this chapter we describe some of the approaches we took to create robots that play like machines—both cognitively and physically—in an effort to push musical experiences and outcomes to uncharted domains.

G. Weinberg et al., *Robotic Musicianship*, Automation, Collaboration, & E-Services 8, https://doi.org/10.1007/978-3-030-38930-7_4

4.2 Genetic Algorithms

One of our first efforts to create a robotic musician that can play like a machine was the implementation of Genetic Algorithms (GA) in the robotic percussionist Haile, which was retrofitted to play a small xylophone for this project. We chose to use GAs since these algorithms bear the promise of creating novel yet relatable musical outcomes. By employing random mutations and crossbreeding of "musical genes"—processes which are foreign to the human musical experience—GAs can lead to uncommon and surprising musical outcomes. But since each evolutionary generation of a "musical population" can be tested against a human-based musical fitness function, the musical outcome can be shaped by human aesthetics.

4.2.1 Related Work

A few efforts have been made to explore GAs for the purpose of music generation. GenJam [1], for example, uses GAs to improvise over a set of jazz standards in an interactive manner. The program applies a few musical constraints when generating the initial phrase population, while a human designer applies a "fitness function" that determines which phrases remain in the pool based on personal aesthetic preferences. Other examples using human-based fitness functions include Moroni et al. [2] where the fitness function is applied in real-time, and Tokui et al. [3], who uses human feedback offline to train an artificial neural network based fitness function. The Talking Drum project [4] on the other hand, utilized a computer generated fitness function that computes the difference between each member of the population and a target pattern. In comparison, our system uses a hybrid approach based on a human-generated initial phrase population with a computational similarity-based fitness function.

4.2.2 Method

We embedded our novel approach for musical genetic algorithms in Haile—an interactive robotic percussionist that was designed to respond to human musicians using two robotic arms playing a powwow drum. Unlike previous applications developed for Haile, here the robot was adapted to play a specially retrofitted one-octave toy xylophone (see Fig. 4.1). The diatonic scale xylophone (one octave rage from C to B) was placed under Haile's linear motor driven arm. The complementary 5 keys accidental kyes (C#-Bb) were placed under the right solenoid driven arm. Haile's responses were filtered by pitch class to accommodate the available one octave range.

The algorithmic responses developed for this project were driven by analyzed input from humans as well as on Haile's internal musical knowledge. First, the

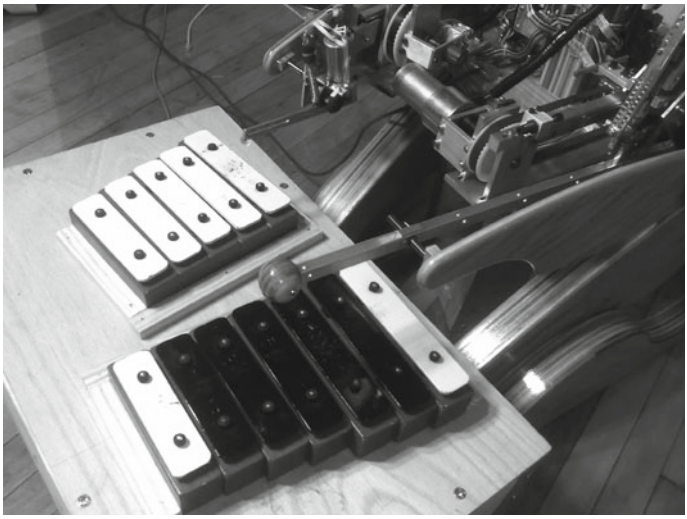


Fig. 4.1 Haile's two robotic arms cover a range of one octave (middle G to treble G) of a specially retrofitted toy xylophone. The left arm is capable of playing five notes, the right arm seven

algorithm listened to MIDI and pitch-detected audio and broke the input into short phrases. It then applied a fitness function in an effort to provide a relevant response by evolving a pre-stored, human-generated population of musical phrases. For each generation, a variety of mutation and crossover functions were applied over a variable number of generations. The fitness function measured similarity to the human generated input phrase. The most similar phrases were selected and extended through mutation and crossbreeding to form a new generation, while the least fit phrases were removed from the dataset population. Since we used a pre-generated population of phrases that evolved over a limited number of generations, some musical elements from the original population were integrated with musical elements from the real-time input to create hybrid, familiar but at times unpredictable, responses for any given input melody.

The initial population contained approximately forty melodic segments of variable lengths and styles, recorded by a jazz pianist. These phrases provided the GA with a rich pool of rhythmic and melodic genes that were later infused into Haile's responses. The fitness function was based on a similarity measure between the real time input and the base population dataset. In an effort to avoid generating an exact imitation of the input melody, we made a deliberate effort to limit the number of generations so that the fitness function does not converge to the ideal response by maximizing the fitness metric. By varying the number of generations and the type and frequency of mutations, we were able to preserve certain "musical genes" of the base population and combine them with the musical characteristic of the input melody.

For the similarity-based fitness function, we used Dynamic Time Warping (DTW), a well-known technique originally used in speech recognition applications. DTW

applies time shifting or stretching, to two given segments with variable lengths, making it possible for comparing two given melodies of potentially unequal lengths without referencing an underlying model. In our implementation, the function deviated from the time-frame-based model to represent melodies as a sequence of feature vectors corresponding to the notes, similarly to the implementation used by Smith et al. [5]. Similarly to Smith's "edit distance", our dissimilarity measure, assigned a cost to deleted and inserted notes, as well as to the local distance between the features of corresponding pairs. The function then chose the smallest distance over all possible temporal alignments, and used the inverse (the "similarity" of the melodies) as the fitness value. The weighted sum of four differences were used to compute local distances. These included absolute pitch, pitch class, log-duration, and melodic attraction. Individual weights were configurable, leading to a distinctive effect over the musical quality of the output. For example, higher weights on the log-duration difference led to more precise rhythmic matching, while weighting the pitch-based differences led to outputs that more closely mirrored the melodic contour of the input. We used Lerdhal's and Jaggendorf's Generative Theory of Tonal Music model [6] to calculate the level of melodic attraction between pitches. The relative balance between the local distances and the temporal deviation led to a lower cost for note insertion/deletion culminating in a highly variant output.

In an effort to utilize the DTW in real-time operation, we had to optimize the algorithm. To reduce computation time, we implemented a standard path constraint on the search through possible time alignments in which consecutive insertions or deletions were not allowed. This reduced computation time by approximately half, but prohibited comaprisons of melodies whose lengths differ by more than a factor of two. We address such situations as special cases by assigning an appropriately low fitness value. Additionally, since the computation time was proportional to the length of the melody squared, the algorithm broke longer input melodies into smaller segments, removing audible time lag and increasing overall efficiency.

In applying the fitness function, a configurable percentage of the phrase population was chosen to be reproduced in each generation. The better fit phrases were more likely to breed, as the selection was made stochastically according to a probability distribution calculated from each phrase's fitness value. The "mating" functions used a wide array of approaches, from simple mathematical operations to more sophisticated music-driven functions. For example, one non-music-driven crossover function allocated a random point on the two parent phrases and concatenated the first section from one parent with the second section from the other to create the child phrase. More musical mating functions were designed to crate musically relevant outcomes without converging the population to a maximized fitness value. An example for such a musical function was the pitch-rhythm crossover, where the pitches of one parent were integrated with the rhythm of the other parent. Since the parent phrases were often of different lengths, the new melodic pitches were linearly interpolated to fit the rhythm of the second parent (see Fig. 4.2).

To vary the population further, an adjustable percentage of each generation was mutated according to a set of functions that range in musical complexity. For instance, one simple mutation function added or subtracted random numbers of semitones to

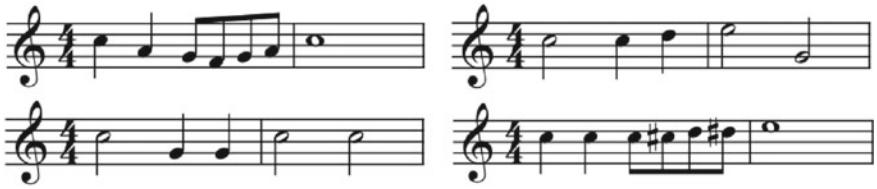


Fig. 4.2 Mating of two prototypical phrases using the pitch-rhythm crossover function. Child 1 has the pitch contour of Parent A and rhythm pattern of Parent B while Child 2 has the rhythm of Parent A and the pitch contour of Parent B

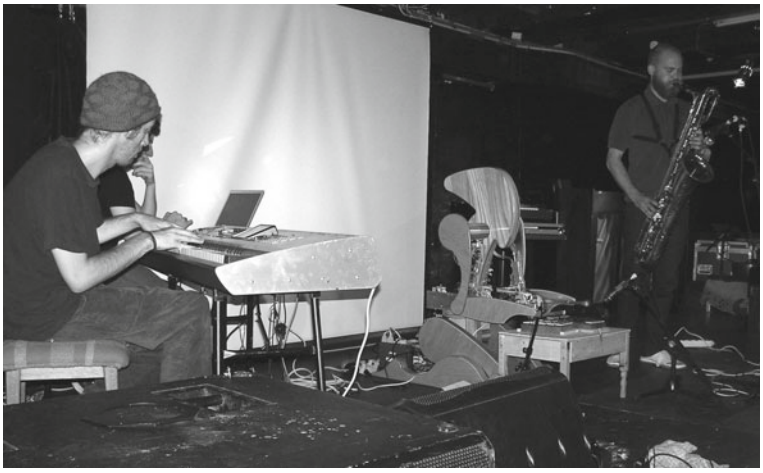


Fig. 4.3 Human players interact with Haile as it improvises based on input from saxophone and piano in Svobod (performed August 31, 2007, at ICMC in Copenhagen, Denmark)

phrase notes, as well as random lengths to note duration. This mutation was designed to add an amount of randomness that could allow a population to converge toward the reference melody over many generations. However, such a non-musically driven approach degraded the musicality of the intermediate populations. To address this problem we also used more musical driven mutation functions so that the outcome was recognizably a derivative of the original. The density mutation function, for example, altered the density of a phrase by adding or removing notes, while maintaining the original pitch contour of the melody. Other simple musical mutations included inversion, retrograde, and transposition operations. In total, seven mutation functions and two crossover functions were available for use with the algorithm.

The algorithm was used in a number of performances where a pianist and a brass player interacted with Haile, providing both MIDI and audio detected input melodies to Haile (see Fig. 4.3). The human players were instructed to listen to and respond to both Haile's GA driven musical output as well as to each others' improvisation.

4.3 Markov Processes ("Playing with the Masters")

The "Playing with the Masters" project features a generative improvisatory system that allows novice and proficient musicians to interact with the robot Shimon in a call-and-response manner, building on each other's ideas. As part of the project, Shimon was designed to listen to MIDI input and generate robotic algorithmic music responses by morphing between different styles of Jazz masters. This approach bears the promise of leading to a new form of human-robot interaction through adaptive and nonlinear weighting of past and present musical content, using a friendly mobile app interface. The project had three main goals: (1) allowing novices with little or no musical background to become engaged in interactive music making; (2) engaging users in social interaction with the robot; and (3) exploring personal musical styles of notable Jazz masters, while providing players with high-level control over the robot's musical responses.

4.3.1 Related Work

Related work can be divided into the field of interactive computer music systems and the field musical robots. In the first area, systems such the Continuator [7] use statistical models such as Hidden Markov Models (HMM) to analyze musical data. The Continuator then uses the product of the analysis to generate musical outcome in a similar style to the input, building memory of musical events and creating chains of musical material that humans can interact with. The Continuator can play a software synthesizer as well as a robotic Disklavier piano. The other areas of related work include musical robots such as Lemur bots [8] and the WF-4 flutist robot [9]. These musical robots are programmed to follow sequences generated by humans or computer code. Pat Metheny's composition Orchestrion, for example, employs two Lemur GuitarBot [8] which are programmed to mimic Matheny's guitar playing as well as follow pre-composed computer sequences. In this project we attempt to combine elements from these related research areas by using a Markov Process to drive a robotic musician, Shimon.

4.3.2 Implementation

To address the goals of the project, we developed an offline system that performs statistical analysis of music transcriptions of Jazz improvisation and uses this analysis to generate new music in a similar style. To facilitate interaction between Shimon and human collaborators, the system also incorporates music played in real-time by human performers into its generated musical output. In order to allow novices to interact with the robot, we developed a mobile app called Zoozbeat ([10]) that allows

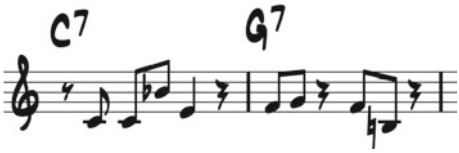


Fig. 4.4 Example of lead sheet/jazz notation)

users to generate musical sequences using an easy to use graphical user interface. Zoozbeat provides an intuitive method for both trained musicians and music novices to quickly and expressively create music and share it with Shimon (Fig. 4.4).

4.3.2.1 Model Design

The musical HMM (Hidden Markov Model) we developed searched for matches between pitches and rhythmic values in incoming note sequences and a chain of all notes that have been played and analyzed so far, in an effort to determine what note to play next. In addition, we classified the music according to a harmonic function in an effort to improve the quality of the musical generator. We, therefore, categorized the musical dataset into a series of chains, rather than using one single chain as was done in previous work. Furthermore, previous work has been designed for real-time interaction and therefore was optimized to process a limited amount of information. Our system, on the other hand, managed a large and varied corpus of transcribed improvisation, which required an efficient process of categorization in a timely manner. This could be achieved by letting the algorithm process the data in specific relevant harmonic contexts instead of using the full dataset.

To streamline the process we decided to annotate notes and chords for their the harmonic functional application (for example leading tone or dominant chord), rather than their generic musical description (for example a C-sharp note or F-minor chord). This allowed our system to treat any pieces of musical input, regardless of key. The system categorized the harmonic data according to the function of chord changes (e.g. tonic, dominant, etc), which improved the improvisation generator by providing it with harmonic context.

For the offline analysis, we used two types of files—a MIDI file and a chord changes score file which indicated chord changes in numerical representation. The algorithm separated MIDI information into series of pitch and rhythmic data points. To represent MIDI pitch information as function rather than absolute pitch, the program uses Eq. 4.5, which converted pitch material into twelve pitch classes. This allowed the algorithm to process note function relative to the scale. This way, the note D as the root of a F-scale was treated equivalent to the note D as the root of a D-scale. With each chord change, the program added a new line to the file, indicating a new chain of events for the particular chord context. The sequence was added with all of the scale pitches for the particular chord change. A similar process was designed for rhythmic values of each pitch (Fig. 4.5).

$$F(p)=\begin{cases}(p-S)\%12 & if\ p-S\geq 0\\ (p-S)\%12+12 & if\ p-S<0\end{cases}$$

Fig. 4.5 For pitches in scale-reduced form, S is the numerical pitch class of scale root (0–11) and p is the MIDI pitch number)

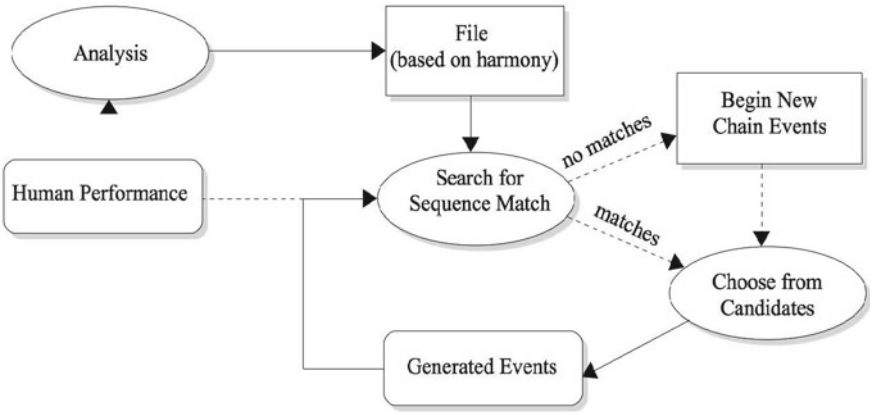


Fig. 4.6 Block diagram of playing with the masters algorithm, first, pitch and rhythmic input from the human performance are collected, followed by harmonic context analysis. The program retrieves all files that correspond to the determined harmonic context. The files contain a sequential order of all of analyzed notes and the selected harmony. A search method finds every occurrence of sequences in these files which match the input notes

4.3.2.2 Real-Time Processing

The algorithm listened to the human performer in real-time and analyzed the incoming data for scale and harmonic structure (see block diagram in Fig. 4.6). It then saved the incoming sequence in memory in order to expand upon on it during improvisation. The system relied on scale-reduced notation in order to equate notes that have the same function but appear in different keys. Our key detection algorithm is based on Carol Krumhansl's theoretical model of pitch profiling [11]. This model used a histogram of the most recent pitches performed, and matched it to the profile of a major and minor scale for all twelve roots. The best match determined the key. The algorithm used a similar method to determine the current chord and harmony by comparing the most recent notes with every possible major, minor, and diminished triad. The algorithm resolved the current harmonic context by scoring each triad based on its matches with recent played pitches. To generate the improvised sequences the algorithm utilised stochastic models based on Markov models, where previous events influence the likelihood of next event.

Figure 4.7 illustrates an example for the real-time process, where the input note sequence is F – A – B. Based on this input, the algorithm found a number of matches, including sequences B, A – B, or F – A – B. The note succeeding the matched

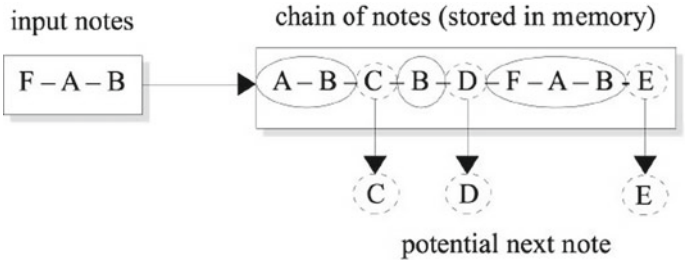


Fig. 4.7 Markov-based search sequence. Solid circles represent matched notes and broken circles represent potential output notes)

sequence could be any one for the marked notes in the sequence. For instance, when the algorithm detected the first match (A – B), C became a candidate for the first next note. The algorithm selected one of the possible matches stochastically. When informally testing the system, we found that low-order matches (one or two note sequences) yielded more novel musical phrases, which is expected since low-order matches are less bound to the original musical material. High-order models (three and four note sequences) reflected the original analyzed musical more reliably, leading to less novel output sequences. Since we wanted to use both of these characteristics, we developed a variable-order system that could benefit from the strengths of both low- and high-order matches. In order to represent the original analyzed music more accurately, the algorithm could emphasize high-order matches, which increased the likelihood of the program to select high-order rather than low-order matches. After selecting a matched sequence, the system generated the following note (in this case E). This note was fed back into the system, before the next search.

4.3.2.3 User Interaction

In an effort to facilitate intuitive and expressive interaction for novices with Shimon, we used the mobile application Zoozbeat [10], which allowed users to generate melodic sequences using graphical and gestural interaction with the phone. For this project, we also developed a wireless module for Zoozbeat that sent pitch and rhythmic values as well as continuous control to the robot via WIFI.

Two different models were designed for user interaction. In the first model both human and robot generated musical outcomes, while in the second model, the human could manipulate and affect the robot's musical language. The first interaction model was based on call-and-response routines, allowing both the human and the robotic player to listen to each other in turn and to respond in an improvisatory manner based on the musical input from their peers. To start the interaction, a human used Zoozbeat to play initial musical material by tapping the interface and shaking the phone. The app let players play and refine their music until they were satisfied with it before sending it the robot. A single button press and shake of the phone sent the sequence to

Shimon. The algorithm then expanded on the received material in an improvisational manner using the Markov process described above. The robot continued to play and expanded on the musical input until the human player decided to take control back from the robot and entered new musical material. The human could also shake their phone to play along with Shimon using a variety of instruments. Taking control back and playing simultaneously with the robot were interaction modes that were based on traditional musical group play schemes. The robot's response could have inspired the human performer to create new musical ideas. Humans could also restart with a blank score, creating new ideas and sending them to the robot to restart the interaction with a new melody.

The second form of interaction occurred when the human performer took control and manipulated the robot's musical outcome using virtual sliders on the phone app. The sliders affected the robot's musical language by altering the level of influence of musical material from different sources, including the human performer, as well as the off-line analysis of transcribed improvisation by great jazz maters such as John Coltrane and Thelonious Monk. To change the relative influence, users could turn the GUI (Graphical User Interface) sliders up and down to manipulate probability coefficients in the Markov model. For example, if the user raised the Coltrane slider, matched sequences that originated from Coltrane transcriptions would be more likely to play.

4.3.3 Summary

We developed a system that analysed statistical probability from transcriptions of great Jazz master improvisations and provided control over their respective influence. By introducing transcriptions from certain improvisational artists, the offline learning system we developed allowed shaping of an improvisatory robotic musical language. After offline analysis, players could manipulate the weighting of the model to the analyzed music it learned, providing the robot a statistic-based modeling approach for the creation, modulation, and refinement of generative musical language.

By analyzing musical languages in an interactive context our system could provide novel insights about style and creativity. Future work can focus on studying how the robot responds to new unpredictable musical input from human performers and how the response to spontaneous musical events defines the style and the improvisational language. With real-time control over the musical influences that comprise the language, we can study their subtle nuances and assess Shimon's function as a musician rather than a robotic instrument or an algorithm that generates sound.

4.4 Path Planning Driven Music Generation

Embodied robotic musicians require musical generation methods that are based on the physical capabilities of the robot. In this section a proof of concept demonstrating the utility of an embodied musical cognition for robotic musicianship is described. The problem is approached as a path planning task and uses search techniques to develop solutions that jointly optimize for the robot's physical constraints and musical semantics. The necessity for planning is demonstrated in an experiment that evaluates the planning algorithm against a baseline algorithm which makes locally optimal decisions (across physical constraints and musical semantics) in a greedy fashion. This experiment addresses the optimality goal of embodied processing. In the second part of the chapter the goal of musical emergence is addressed in the context of jazz improvisation. A knowledge-based implementation of jazz is described and the resulting generative musical path planning system is capable of creating different musical motifs in which the source of variance stems from the physical constraints. This allows for efficient and autonomous exploration of the relationship between music and physicality and the resulting music that is contingent on such a connection. The system is qualitatively examined and applied to the Shimon robot and used in performance settings.

4.4.1 Search and Path Planning

Computational tasks that attempt to find optimal solutions or sequences are a hallmark of artificial intelligence. Often what makes the task difficult or interesting is computational intractability. NP-complete puzzles require clever methods to prune pathways and nodes in order to make the problem solvable (or at least capable of finding sufficient local solutions). A classic example is that of chess playing computers. Of course, a chess game can be considered solvable because every possible sequence of moves in every possible game amounts to a finite number. However, this finite number is so large that a 'brute-force' method of a complete game is not feasible. Instead, a brute-force method of the next n possible moves is computed with emission scores that rely on metrics evaluating the strength and potential of different pieces and their respective locations, as opposed to simply relying on whether the move lies on a path that leads to winning [12]. The pathways and next moves are then chosen using different search algorithms (including minimax, A*, iterative deepening, depth-first search, and alpha-beta pruning) that jointly optimize for many parameters or metrics.

A similar method is used in this work. However, in order to function properly in a musical context the appropriate state space and useful metrics must be established. Just as the number of possible games of chess is insurmountably huge, the number of note sequences that can be composed is equally massive. Moreover, a pianist can choose to play a middle 'C' with any one of his or her ten fingers; including physical

parameters expands the possible options, thus, the decision-making process becomes more expensive. Therefore, good solutions must be identified without a brute-force search over all possible state sequences. The following sections describe a strategy for achieving this.

4.4.2 Musical Path Planning

As a first step towards demonstrating the effects of integrating the cognitive and physical in music generation a general path planning method is needed. Ignoring the goal of musical emergence and excluding generation for the moment, the first task is to develop a strategy so that a robot can perform a precomposed set of notes. Not only should the robot perform these notes, but it should do so in manner such that it avoids physical damage to itself, energy is conserved, and the perceptual advantages of visual cues (for people witnessing the performance) are not lost (i.e. avoid spurious movements). The following traits, listed in order of importance (for this system), are considered:

1. **Physical Harm**—Avoid catastrophic movements such as limb collision.
2. **Music Quality**—Maximize the number of notes it plays in the precomposed sequence.
3. **Perception**—Avoid spurious movements to maintain effective sound-producing visual cues.
4. **Efficiency**—Minimize distance traveled, torque applied to the motors, or power consumption.

4.4.2.1 Musical C-Space

In order to make optimal decisions that address the above factors a map representing the planning environment is needed, also known as the configuration space or *C-space*. In this case, the environment consists of possible musical notes as well as the physical configurations of the robot. Though music can be thought of as a continuous space in terms of intonation and timing, typically, it is represented as a discrete space. The notes on a musical score are represented by a finite number of possibilities of pitch and duration and mapped out according to the their absolute temporal position (again finite) within the context of the score. Even dynamics and tempo are discretized in score representations (fortissimo, mezzo forte, piano, presto, andante, largo, etc.) and it is the conductor or performer's interpretation that serves as a function to project these cues into a continuous space. Here, music is represented discretely as individual notes, and, in an attempt to minimize complexity, only pitch and rhythm are addressed. Musical features including dynamics, articulation, and temporal expression (rubato, accelerando, etc.) are not represented in the map.

Similarly, the poses and movements of a robot can be represented continuously, however, the physicality is reduced to a discrete space in order to integrate with the musical representation. Discretizing the C-space is highly typical in search and path planning methods for robots. The discrete space is usually represented using Voronoi diagrams, regular grids/occupancy grids, generalized cones, quad-tree, and vertex graphs [13]. A single state in this space may describe a posture, orientation, or even a motion primitive depending on the task. In this work the manner in which the space is discretized is motivated by the objective of the robot's ability to play a sequence of notes. Therefore, a single physical state should somehow be integrated with the available musical states.

In this implementation, a single state represents the pitch or pitches that can be reached given a specific physical configuration. The complete C-Space describes all possible configurations of the motors that can be used in order to play all possible pitches. As an example the design and specific physical constraints of Shimon, a four armed marimba playing robot, is considered [14]. An example of a state is shown in Fig. 4.8. Velocity is not addressed in the C-Space and duration is addressed using a

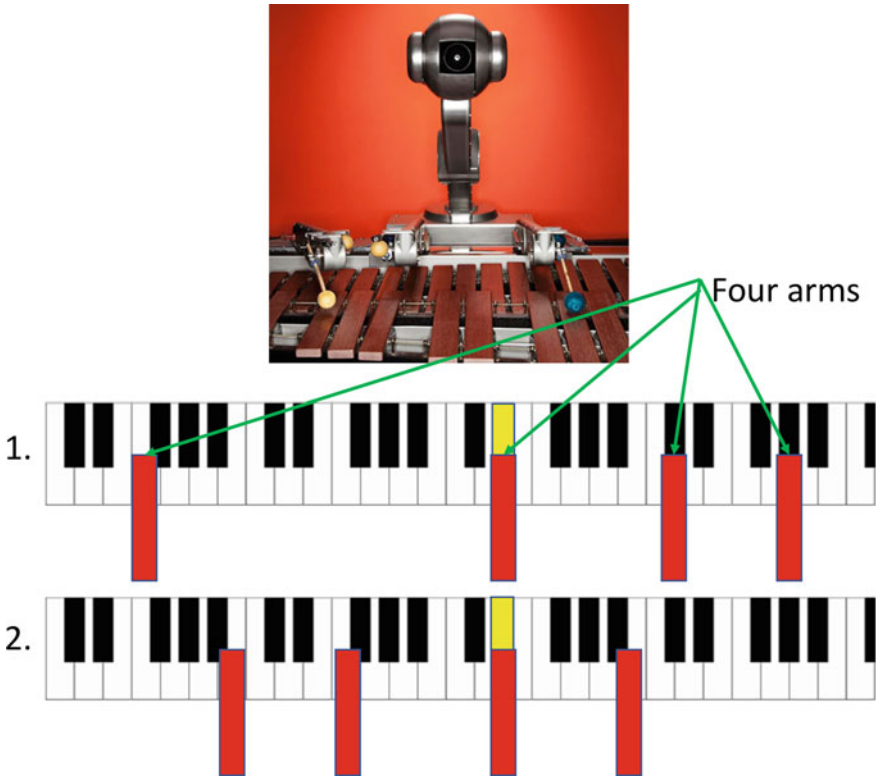


Fig. 4.8 Shimon has four arms which can be configured in many possible ways. A single state in the C-Space describes the possible physical configurations of the arms capable of playing a specific note(s). In this figure two states in which Shimon can play the highlighted 'Eb' are shown

transition function between states (described more in the next section). Additionally, Shimon is a percussionist so the duration of a single strike is constant, though the resting space between notes is variable allowing for rhythm. Such a constraint is analogous to a trumpet or clarinet player using only staccato articulated notes.

4.4.3 Planning

Now that the state space has been established a method for choosing particular states is needed (Fig. 4.9). While the designs of many robotic musicians prevent them from damaging themselves, Shimon's four arms share the same physical space making propioception integral for damage prevention. Alternative designs in which a solenoid is permanently stationed over every key do not pose such a risk. However, for any system in which the moving parts share the same space a form of path planning is necessary. Shimon's design limits its ability to play certain musical passages. For example, because Shimon's arms move on only one axis some transitions at certain speeds are impossible such as pitch intervals less than a minor third with temporal intervals less than 130 ms. This is due to two factors: (1) the speed at which a single arm can transition from one bar to another (130 ms for a half step) and (2) the width of Shimon's arms is larger than a single marimba bar meaning two arms can't previously be positioned to play a minor second very quickly. Defining the physical constraints that impose these limitations in a manner a computer can understand is essential.

The cost of each possible state is derived from the attributes enumerated above (physical harm, music quality, perception, and efficiency). It is computed using a combination of each state's instantaneous (or emission) score as well as a transition score describing the movement from one state to another. The variables involved include the state space, S, observation space, O, sequence of observations, Y, transition matrix A, and emission matrix B. When playing a precomposed set of notes the observation space is made up of all the possible pitch and rhythm combinations.

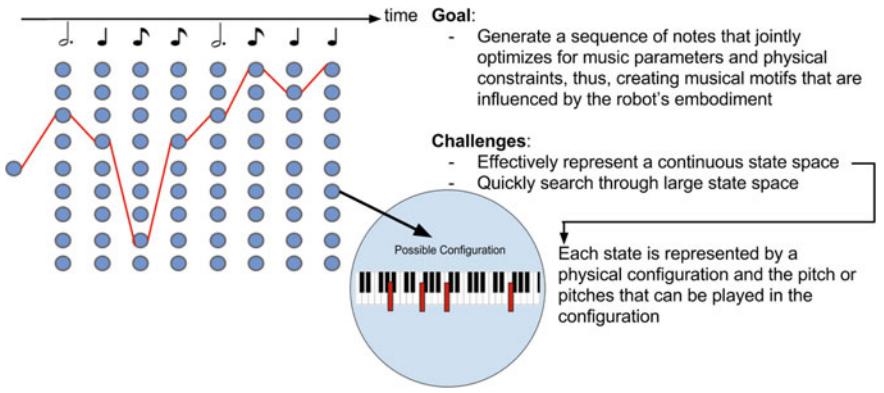


Fig. 4.9 General premise for finding a path through states in the configuration space

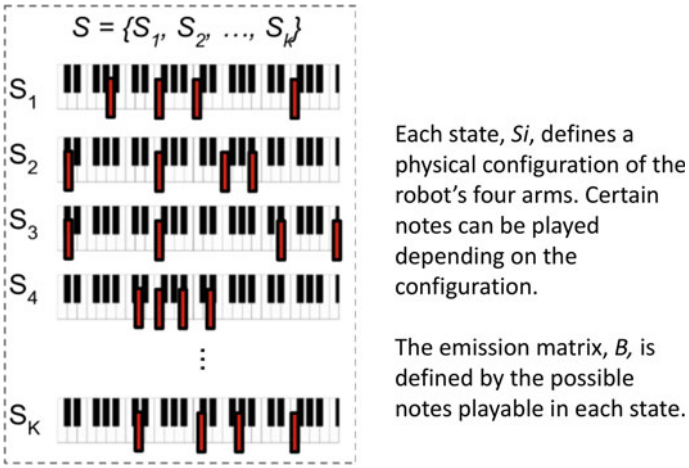


Fig. 4.10 State representation and emission score

The composition then represents the sequence of states within this space. Transition and emission matrices are typically defined as probability matrices that describe the likelihood of certain events occurring. In path planning these matrices do not describe probabilities, but rather binary descriptors of whether an event is possible with added heuristics to encourage specific types of behavior.

S is the state space $S = \{S_1, S_2, S_3, \ldots, S_k\}$, where k is the total number of states and S_i denotes the ith state in S (see Fig. 4.10). Recall, that a state represents a physical configuration of Shimon's arms. The emission score of a given state, B_{S_i}, is described by its ability to play the pitch in note(s), Y_n, where Y_n is the nth set of notes to be played in the observation sequence.

$$B_{S_i} = \begin{cases} \alpha & Y_n \in S_i \\ 0 & Y_n \notin S_i \end{cases} \tag{4.1}$$

where α is a weight describing the strength of the parameter. For playing precomposed note sequences there is only one instantaneous parameter, which is the musical heuristic or *musical quality* parameter. In this scenario the value simply depicts whether the pitch of a given note can be reached by the physical configuration represented in the current state. The weight, α, and subsequent weights are determined empirically on a held out test set.

A state only describes a static physical configuration and the possible pitches that can be reached from that configuration. Therefore, to consider attributes pertaining to time (such as note intervals) a measure describing the transition between states is necessary. Though this is computed as a matrix A, this matrix is not static, but tempo and interval dependent and thus must be recomputed for different tempi. Therefore, the transition matrix, A, can also be thought of as the result of a function that evaluates the quality of transitions between states. The quality is determined according to the physical harm, perceptual, and efficiency heuristics described below (Fig. 4.11).

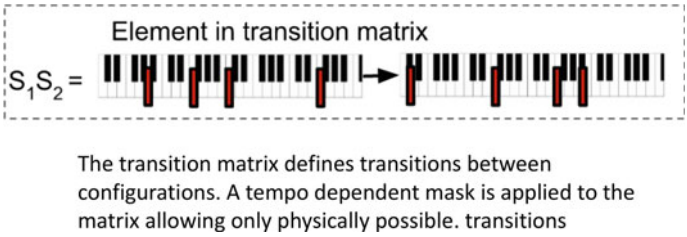


Fig. 4.11 The transition between two states

Physically possible transitions, as allowable by the design of the robot, can be determined a priori and formalized in a matrix, P. This matrix can be thought of as a mask of zeros and ones where a value of one denotes a possible transition and zero an impossible transition. Here, a penalty is imposed on the cost for attempting impossible transitions. The physical harm heuristic, h, describing the transition between state, S_i, and state, S_j, is computed as:

$$h_{S_i,S_j} = \begin{cases} 0 & for\ P_{S_i,S_j} = 1 \\ -\beta & for\ P_{S_i,S_j} = 0 \end{cases} \tag{4.2}$$

where β is a weight describing the strength of the parameter. Though this metric serves to penalize and prevent harmful movements, in practice, however, it is better to use a method that *guarantees* prevention of collision. This can be done by pruning the impossible states from the search and selection process, thus, preventing the system from ever making a bad move.

A significant argument for robotics in music is the presence of visual cues associated with sound producing movements which allows interacting musicians to anticipate what the robot is going to play. For example, when a pianist moves his or her hand to a particular position over the piano, the viewer can reasonably assume that he or she is going to play in the pitch range defined by that location. The perceptual heuristic is an attempt to minimize the number of times a movement associated with sound production does not produce sound. At times, this may be necessary to avoid collision. In Shimon, for example, movement of multiple arms may be necessary to play one note (recall that the width of Shimon's arms are bigger than the width of a marimba key). Finding paths that reduce instances of this will help preserve a person's ability to utilize the gestural cues and visually anticipate what will be played. In a perceptually optimal transition the cardinality of the set describing the difference between the transitioning sets, $S_i \rightarrow S_j$, should be equal to the cardinality of set, O_n. Therefore, the perceptual heuristic, p, between two states can be defined as:

$$p_{S_i,S_j} = \begin{cases} 0 & for\ |S_j - S_i| = |Y_n| \\ -\lambda & for\ |S_j - S_i| > |Y_n| \end{cases} \tag{4.3}$$

where λ is a weight describing the strength of the parameter.

Efficiency should be considered when there is a measurable amount of energy associated with a movement. This is largely defined by the robotic system and instrument design. For Shimon, the distance each arm travels is directly correlated with the amount of power drawn for each motor of its motors. However, for a device like the robotic drumming prosthesis the values of the PD controller may predict power draw [15]. The efficiency heuristic between two transitioning states is just defined as a variable, d_{S_i,S_j}. For Shimon, specifically, it is defined as:

$$d_{S_i,S_j} = -\omega \sum_{n=1}^{4} |\mathbf{S_{i_n}} - \mathbf{S_{j_n}}| \tag{4.4}$$

in which the distance traveled (in terms of pitches) is summed across all four arms and ω is a weight describing the strength of the parameter.

The final transition between two states is then defined as:

$$A_{S_i,S_j} = h_{S_i,S_j} + p_{S_i,S_j} + d_{S_i,S_j} \tag{4.5}$$

And the total cost, C, of transitioning from $S_i \rightarrow S_j$ is defined as:

$$C_{S_i,S_j} = B_{S_j} + A_{S_i,S_j} \tag{4.6}$$

The proper weights for each parameter are needed in order for the system to perform with the desired behavior. In this implementation preventing physical harm has the highest priority followed by playing the most notes, preventing spurious movements, and finally movement efficiency. To achieve this prioritization the weights must be such that $\beta > \alpha > \lambda > \omega$. However, in the actual implementation β is not actually used because transitions that would cause a collision are simply removed from the state space. The other weights were manually assigned based on empirical testing with $\alpha = 1.0$, $\lambda = 0.5$, and $\omega = 0.1$

4.4.3.1 Greedy Selection

As a baseline method, a greedy search method is implemented. In a greedy search method the cost, C, is used to make selections regarding movements deterministically for every single note. Once the selection has been made the robot makes the necessary movements to play the note and moves on to the next note. The only information that is considered is the current configuration state and the possible next states to which it can transition. With Shimon, one more musical heuristic can be added—the possibility to play notes in alternative octaves. When a certain note cannot be reached then the note's pitch can be re-evaluated in different octaves. The octave transposition results in note sequences that more closely resemble the rhythm of the original sequence, but may not be desirable and does not need to be performed. The greedy search algorithm is described in pseudocode below.

Algorithm 1 Greedy Selection

```
1: function GREEDY(S, current, y) : X
2:
3: // Given the current state evaluate all next possible states
4: // for the observation y
5:     for each state i from 1 : size(S) do
6:         V[i] ← Cost(current, S_i, y)
       X = arg max(V)                                  ▷ The best state.
7:     return X
```

4.4.3.2 Viterbi Search

The baseline does not find a global optimum given a sequence of notes, but rather makes greedy decisions at the rate of an individual note. In Shimon, the behavioral result is that when a note is received the arm closest to the note is chosen to move and strike the note. This may seem reasonable, however, in practice many moves are not possible because of speed limits and the width of each arm being greater than the width of a bar on the marimba (meaning the note cannot be reached without collision). To prevent collision between arms a note can be transposed to a different octave and the method is repeated. In more extreme cases in which no arm can play the original or a transposed version of the note then the note is dropped completely.

A more optimal solution can be found if the system has access to future notes. In this case, the algorithm can plan ahead so that an immediate move helps to set the robot up for success for playing subsequent notes. If the physical limitations of a robotic system are restrictive then planning for multiple notes at a time can be useful for creating a more suitable movement sequence.

Given the state space representation there are two suitable algorithms that lend themselves to this type of planning. The first is the A* (A-star) algorithm. A* is a pathfinding search algorithm that is guaranteed to find a solution if one exists. The algorithm uses heuristics to guide its search, however, it is necessary for the heuristic to be admissible. This means that the heuristic never overestimates the true minimum cost of reaching the goal. In music there is no clear admissible heuristic and without the heuristic A* is essentially breadth-first search. Instead, this work uses a beamed Viterbi search algorithm to generate an optimal sequence. Other than no admissible heuristic, the major benefit of Viterbi is that the computation time is stable. In musical applications timing is very important, therefore, having this stability can inform the developer how to best design various applications or interactions with the robot.

In order to decode the graph using Viterbi then the state space, S, observation space, O, sequence of observations, Y, transition matrix A, and emission matrix B are necessary. The algorithm traverses the states in the graph to generate multiple hypotheses that maximize the given parameters. These parameters are identical to the ones outlined above and used in the greedy selection process. The pseudocode for the implementation is written below and the general concept for finding a good sequence is provided in Fig. 4.12.

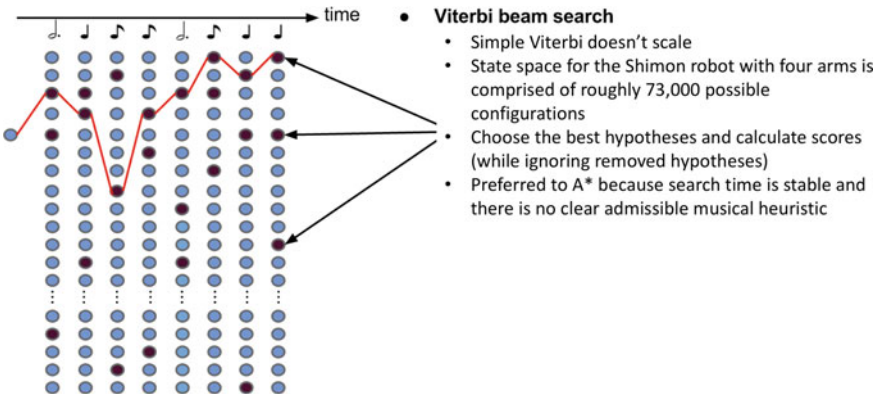


Fig. 4.12 A beamed Viterbi search is used to traverse the state space and find good movement sequences as defined by the heuristics

Considering that the total number of arm configurations is about 73,815 (each arm is capable of playing roughly 36 notes), finding the global optimal path is not always feasible if the path is to be found in a timely manner. Therefore, a beam is applied in order to prune unreasonable branches quickly. The beam narrows the search to only look at states capable of playing the observed note in the sequence, Y. This significantly reduces the number of possible states and allows the system to compute paths for precomposed note sequences quickly enough for interactive applications.

4.4.4 Evaluation

The effects of simulated action and planning can be measured objectively by comparing the Viterbi based metric to the greedy selection process. In the first experiment both algorithms were given one hundred pre-composed monophonic note sequences. The sequences came from a mix of publicly available classical melodies and transcribed jazz improvisations with high degrees of technical complexity such as Andersen's 18 Etudes for flute and Coltrane's improvisation on Giant Steps. The make of the dataset is shown in Table 4.1.

Using the two algorithms, note sequences, using the physical constraint parameters of Shimon, were generated. The rate at which each algorithm transposed a note and dropped a note was computed. Additionally, the average distance traveled over four beats of music was also computed (across all arms). The unit of measurement for distance is the width of a marimba bar, therefore, a distance traveled equal to one is equivalent to one arm moving by one note. A distance traveled equal to four can be equivalent to one arm moving four notes, two arms moving two notes each, four arms moving one note each, and so on. In order to remove a zero distance bias for

Algorithm 2 Viterbi Path Planning

function VITERBIDECODE($S, Y, A, B, start$) : X

// Initialize scores with starting state and first note in observation sequence
  for each state **i** from 0 : $size(S)$ **do**
    $V[1, i] \leftarrow B_{start,i} \cdot \max_k(V[start, k] \cdot A_{k,start})$
    $P[1, i] \leftarrow \arg\max_k(V[start, k] \cdot A_{k,start})$ ▷ Store back-pointers

// Compute scores over entire state lattice
  for each time step **i** from : $size(Y)$ **do** ▷ Iterate through observations
    for each state **j** from 0 : $size(S)$ **do** ▷ Iterate through each state
      $V[i, j] \leftarrow B_{ji} \cdot \max_k(V[i-1, k] \cdot A_{kj})$
      $P[i, j] \leftarrow \arg\max_k(V[i-1, k] \cdot A_{kj})$

// Trace the back pointers beginning from the best end state
// in order to identify the optimal state sequence
  $z_{size(Y)} \leftarrow \arg\max_k(V[i-1, size(O)])$ ▷ Identify best end state
  $x_{size(Y)} \leftarrow state_{z_size(Y)}$
  for each time step **i** from $size(Y)$: 2 **do** ▷ Trace backpointers
    $z_{i-1} \leftarrow P[z_i, i]$
    $x_{i-1} \leftarrow state_{z_{i-1}}$
  return X ▷ The optimal state sequence.

Table 4.1 Distribution of data in test set

Performer/composer	Number of pieces	Original instrument
Carl Stamitz	5	Clarinet
Carl Maria Weber	6	Clarinet
Joachim Andersen	18	Flute
Ernesto Kohler	12	Flute
Dan Adler	10	Guitar
John Coltrane	6	Saxophone
Dexter Gordon	8	Saxophone
Sonny Rollins	7	Saxophone
Sonny Stitt	14	Saxophone
Clifford Brown	5	Trumpet
Freddie Hubbard	9	Trumpet

when a note is dropped the distance is normalized by the number of notes played. The results are shown in Table 4.2.

The experiment was repeated, however, a set of physical constraints were used for a hypothetical robot. The new constraints were similar to those of Shimon, but instead the simulated system is equipped with two arms with each having a width equal to one half of one marimba bar (meaning it is possible for the two arms to play a minor second interval without moving). With this set of constraints there are no dropped

Table 4.2 Greedy versus planned experiment 1 results

	Octave transpose rate (%)	Drop rate (%)	Normalized distance
Greedy	39.2	14.9	2.8
Viterbi	11.1	1.7	3.1

Table 4.3 Greedy versus planned experiment 2 results

	Normalized distance
Greedy	4.8
Viterbi	3.4

or modified notes as it is possible with both the greedy and planning algorithms to play all notes in the monophonic sequences. Energy efficiency as represented by total distance traveled (across all 100 note sequences) is examined. The results are shown in Table 4.3.

4.4.5 Discussion

The results demonstrate that there are benefits to planning versus on-line decision making. The planned movement sequences that are optimized for sequences of notes (such as a phrase) dropped much fewer notes and performed fewer octave jumps. The greedy method was slightly more energy efficient (according to distance traveled) in the first experiment. However, the second experiment was designed to explicitly evaluate distance traveled and energy efficiency in isolation. The system using Viterbi planning was roughly 30% more efficient.

4.5 Rule Based Jazz Improvisation

The integration of physical and musical components in a single state space for joint optimization is the paramount aspect of this work. Though Viterbi was the chosen algorithm it is likely that several search algorithms would have worked with some modifications to the state space. Ultimately, the efficacy of the search algorithm depends on two factors (1) the ability to represent a state such that it adequately encapsulates all relevant components and (2) the ability to evaluate states and state transitions with meaningful metrics.

The first factor was achieved with a representation that is relatively simple to interpret. The optimal path through the state lattice describes the sequence of physical configurations necessary to play the observed sequence of notes. The metrics were shown to be effective for the task of playing precomposed music, however, the musical

heuristic (maximizing number of notes played) is not useful for generating music. In this section, the prospect of using a joint optimizing search methodology to generate music is explored.

4.5.1 *Parametrized Representations of Higher-Level Musical Semantics*

Search and path planning requires that there is some information guiding the search towards good solutions (or what is deemed good according to a pre-defined metric). For the Viterbi algorithm this information includes the sequence of observations and the metric stating how well the path through the lattice explains or represents the observations. Previously, the observation sequence, Y, was a musical score, the observation space, O, constituted all the possible notes a score could contain, and the state space, S, constituted all the possible physical configurations of the robot.

For the system to create the note sequence as well as the movement sequence the state space must be expanded to include note information so that a single state combines a physical configuration and a specific note or notes to be played. The observation space and observation sequence need to be modified so that they encourage certain musical behaviors without explicitly dictating the notes to play. This is the 'musicianship' part of the problem.

The notion of dictating note sequences based on higher level musical features for jazz improvisation is markedly different from previous machine jazz improvisation systems. Typically, note-level models are developed using rule-based or statistical machine learning methods and any higher level musical intentions are an aftereffect [16]. Note-level models have been constructed using genetic algorithms, Markov chains, recurrent neural networks, and probabilistic grammars [1, 14, 17, 18]. Each of these systems uses note transitions as the building blocks for the generative algorithm. Though this may be sufficient for pure software applications, it is not the most desirable approach for a robotic performer that may not be able to play what the system generates because it provides no higher-level context for finding more suitable alternatives.

Note-level models may be preferred because they are convenient for training statistical models. The system in this work does not utilize machine learning and instead employs knowledge-based heuristics that describe higher level musical concepts. Luckily, several musicians and academics have extensively studied the approaches of great jazz musicians over the last seventy or so years and have used their techniques to define and teach modern jazz theory. This theory has been presented in numerous books and some of the higher level jazz concepts of this system are codified using these resources. Two books in particular (that have become canonical resources for jazz musicians) are the 'Jazz Piano Book' by Mark Levine and 'How to Improvise' by Hal Crook [19, 20]. The semantic concepts that are quantified in this system include harmonic tension as it pertains to scale theory, pitch contour, rhythmic complexity, and note density. The system chooses notes that prioritize the observation sequences describing these features while considering its physical constraints.

4.5.1.1 Harmonic Tension and Color

Arguably the most important feature for determining the pitches in a jazz solo is the chord progression. In fact, Johnson-Laird argues that the two most significant constraints for choosing pitches is the underlying chord progression and the contour considerations [21].

At the most basic level, harmonic theory describes that a certain scale be played over a certain tonal center or more specifically a certain chord. For example, a 'C major' scale can be used to play over a *Cmaj7* chord or a 'G mixolydian' scale can be played over a *G7* chord. The 'C major' and 'G mixolydian' are actually comprised of the same notes as the *Cmaj7* chord is the 'I' chord (or tonic) of the C-tonal center and the *G7* functions as the 'V' chord (or dominant). In popular music the tonal center can usually be identified by just looking at the key signature of the piece. In jazz, however, the tonal center may modulate frequently throughout the course of a piece and the improviser must identify the tonal center given the chord sequence. It is not obvious what the tonal center is given a single chord because one chord can serve different harmonic functions depending on the tonal center. For example, a *D-7* can be the 'ii' chord of a progression with a tonic of 'C major' or it can function as the 'vi' chord of a progression with a tonic of 'F major'. Fortunately, there are common sequences of specific chord functions such as *ii-V-I, IV-V-I, or V-iv-i* that allow musicians to make reasonable assumptions where the tonal center lies. These sequences are referred to as cadences. Similarly to human interpretation strategies, a computational system can utilize this information to identify the tonal center.

To label the tonal centers and chord functions for each given chord a system can look for common cadences and find the optimal chord function sequence that maximizes the use of these cadences. This type of problem is referred to as a sequence tagging problem. It is similar to identifying the parts of speech of words in a sentence and, in fact, it is another type of problem well-suited for Viterbi decoding. There is no dataset to train a model to find the proper weights. The system described in the remainder of this section will instead tag sequences using hardcoded priors based on what is most frequently used in classical and jazz music.

In this case, the state space is comprised of tuples consisting of all possible combinations of tonal centers and chord functions. Scales representing each mode derived from Major, Melodic Minor, Harmonic Minor, Harmonic Major, and Double Harmonic Major harmonies are defined. This means the system knows 35 scales, however, in tonal music stemming from classical and jazz only a small subset of these scales can serve as tonics. This is because in typical harmonies of tonal music the dominant triad is a major or minor chord. This limits the number of scales that can be used as the basis for the harmony.

Each of the eight tonics has seven possible scale degrees meaning there are seven chord functions within that harmony that can be played. These can be played in 12 keys making the total number of possible states equal to 672 ($8 \times 7 \times 12$). The transition matrix is 3-dimensional (to include cadences with a length of three) and gives weight to commonly seen cadences. These weights are not learned, but manually labeled using expert knowledge and then evaluated with a test set of ten jazz standards (Table 4.4).

Table 4.4 Frequently used chord sequences with manually labeled transition weights

Scale	Sequence	Weight
Ionian	IV-V-I	3
	ii-V-I	3
	V-ii-I	3
	iii-ii-I	1
	iii-IV-I	2
	V-I	3
	IV-I	2
	ii-V	3
	IV-V	3
	I-vii°	2
Aeolian	v-iv-i	3
	bVII-iv-i	3
	bVI-bVII-i	3
	iv-bVII-i	3
	bVI-v-i	2
	iv-v-i	1
	bVII-i	3
	v-i	1
Harmonic minor	ii°-V-i	3
	iv-V-I	3
	V-iv-i	2
	bVI-V-i	2
	v-bVI-i	2
	V-i	3
	ii°-V	3
	i-vii°	2
	i-V	2
Harmonic major	iv-V-I	3
	V-iv-I	3
	iii-iv-I	2
	V-I	2
Minor major	IV-V-i	3
	ii-V-I	3
	V-I	2
	I-vii°	2
Mixolydian b6	v-iv-I	2
	bVII-iv-I	2
	bVIII-I	2
Double harmonic	iii-bII-I	1
Major	bII-I	1
Hungarian minor	bVI-vii-i	1

The observation space is made up of the chord progression and melody of a piece. If there is no melody and only the chord progression is provided there is no instantaneous score to evaluate a single chord. However, when a melody is present the emission matrix describes the relationship between the notes in the melody that are played over a given chord. The score is derived by counting the number of melody notes that are part of the scale for the tonic and chord function being evaluated.

Finally, using the chord function state space, a transition matrix describing common cadences, and an emission matrix describing note/scale relationships Viterbi can be used to find the optimal sequence through the chord function and tonal center state space. With a small test set of ten jazz standards with labeled chord functions (labeled by one expert) it is possible to achieve greater than 95% agreement between the Viterbi tagged chord functions and those hand labeled in the test set. However, there is some subjectivity when it comes to tagging these sequences. Two experts are likely to agree on the majority of functions a specific chord serves in the context of a piece, but will also likely differ in some places. Differing interpretations can be useful and emphasis on certain centers and harmony is likely to be a key a component describing an individual's style. An example of the sequence tagging is shown in Fig. 4.13.

Understanding the tonal context of specific chord sequences in an entire progression is the first part of modeling tonal tension. The second part evaluates different pitches used with the specific chords and tonal centers. When human improvisers and composers evaluate individual pitches against a certain chord they are considering its context in the scale that is being used with the chord [19, 20]. The tonal center and chord function determines the scale and with this information the improviser can evaluate the tonal quality of individual pitches relative to the scale and chord. For example, in a *D-7, G7, Cmaj7* (a major *ii-V-I*) progression the D-dorian, G-mixolydian, and C-ionian scales are the scales that represent the C-major harmony for each chord in the sequence, respectively. Certain pitches are more likely to create dissonance when played over this progression.

This model is used to measure the tonal distance between a pitch and a chord. The model can be thought of as a nonlinear projection of the pitch space (similar to the circle of 5ths or tonnetz models) based on heuristics provided by scale theory and the current tonal center. See Fig. 4.14 for an outline of the model. The tonal model produces five levels of harmonic distance a pitch can have from a given chord from closest to furthest including: (1) the root pitch; (2) pitches other than the root that are in the chord; (3) available pitch tensions that can be used with the chord; (4) avoid pitches in the scale represented by the tonal center and chord; (5) all pitches outside of the scale represented by the tonal center and chord. These five levels are used to measure 'tonal color' or harmonic tension of an individual pitch at any given moment within the chord progression. Given a time series or observation sequence of tonal color values the system will find the best path that explains the sequence.

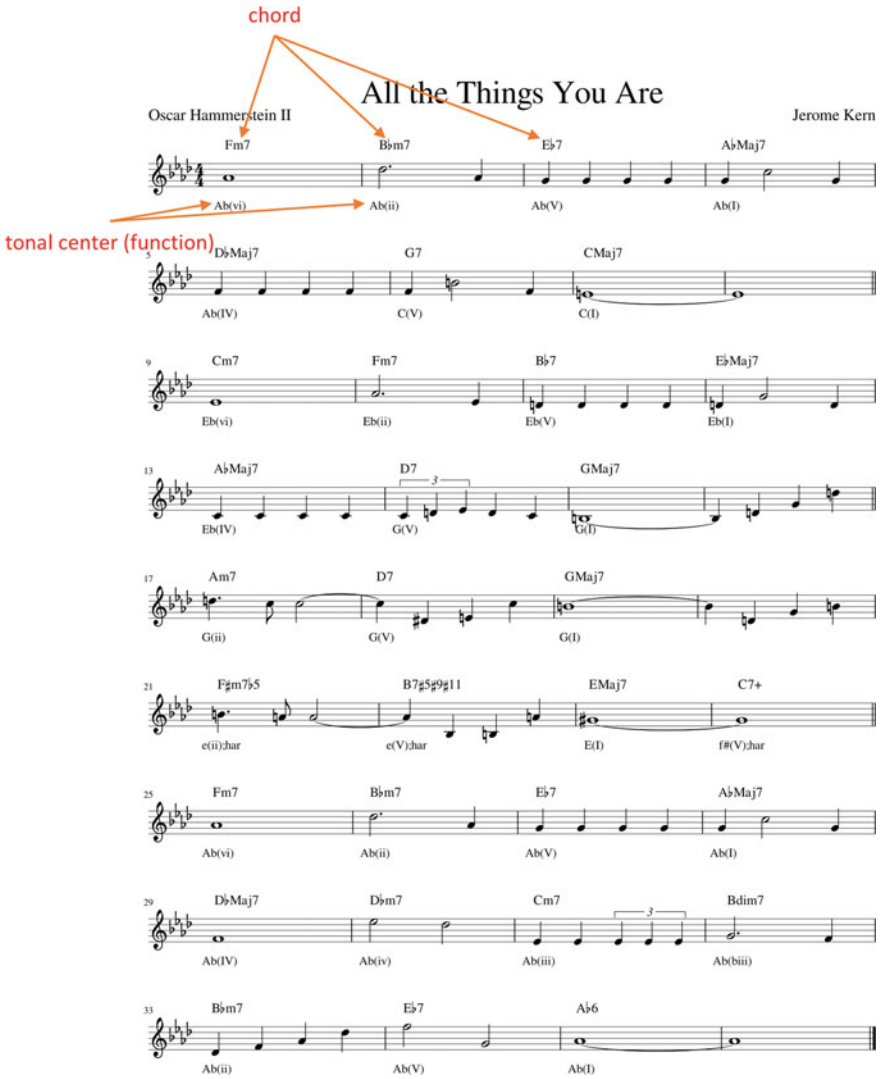


Fig. 4.13 The chords (annotated above the measure) are tagged with their corresponding tonal center (annotated below the measure) using automatic sequence tagging based on a trigram model. Viterbi decoding is used to find the optimal sequence

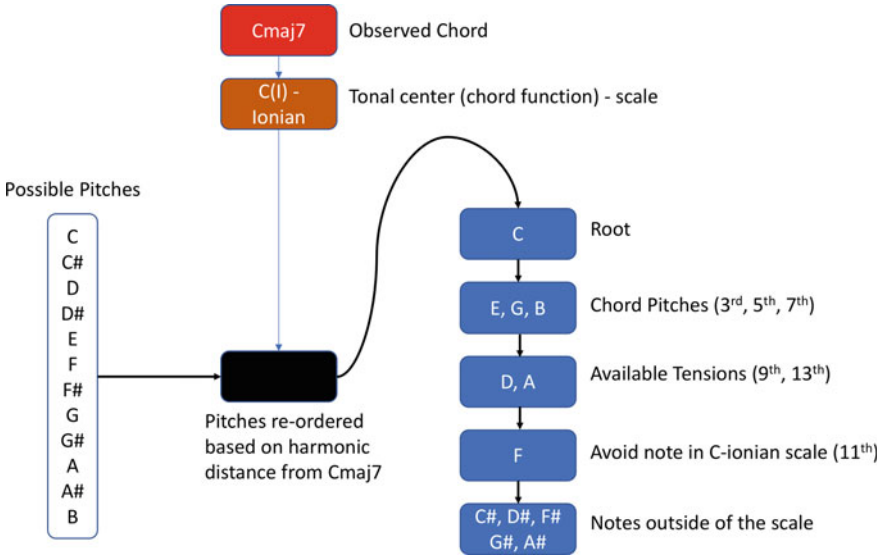


Fig. 4.14 Harmonic distance metric—the observed chord (Cmaj7) is labeled by the chord function Viterbi decoding process. The possible pitch classes are measured in terms of harmonic distance (or harmonic stability) relative to the particular chord function and tonal center. Using harmonic theory, the pitches are projected into a space that organizes them according to their stability over the chord function determined by the Viterbi process. There are five levels in this space and the level number is used to represent the harmonic distance of the pitch to the given chord. In this example, the root note, *C*, is the closest note to the 'Cmaj7' chord and the notes outside of the C-ionian scale are considered the furthest

4.5.1.2 Pitch Contour

The second constraint for determining the pitch is the overall contour and octave location. Here, the observation sequence is a time series representing the absolute pitch. The path planning will create a path that best describes this sequence. The emission metric used is simply the distance between the state's note and the value of the observation in absolute pitch space. The transition metric is simply a descending, ascending, or non-moving descriptor between two notes in the state space. There is a small penalty if the transition between two states does match the transition between the corresponding two observation states.

4.5.1.3 Rhythm

The note choices are also constrained by aspects of rhythm. The two factors used in this work to describe rhythm are note density and complexity. Rhythm choices are made on a per beat basis. The system contains hundreds of units encapsulating different rhythms that can be played within the duration of a single beat (Fig. 4.15). These

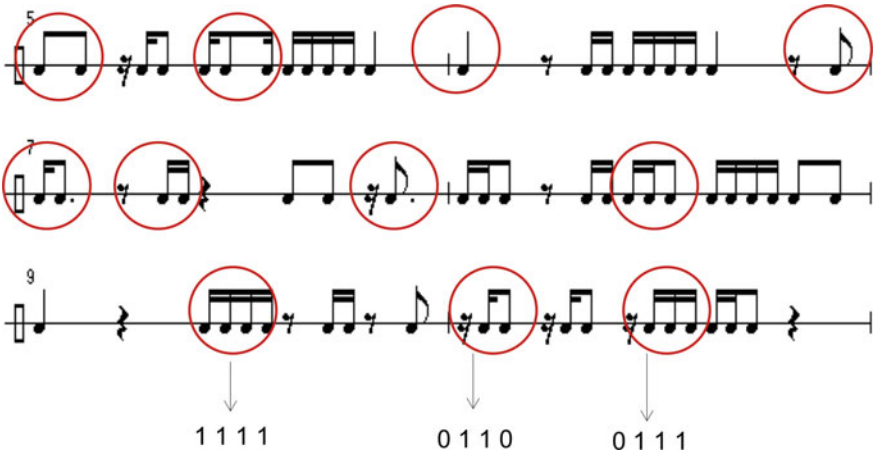


Fig. 4.15 Rhythmic decisions are made per beat. A library of hundreds of unique rhythmic units exist and the system pulls from this library. In this figure the unique rhythms are circled. The rhythms can be encoded as binary strings encoding only the onset and ignoring duration. They are shown as having a resolution of one sixteenth note, but the implementation provides up to 128th note resolution containing both duples and triples. The decision to remove durations was to reduce the dimensionality of the task and focus on percussionist robots such as Shimon

units were taken from examples in George Lawrence Stone's classic instructional drumming book *Stick Control: For the Snare Drummer* [22]. Note density describes the concentration of notes played within a finite period of time. The observation space for the note density feature represents this concentration per beat.

The complexity of an object describes the amount of information encoded in it. Therefore, complexity of a system can be measured by how much it can be compressed. One common way of measuring rhythmic complexity is to use Lempel-Ziv compression on a binary string [23, 23–27]. Similarly, a string can be represented as a finite state machine (FSM). The lower bound of the output length of a sequence generated by a finite state machine is equal to the upper bound of the output length of Lempel-Ziv algorithm. Here, a rhythmic unit is encoded as an FSM and the size of the FSM represents the complexity of the rhythm. This method can be thought of as a baseline for rhythmic complexity. It is capturing hierarchical redundancies encapsulated in the rhythmic times series, but does not specifically address the subjectivity and human perceptions of rhythmic complexity. For example, people often correlate rhythmic complexity with how difficult it is perceived to play rather than the hierarchical structure. However, it has been shown that this method of encoding does correlate, to a degree, with human perception and that is why it is considered a baseline [27].

In this implementation a single rhythmic unit complexity is measured using the value described by the size of the state machine. The transition between units can also be measured by creating an FSM of the entire concatenated rhythmic string across both rhythmic states in the transition. Because rhythms are represented in predefined

units their complexity and note density can be computed a priori. Additionally, rhythmic features are tempo dependent when considering the physical constraints of the robot. The rhythmic unit library can be pruned prior to path planning by removing impossible rhythms according to the physicality.

4.5.2 Joint Optimization

Previously, the Viterbi algorithm was used to find the physical movement sequence necessary to play a sequence of notes, Y, that was provided prior to the search. Here, there exists an observation sequence for each musical semantic (as shown in Fig. 4.16) and Viterbi is used to find the optimal note sequence derived from both the musical parameters and physical constraints. Therefore, the emission and transition matrices not only constrain decisions to the robot's embodiment, but also evaluate the note-to-note decisions based on their ability to satisfy the semantic goals.

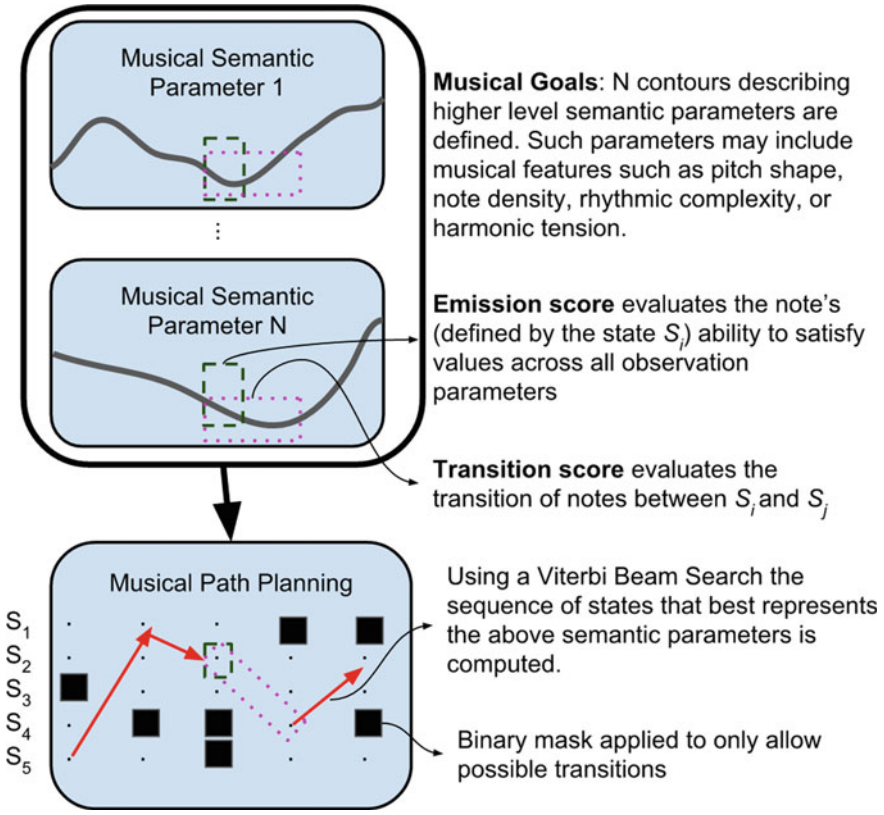


Fig. 4.16 Framework of the generative embodied music system

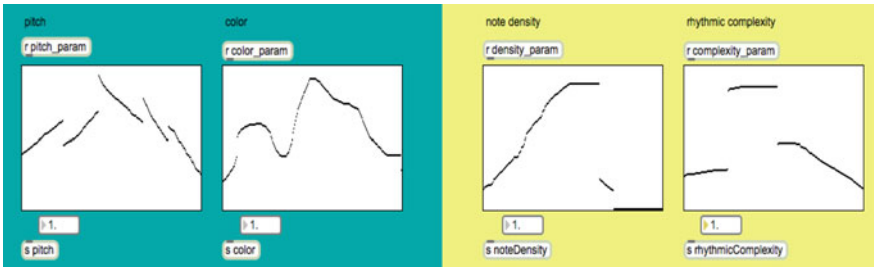


Fig. 4.17 GUI for drawing contours of pitch contour, harmonic color, note density, and rhythmic complexity

Pitch contour, harmonic color, note density, and rhythmic complexity are the core semantics driving the decisions during the path planning process. These were chosen based off of Johnson-Laird's description of how musicians improvise [21]. He argues that successful melodies can be generated by prioritizing individual pitches based on the constraints of the chord functions and contour with appropriate rhythmic figures.

Each of these semantics is represented as a time series. Each set of time series represents the duration of an individual phrase that is defined by the user in numbers of beats. The path planning generates the sequence that jointly describes all the semantics. A parameter weighing one semantic over another can be applied. A GUI (using MaxMSP) gives a user the ability to create observation sequences of each semantic and its weight (Fig. 4.17).

Each state in S is now defined by a configuration and a note to be played. For Shimon, the state space includes about 73,815 possible arm configurations. This state space is likely to vary considerably depending on the robotic platform. For platforms with enormous state spaces finding the global optimal path is not always feasible if the path is to be found in a timely manner. Therefore, a beam search is applied in order to prune unreasonable branches quickly. Additionally, the Viterbi algorithm lends itself to distributed computing techniques and can be computed across multiple CPUs if the state spaces are very large.

For each musical semantic, p_n (note density, rhythmic complexity, etc.) a time series of observations is provided (assume these time series are manually provided to the system). The emission score for state s_i at observation time t describes the aggregate score across all N parameters

$$b_t = \sum_{n=0}^{N} \lambda_n R(p_{n_t}, s_i) \tag{4.7}$$

where $R(p_n, s_i)$ describes the instantaneous score of the note in s_i given the semantic parameter and λ_n is applied to each parameter to describe its weight within the overall score. The transition score is described as

$$a_t = \sum_{n=0}^{N} \lambda_n R(p_{n_t}, p_{n_{t-1}}, s_t, s_{t-1}) \tag{4.8}$$

where $R(p_{n_t}, p_{n_{t-1}}, s_t, s_{t-1})$ is the score describing how well the transition between notes in the states s_t and s_{t-1} represent the transition between p_{n_t} and $p_{n_{t-1}}$. The optimal path is computed given these emission and transition scores for the semantic parameters.

4.5.3 Musical Results

The system is used to generate music under different conditions. The objective is to demonstrate that alternative musical decisions are made as a result of the physical constraints. Figure 4.18 shows musical outputs in which the semantic observation sequences are static, but the physical constraints are modified (see video for audio examples[1]). Thus, the differences among the outputs are a result of physical design and unique embodiment.

When applied to a real robotic platform the system performs adequately by generating note sequence the robot is capable of performing. In Fig. 4.19 an excerpt from a generated solo using the physical constraints of the Shimon robot is shown. Notice that during the faster sixteenth riffs the pitch intervals are generally larger than those of the eighth note sequence riffs. This is because it is physically impossible for Shimon to play such small intervals at the faster rate. Instead, the system generates the best sequence that approximates the musical features given Shimon's physical constraints. The higher note density motifs demonstrate arpeggio-like sequences and the pitch contours are a bit more exaggerated and extreme. Recall that Django Reinhardt made similar adjustments and adaptations addressing the loss of his two fingers.

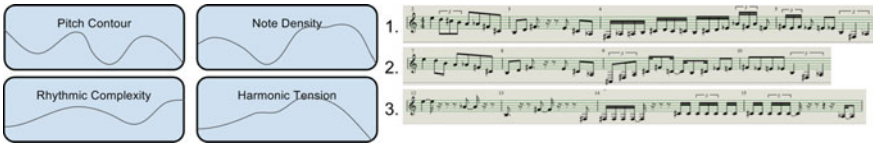


Fig. 4.18 Embodied generation samples. Three motifs are generated to satisfy the musical parameters on the left using different physical constraints. 1. A single arm robot that can move at a fast rate of one half step per millisecond. 2. Robot with four larger and slower arms (Shimon-like setting) that must avoid collision. 3. Very slow moving single arm robot with a fast strike rate of 20 Hz

[1] https://www.youtube.com/watch?v=S2Yda1ndKVc.

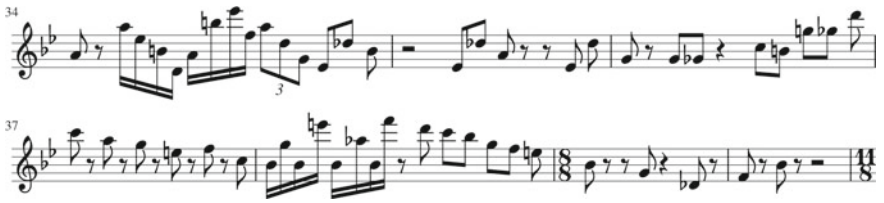


Fig. 4.19 An excerpt from a solo generated for the physical constraints the Shimon robot

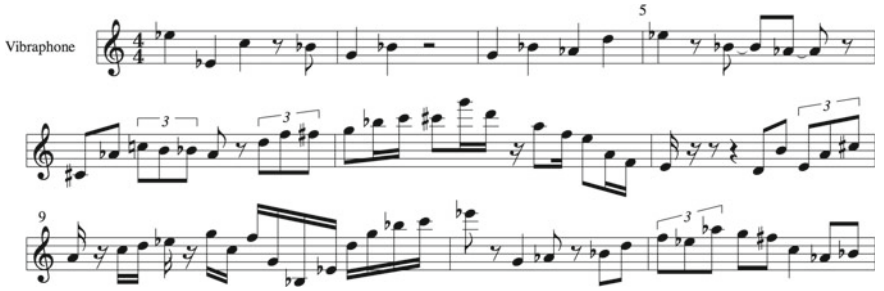


Fig. 4.20 An excerpt from a solo generated over the chord progression from the jazz standard "All the Things You Are" for a set of simulated physical constraints emulating a human vibraphone player.eps

In Fig. 4.20 an excerpt from a generated solo using the physical constraints designed to roughly emulate a human vibraphone player is shown. The physical constraints allow for two arms and the allowable be interval between successive notes of one arm is set to 150 ms, thus, using two arms the system can play rhythms with onset intervals greater than or equal to 75 ms.

While there are signs of stylistic biases and emergence resulting from the parameters used for the examples above, to demonstrate the effect of physical constraints on the musical output we can simulate an extremely capable robot. In Fig. 4.21[2] an example of a solo generated by a set of simulated physical constraints for a hypothetical robot is shown. The physical constraints allow for incredibly fast movement and access to a wide pitch range. The solo was generated for a tempo of 115 bpm and the resulting music produces motifs and riffs that are humanly impossible. This musical output would never have been generated if the system did not have some understanding of its physicality (unless the developer or composer explicitly encouraged these types of behaviors).

[2]This figure shows notes played simultaneously, but this is an artefact resulting from the score editor's inability to represent the necessary temporal resolution. The generated music is monophonic.

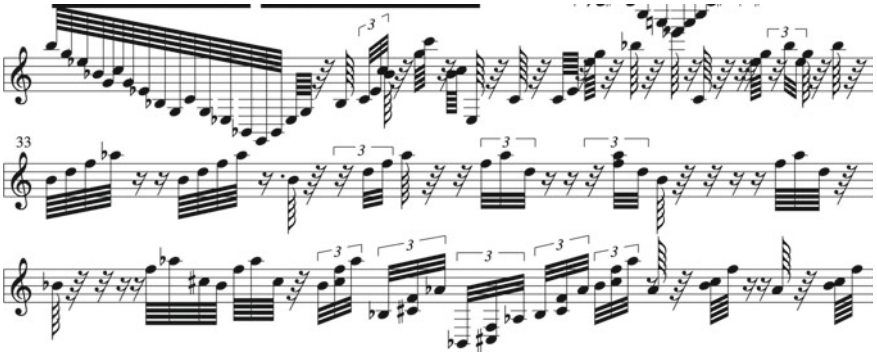


Fig. 4.21 An excerpt from a solo generated over the chord progression from the jazz standard "All the Things You Are" for a set of simulated physical constraints given a hypothetical robot a wide range of capabilities

4.5.4 Discussion

The jazz composition system presents a proof of concept that physical constraints can influence the musical decisions. While this is qualitatively true, the nature of this particular system poses some challenges.

Perhaps the most significant caveat of this system is that it requires planning. Having developed many real-time interactive music systems, we understand the appeal from a developer's perspective of greedy decision-making processes and systems that allow for notes to be generated in an on-line fashion. To take advantage of Viterbi and a planning process, multiple notes need to be processed at once. However, the use of planning makes sense on multiple levels. If all important musical features could be described by extremely localized characteristics without any regard to longer term structure, a simple n-gram model would suffice as the solution to music generation and perfect style modeling. This is not the case, however, and is demonstrated as such from how people talk and think about music. Norgaard explains the importance of planning as a key component of the thinking processes of artist-level human improvisers [28]. Not only does planning exist, but it is linked to musical expertise. Fidlon shows that more expert musicians were planning their musical strategies much further into the future compared to less capable musicians [29]. Recently, Norgaard demonstrated similar findings. Developing musicians tended to make note-level decisions, while the artist-level musicians planned out ideas that stretched over entire sections [30].

Additional elements of human improvisational thinking support the notion of planning over multiple notes as well. Often jazz musicians describe their decisions as stemming from an initial pool of musical ideas [28]. This "idea bank" consists of higher level musical semantics encapsulating sequences of notes. This is evidenced through the repeated use of specific licks and riffs by artist-level musicians and their modifications for different harmonic contexts.

These examples of human thinking suggest planning can be useful for generating optimal sequences from a purely musical perspective, but planning concepts have additional relevance to robotic musicianship applications compared to pure software applications. In software, sounds can be generated instantaneously, however, in the natural world some energy must be provided to the system to create sound acoustically. In robotic musicianship, the robot supplies the necessary kinetic energy to create a sound, thus, requiring some form of movement. If the robot really does have noticeable constraints, there will be inherent delays resulting from this movement. Most good solenoid-based striking or fast motor-based striking mechanisms do not have a noticeable delay between the time the computer sends the message to strike and the system strikes and produces sound. However, if additional motions are necessary, as is the case with Shimon, a noticeable delay may become present. Though it is possible for the delay to be very small, in practice these types of motions usually involve motors moving heavier components (like an entire limb) that can introduce delays of hundreds of milliseconds. Therefore, for real-time systems, some planning is already required in order for the robot to adhere to specific beat markers.

Another challenging characteristic of this system is the enormous state space. While Viterbi lends itself to a relatively straightforward distributed computing implementation, without significant hardware resources it is impossible to compute the paths in real-time. Alternative (and perhaps better) solutions to speeding up computation require heuristically driven pruning of the state space. It is likely that the states can be prioritized according to both musical and physical heuristics. This idea is addressed in the next section.

4.6 Neural Network Based Improvisation

This section builds off the concept of learning features using neural network methods described in Chap. 3 and applying those features to a generative system. The proposed method leverages unit selection and concatenation as a means of generating music using a procedure based on ranking, where a unit is considered to be a variable length number of measures of music. A generative model combining a deep structured semantic model (DSSM) with a long short term memory (LSTM) network to predict the next unit is described. This model is evaluated using objective metrics including mean rank and accuracy and with a subjective listening test in which expert musicians are asked to complete a forced-choiced ranking task. The model is compared to a note-level generative baseline that consists of a stacked LSTM trained to predict forward by one note. Finally, a method for incorporating the physical parameters of an embodied system is described. The embodied process uses a re-ranking strategy based on a metric defined by the Viterbi path planning process.

4.6.1 Introduction

In the previous section a generative system based on knowledge-based heuristics was described. While knowledge-based systems can produce compelling results, they are constrictive in that they can only represent what is codified by the developer. Though an expert musician should be able to build an expert system, there are important features in music that are likely to go unaddressed. Music is an extremely high dimensional domain and capturing all relevant features using a combination of rules and various hand-designed heuristics is not possible. Therefore, an alternative methodology to creating the heuristics is desirable.

For the last half century researchers and artists have developed many types of algorithmic composition systems. Many of these efforts are driven by the allure of both simulating human aesthetic creativity through computation and tapping into the artistic potential deep-seated in the inhuman characteristics of computers. Some systems may employ rule-based, sampling, or morphing methodologies to create music [31]. In this section, we present a method that falls into the class of symbolic generative music systems consisting of data driven models which utilize statistical machine learning.

Within this class of music systems, the most prevalent method is to create a model that learns likely transitions between notes using sequential modeling techniques such as Markov chains or recurrent neural networks [32, 33]. The learning minimizes note-level perplexity and during generation the models may stochastically or deterministically select the next best note given the preceding note(s). However, there is significant evidence that musical decisions are made using collections of predetermined note groupings [34], in other words, instead of making a decision about one note at a time, humans make decisions about groups of notes are made and inserted into the performance. In a recent study, Norgaard analyzed a collection of Charlie Parker solos and concluded, "the sheer ubiquity of patterns and the pairing of pitch and rhythm patterns support the theory that preformed structures are inserted during improvisation. The patterns may be encoded both during deliberate practice and through incidental learning processes" [35]. However, from his qualitative investigation based on artist-level interviews he found that improvisation is likely a combination of inserting well-learned ideas from memory and tweaking those ideas to fit the specific harmonic context. Here, a system that uses a similar combination approach is presented and a method to generate monophonic melodic lines based on unit selection is outlined. In this work a unit is a precomposed section of music with a fixed duration such as one or two measures.

This first part of this section focuses on a method for applying what is learned using a disembodied generative music approach. The second part of the section addresses how this method can be augmented to address the physical constraints of a robotic system such that units are chosen according to music as well as physical related metrics. The two immediate research objectives include:

1. **Develop a method for selecting and concatenating units**—For generating sequences it is necessary to be able to do more than just numerically describe a

unit. Adjacent units must be semantically similar and seamlessly connect together based on a combination of musical priors and the physical constraints of the performer.

2. **Modify pitches in units by jointly optimizing for harmonic context and physical constraints**—For a robotic musician to use unit selection it must be able to play the units. Therefore, path planning is used as a final step in the process. The units themselves serve as a sort of heuristic that can help to bias the types of decisions the path planning will make, thus, making it possible to prune the state space and make decisions in a timely manner.

The architecture of this work is inspired by a technique that is commonly used in text-to-speech (TTS) systems. The two system design trends found in TTS are statistical parametric and unit selection [36]. In the former, speech is completely reconstructed given a set of parameters. The premise for the latter is that new, intelligible, and natural sounding speech can be synthesized by concatenating smaller audio units that were derived from a preexisting speech signal [37–39]. Unlike a parametric system, which reconstructs the signal from the bottom up, the information within a unit is preserved and is directly applied for signal construction. When this approach is applied to music, the generative system can similarly get some of the structure inherent to music "for free" by pulling from a unit library.

The ability to directly use the music that was previously composed or performed by a human can be a significant advantage when trying to imitate a style or pass a musical Turing test [40]. However, there are also drawbacks to unit selection that the more common note-to-note level generation methods do not need to address. The most obvious drawback is that the output of a unit selection method is restricted to what is available in the unit library. Note-level generation provides maximum flexibility in what can be produced. Ideally, the units in a unit selection method should be small enough such that it is possible to produce a wide spectrum of music, while remaining large enough to take advantage of the built-in information.

Another challenge with unit selection is that the concatenation process may lead to "jumps" or "shifts" in the musical content or style that may sound unnatural and jarring to a listener. Even if the selection process accounts for this, the size of the library must be sufficiently large in order to address many scenarios. Thus, the process of selecting units can equate to a massive number of comparisons among units when the library is very big. Even after pruning the computational demands can be high. However, the method can be effective as long as the computing power is available and unit evaluation can be performed in parallel processes. Additionally, methods such as vector quantization can be applied to reduce the number of comparisons in the nearest neighbor search.

The proposed generation system ranks individual units based on two values: (1) a semantic relevance score between two units and (2) a concatenation cost that describes the distortion at the seams where units connect. The semantic relevance score is determined by using a deep structured semantic model (DSSM) to compute the distance between two units in a compressed embedding space [41]. The concatenation cost is derived by first learning the likelihood of a sequence of musical events

(such as individual notes) with an LSTM and then using this LSTM to evaluate the likelihood of two consecutive units. The model's ability to select the next best unit is evaluated based on ranking accuracy and mean rank. To measure the subjective nature of music and evaluate whether the networks have learned to project music into a meaningful space a listening study is performed. The study evaluates the "naturalness" and "likeability" of the musical output produced by versions of the system using units of lengths four, two, and one measures. Additionally, these unit selection based systems are compared to the more common note-level generative models. As a baseline an LSTM trained to predict forward by one note is used.

4.6.2 Semantic Relevance

In both TTS and the previous musical reconstruction tests a target is provided. For generation tasks, however, the system must predict the next target based on the current sequential and contextual information that is available. In music, even if the content between two contiguous measures or phrases is different, there exist characteristics that suggest the two are not only related, but also likely to be adjacent to one another within the overall context of a musical score. We refer to this likelihood as the "semantic relevance" between two units.

This measure is obtained from a feature space learned using a DSSM. Though the underlying premise of the DSSM is similar to the autencoder in that the objective is to learn good features in a compressed semantic space, the DSSM features, however, are derived in order to describe the relevance between two different units by specifically maximizing the posterior probability of consecutive units, $P(u_n|u_{n-1})$, found in the training data. This idea stems from word in embeddings in which a word learns the context in which it would be used. The DSSM model and others such as the skip-gram have demonstrated to learn effective embeddings using this method [42, 43]. Recently, efficacy has been demonstrated in music using the skip-gram model on chord progressions. In this context, the embeddings learn a pitch space similar to that of the circle of fifths [44].

In this work, a space representing both rhythm and pitch features is learned. The same BOW features described in the previous section are used as input to the model. There are two hidden layers and the output layer describes the semantic feature vector used for computing the relevance. Each layer has 128 rectified linear units. The same softmax that was used for the autoencoder for computing loss is used for the DSSM. However, the loss is computed within vectors of the embedding space such that

$$-log \prod_{(u_{n-1},u_n)} P(\mathbf{u_n}|\mathbf{u_{n-1}}) \tag{4.9}$$

where the vectors, $\mathbf{u_n}$ and $\mathbf{u_{n-1}}$, represent the 128 length embeddings of each unit derived from the parameters of the DSSM. Once the parameters are learned through

gradient descent the model can be used to measure the relevance between any two units, U_1 and U_2, using cosine similarity $sim(\mathbf{U_1}, \mathbf{U_2})$ (see Eq. 4.1).

The DSSM provides a meaningful measure between two units, however, it does not describe how to join the units (which one should come first). Similarly, the BOW representation of the input vector does not contain information that is relevant for making decisions regarding sequence. In order to optimally join two units a second measure is necessary to describe the quality of the join.

4.6.3 Concatenation Cost

By using a unit library made up of original human compositions or improvisations, we can assume that the information within each unit is musically valid. In an attempt to ensure that the music remains valid after combining new units we employ a concatenation cost to describe the quality of the join between two units. This cost requires sequential information at a more fine grained level than the BOW-DSSM can provide.

The general premise for computing the quality of the concatenation is shown in Fig. 4.22. A multi-layer LSTM is used to learn a note-to-note level model. This is akin to a character level language model. Each state in the model represents an individual note that is defined by its pitch and duration. This constitutes about a 3,000 note vocabulary. Using a one-hot encoding for the input, the model is trained to predict the next note, y_T, given a sequence, $\mathbf{x} = (x_1, ..., x_T)$, of previously seen notes. During training, the output sequence, $\mathbf{y} = (y_1, ..., y_T)$, of the network is such that $y_t = x_{t+1}$. Therefore, the predictive distribution of possible next notes, $Pr(x_{T+1}|\mathbf{x})$, is represented in the output vector, y_T. We use a sequence length of $T = 36$.

The aim of the concatenation cost is to compute a score evaluating the transition between the last note of the unit, u_{n-1,x_T}, and the first note of the unit, u_{n,y_T}. By using an LSTM it is possible to include additional context and note dependencies that exist further in the past than u_{n-1,x_T}. The cost between two units is computed as

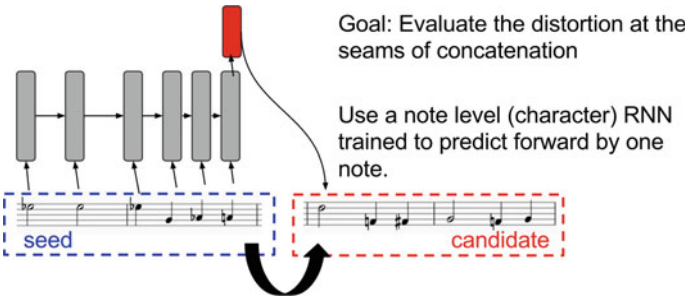


Fig. 4.22 The concatenation cost is computed by evaluating the sequence of notes where two units join

$$C(u_{n-1}, u_n) = -\frac{1}{J}\sum_{j}^{J} log\, Pr(x_j|\mathbf{x_j}) \tag{4.10}$$

where J is the number of notes in u_n, x_j is the jth note of u_n, and $\mathbf{x_j}$ is the sequence of notes (with length T) immediately before x_j. Thus, for $j > 1$ and $j < T$, $\mathbf{x_j}$ will include notes from u_n and u_{n-1} and for $j \geq T$, $\mathbf{x_j}$ will consist of notes entirely from u_n. In practice, however, the DSSM performs better than the note-level LSTM for predicting the next unit and we found that computing C with $J = 1$ provides the best performance. Therefore, the quality of the join is determined using only the first note of the unit in question (u_n).

The sequence length, $T = 36$, was chosen because it is roughly the average number of notes in four measures of music (from our dataset). Unlike the DSSM, which computes distances based on information from a fixed number of measures, the context provided to the LSTM is fixed in the number of notes. This means it may look more or less than four measures into the past. In the scenario in which there is less than 36 notes of available context the sequence is zero padded.

4.6.4 Ranking Units

A ranking process that combines the semantic relevance and concatenation cost is used to perform unit selection. Often times in music generation systems the music is not generated deterministically, but instead uses a stochastic process and samples from a distribution that is provided by the model. One reason for this is that note-level Markov chains or LSTMs may get "stuck" repeating the same note(s). Adding randomness to the procedure helps to prevent this. Here, we describe a deterministic method as this system is not as prone to repetitive behaviors. However, it is simple to apply stochastic decision processes to this system as the variance provided by sampling can be desirable if the goal is to obtain many different musical outputs from a single input seed.

The ranking process is performed in four steps:

1. Rank all units according to their semantic relevance with an input seed using the feature space learned by the DSSM.
2. Take the units whose semantic relevance ranks them in the top 5% and re-rank based on their concatenation cost with the input.
3. Re-rank the same top 5% based on their combined semantic relevance and concatenation ranks.
4. Select the unit with the highest combined rank.

By limiting the combined rank score to using only the top 5% we are creating a bias towards the semantic relevance. The decision to do this was motivated by findings from pilot listening tests in which it was found that a coherent melodic sequence relies more on the stylistic or semantic relatedness between two units than a smooth transition at the point of connection.

Table 4.5 Unit ranking

Model	Unit length (measures)	Acc (%)	Mean rank@50 + standard dev.
LSTM	4	17.2	14.1 ± 5.6
DSSM	4	33.2	6.9 ± 3.8
DSSM+LSTM	4	36.5	5.9 ± 2.7
LSTM	2	16.6	14.8 ± 5.8
DSSM	2	24.4	10.3 ± 4.3
DSSM+LSTM	2	28.0	9.1 ± 3.8
LSTM	1	16.1	15.7 ± 6.6
DSSM	1	19.7	16.3 ± 6.6
DSSM+LSTM	1	20.6	13.9 ± 4.1

4.6.5 Evaluating the Model

The model's ability to choose musically appropriate units can be evaluated using a ranking test. The task for the model is to predict the next unit given a never before seen four measures of music (from the held out test set). The prediction is made by ranking 50 candidates in which one is the truth and the other 49 are units randomly selected from the database. We repeat the experiments for musical units of different lengths including four, two, and one measures. The results are reported in Table 4.5 and they are based on the concatenation cost alone (LSTM), semantic relevance (DSSM), and the combined concatenation and semantic relevance using the selection process described above (DSSM+LSTM). The accuracy depicts the rate at which the truth is ranked the best.

4.6.6 Discussion

The primary benefit of unit selection is being able to directly apply previously composed music. The challenge is stitching together units such that the musical results are stylistically appropriate and coherent. Another challenge in building unit selection systems is determining the optimal length of the unit. The goal is to use what has been seen before, yet have flexibility in what the system is capable of generating. The results of the ranking task may indicate that units of four measures have the best performance, yet these results do not provide any information describing the quality of the generated music.

4.6.7 Subjective Evaluation

The mean rank metric of the last task provides a rough estimate that the model has learned something of importance. However, to be certain that the space is perceptually and musically meaningful, a user study is necessary. In this section a subjective listening test is described. Participants included 32 music experts in which a music expert is defined as an individual that has or is pursuing a higher level degree in music, a professional musician, or a music educator. Four systems were evaluated. Three of the systems employed unit selection using the DSSM+LSTM approach with unit lengths of four, two, and one measures. The fourth system used the note-level LSTM to generate each note at a time.

The design of the test was inspired by subjective evaluations used by the TTS community. To create a sample each of the four systems was provided with the same input seed (retrieved from the held out dataset) and from this seed each system then generated four additional measures of music. This process results in four eight-measure music sequences with the same first four measures. The process was repeated 60 times using random four measure input seeds.

In TTS evaluations participants are asked to rate the quality of the synthesis based on naturalness and intelligibility [45]. In music performance systems the quality is typically evaluated using naturalness and likeability [46]. For a given listening sample, a participant is asked to listen to four eight-measure sequences (one for each system) and then are asked to rank the candidates within the sample according to questions pertaining to:

1. Naturalness of the transition between the first and second four measures.
2. Stylistic relatedness of the first and second four measures.
3. Naturalness of the last four measures.
4. Likeability of the last four measures.
5. Likeability of the entire eight measures.

Each participant was asked to evaluate 10 samples that were randomly selected from the original 60, thus, all participants listened to music generated by the same four systems, but the actual musical content and order randomly differed from participant to participant. The tests were completed online with an average duration of roughly 80 min.

4.6.8 Results

Rank order tests provide ordinal data that emphasize the relative differences among the systems. The average rank was computed across all participants similarly to TTS-MOS tests. The percent of being top ranked was also computed. These are shown in Figs. 4.23 and 4.24.

In order to test significance the non-parametric Friedman test for repeated measurements was used. The test evaluates the consistency of measurements (ranks)

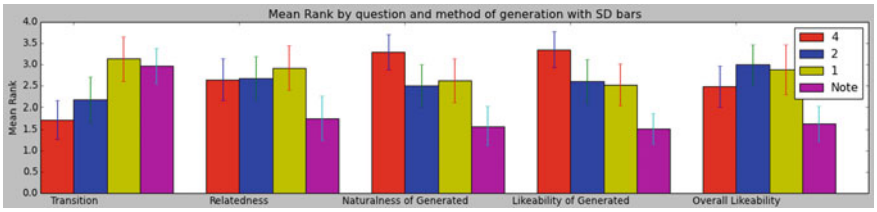


Fig. 4.23 The mean rank and standard deviation for the different music generation systems using units of lengths 4, 2, and 1 measures and note level generation. A higher mean rank indicates a higher preference (i.e. higher is better)

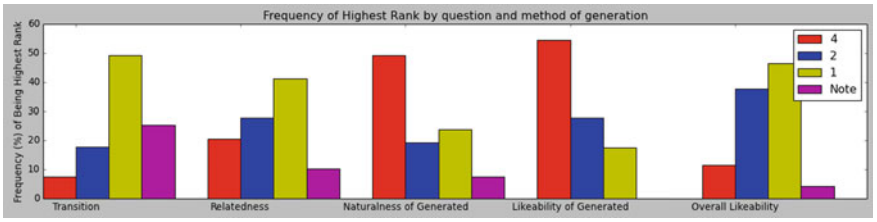


Fig. 4.24 The frequency of being top ranked for the different music generation systems using units of lengths 4, 2, and 1 measures and note level generation. In both Fig. 5 and 6 results are reported for each of the five hypotheses: (1) **Transition**—the naturalness of the transition between the first four measures (input seed) and last four measures (computer generated), (2) **Relatedness**—the stylistic or semantic relatedness between the first four measures and last four measures, (3) **Naturalness of Generated**—the naturalness of the last four measures only, (4) **Likeability of Generated**—the likeability of the last four measures only, and (5) **Overall Likeability**—the overall likeability of the entire eight measure sequence

Table 4.6 Subjective ranking

Variable	Best –> Worst
H1—Transition naturalness	1, N, 2, 4
H2—Semantic relatedness	1, 2, 4, N
H3—Naturalness of generated	4, 1, 2, N
H4—Likeability of generated	4, 2, 1, N
H5—Overall likeability	2, 1, 4, N

obtained in different ways (audio samples with varying input seeds). The null hypothesis states that random sampling would result in sums of the ranks for each music system similar to what is observed in the experiment. A bonferonni post-hoc correction was used to correct the p-value for the five hypotheses (derived from the itemized question list described earlier).

For each hypothesis the Friedman test resulted in $p < 0.05$, thus, rejecting the null hypothesis. The sorted ranks for each of the generation system is described in Table 4.6.

The results indicate that there exists an optimal unit length that is greater than a single note and less than four measures. This ideal unit length appears to be one or two measures with a bias seemingly favoring one measure.

4.6.9 An Embodied Unit Selection Process

So far this section has described a successful method of learning musical semantics and a method for generating music using unit selection. The selection process incorporates a score based on the semantic relevance between two units and a score based on the quality of the join at the point of concatenation. Two variables essential to the quality of the system are the breadth and size of the unit database and the unit length. An autoencoder was used to demonstrate the ability to reconstruct never before seen music by picking units out of a database. In the situation that an exact unit is not available the nearest neighbor computed within the embedded vector space is chosen. A subjective listening test was performed in order to evaluate the generated music using different unit durations. Music generated using units of one or two measure durations tended to be ranked higher according to naturalness and likeability than units of four measures or note-level generation.

Applying this generation system in its current form in the context of robotic musicianship would result in the same processing pipeline of *music generation* $\rightarrow$ *path planning* $\rightarrow$ *musical output* that this thesis is looking to avoid. Additionally, this system does not address situations in which the melodies should conform to a provided harmonic context (chord progression), therefore, it is not yet suitable for harmony-based jazz improvisation. This section addresses both of these issues.

In order to include traits of embodiment into the musicianship portion of this system the unit selection process should integrate variables describing the physical constraints. Currently, selections are made using two attributes describing the musical quality of joining two units (semantic relevance and concatenation cost). Though this information remains highly relevant, by itself it is not suitable for an embodied musical processing system that makes decisions by jointly optimizing for musical heuristics and physicality.

An additional cost, the *embodiment cost*, is computed describing the physical capability of playing a single unit. Using the path planning method described in the previous measure it is possible to determine if a robot can play a specific unit given its physical constraints and current state in the C-Space. This means that the robot's most recent configuration is vital for evaluating potential next units. In this case, the observation sequence in the Viterbi process comes from the notes sequence of a unit. The embodied cost is measured by performing Viterbi with this observation sequence and computing the most efficient movement sequence. If N represents the number of notes in the observation sequence O and T represents the number of notes that are performed given the Viterbi computed path for O then the embodiment cost, E, is the fraction of notes in the unit that can be played:

$$E = T/N \tag{4.11}$$

The goal is to pick a unit that the robotic system is fully capable of performing. Therefore, in this implementation all units with $E < 1.0$ are removed from consideration. The final unit selection procedure includes semantic relevance, concatenation cost, and embodiment cost. Using these metrics the robot creates musical opportunities based off of its current physical configuration and the learned musical heuristics, hence, satisfying Vijay Iyer's declaration that a good musician "requires an awareness of the palette of musical acts available in general, and particularly of the dynamically evolving subset of this palette that is physically possible at any given moment" [47].

Figure 4.25 shows three generated sequences of music using the unit selection process. The first measure in each four measure sequence is the same and used as the seed to generate the next three measures. Examples are shown using unit selection with and without an embodiment measure.

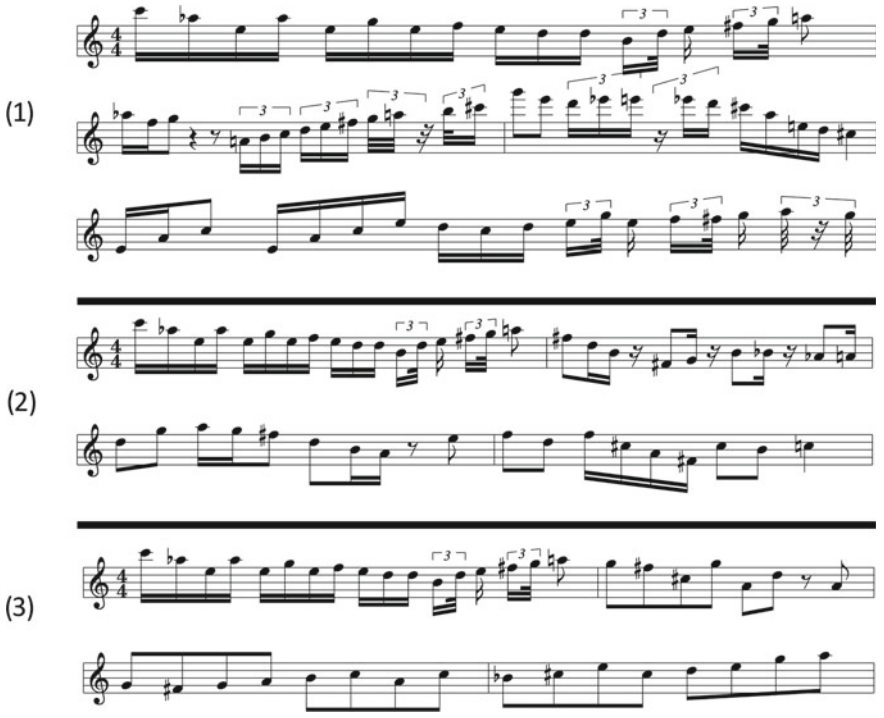


Fig. 4.25 Three measures of music are generated using the unit selection process. The first measure in each sequence serves as the seed. Units are chosen using three different methodologies: (1) The units are selected using the semantic relevance and concatenation cost; (2) The units are selected using semantic relevance, concatenation cost, and an embodiment cost computed for the physical constraints of the Shimon robot; (3) The units are selected using semantic relevance, concatenation cost, and an embodiment cost based on a robot similar to Shimon, but with more significant speed limitations

Unfortunately, an adequate dataset with labeled chord progressions and improvisations does not exist. Therefore, in order to use this system with a given chord progression a "note tweaking" step is applied. The process is fairly straightforward:

1. **Rank units according to musical quality**—The units are initially ranked according to the semantic relevance and concatenation cost.
2. **Re-rank the top n units according to physicality and chord changes**—After units are initially sorted the top n units are selected for further evaluation based on the path planning algorithm.

Item two in this process has two objectives: (1) tweak pitches in the unit to fit a specific harmonic context and (2) find a movement path that jointly addresses the physical constraints of the robot. These objectives are integrated into a single task such that the result jointly optimizes across both objectives. The path planning process is similar to the Viterbi planning of precomposed melodies (as opposed to the knowledge-based generative system), however, the pitches can be modified slightly. Therefore, the unit serves as a type of heuristic to help prune away possible branches in the search. In this implementation, the bias is fairly extreme and only allows the pitches of a unit to be modified by ± 2 half steps. The pitches are modified according to their tonal distance from the observed chord (using the metric described in the previous section) and the robotic movement variables. The general pitch contour is kept and rhythm is maintained exactly.

$$\hat{U} = \arg\max_{U} PATH(x, U) \tag{4.12}$$

where x is the last state (in the robot and music c-space) of the current unit, Y is all the notes of a single unit chosen from the initially top n ranked units, and *PATH* is a function denoting the resulting score of the path generated from the Viterbi decoding process. The process is outlined in Fig. 4.26.

Ideally, units wouldn't be ranked and re-ranked in two separate processes. In practice, however, the Viterbi process is a computationally expensive metric, particularly if the state space is massive. An additional preprocessing step can be performed to reduce this expense. For each unit in the library, path planning can be performed. All units that are incapable of being performed in their current form (without pitch tweaking) can be removed from the database.

4.7 Conclusion

In this chapter we discussed approaches to autonomous composition for robotic musicians. While many of the same techniques can be used for software-based or non-robotic musicians, for robots we emphasize physical embodiment. We consider how embodiment influences the generative processes through concepts such as path planning and also how to leverage mechanical abilities that are humanly impossible.

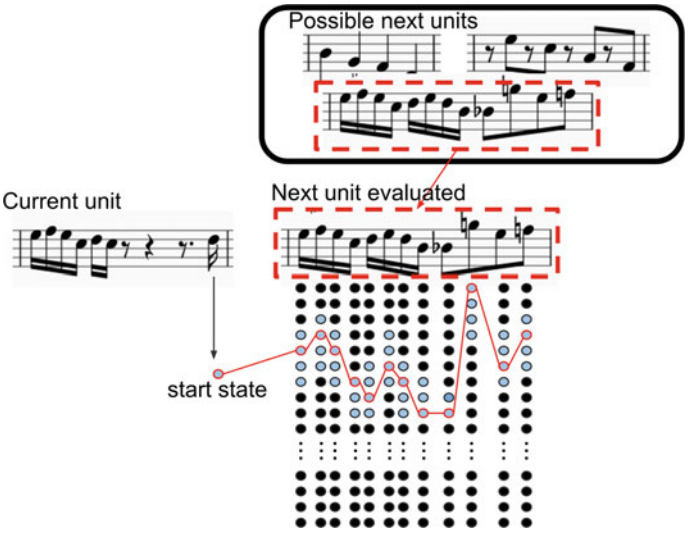


Fig. 4.26 Once a unit is selected path planning is performed. The objective is to find a path that maintains the general contour and rhythm qualities of the unit, while finding a sequence that modifies the pitches such that they support the particular tonal center and chord function. Therefore, the search space is pruned to include only pitches close to each pitch in the unit. The generated sequence also addresses the physical constraints of the robot so the resulting score of the path describes both the unit's ability to appropriately fit the chord progression and robot's ability to play the notes. Each possible unit is evaluated according to this metric and the unit with the best score is chosen

By understanding music in a manner that correlates with human perception and simultaneously "playing like a machine" we hope to create generative music systems that expand musical culture and encourage novel interactive experiences.

References

1. Biles, John. 1994. Genjam: A genetic algorithm for generating jazz solos. In *Proceedings of the international computer music conference*, pp. 131–131. International Computer Music Association.
2. Moroni, Artemis, Jônatas Manzolli, Fernando Von Zuben, and Ricardo Gudwin. 2000. Vox populi: An interactive evolutionary system for algorithmic music composition. *Leonardo Music Journal* 49–54.
3. Tokui, Nao, Hitoshi Iba, et al. 2000. Music composition with interactive evolutionary computation. In *Proceedings of the third international conference on generative art*, vol. 17, pp. 215–226.
4. Brown, Chris. 1999. Talking drum: A local area network music installation. *Leonardo Music Journal* 23–28.
5. Smith, Lloyd A., Rodger J. McNab, and Ian H. Witten. 1998. Sequence-based melodic comparison: A dynamic programming approach. *Computing in Musicology: A Directory of Research* (11): 101–118.

6. Lerdahl, Fred, and Ray S. Jackendoff. 1996. *A generative theory of tonal music*. MIT press.
7. Pachet, Francois. 2003. The continuator: Musical interaction with style. *Journal of New Music Research* 32 (3): 333–341.
8. Singer, Eric, Jeff Feddersen, Chad Redmon, and Bil Bowen. 2004. Lemur's musical robots. In *Proceedings of the 2004 conference on new interfaces for musical expression*, 181–184. National University of Singapore.
9. Solis, Jorge, Atsuo Takanishi, and Kunimatsu Hashimoto. 2010. Development of an anthropomorphic saxophone-playing robot. In *Brain, body and machine*, 175–186. Springer.
10. Gil, Weinberg, Beck Andrew, Godfrey Mark. 2009. Zoozbeat: A gesture-based mobile music studio
11. Krumhansl, Carol L. 2001. *Cognitive foundations of musical pitch*, vol. 17. Oxford University Press.
12. Newell, Allen, J.C. Shaw, and Herbert A. Simon. 1988. Chess-playing programs and the problem of complexity. In *Computer games I*, 89–115. Springer.
13. Sariff, N., and Norlida Buniyamin. 2006. An overview of autonomous mobile robot path planning algorithms. In *SCOReD 2006. 4th student conference on research and development, 2006*, 183–188. IEEE.
14. Nikolaidis, Ryan, and Gil Weinberg. 2010. Playing with the masters: A model for improvisatory musical interaction between robots and humans. In *2010 IEEE RO-MAN*, 712–717. IEEE.
15. Gopinath, Deepak. 2015. Enhancing stroke generation and expressivity in robotic drummers—A generative physics model approach. Master's Thesis, Georgia Institute of Technology.
16. Dubnov, Shlomo, Gerard Assayag, Olivier Lartillot, and Gill Bejerano. 2003. Using machine-learning methods for musical style modeling. *Computer* 36 (10): 73–80.
17. Franklin, Judy A. 2001. Multi-phase learning for jazz improvisation and interaction. In *Proceedings of the eighth biennial symposium for arts & technology*.
18. Keller, Robert M., and David R. Morrison. 2007. A grammatical approach to automatic improvisation. In *Proceedings, fourth sound and music conference, Lefkada, Greece, July. Most of the soloists at Birdland had to wait for Parkers next record in order to find out what to play next. What will they do now*.
19. Levine, Mark. 2011. *The jazz piano book*. O'Reilly Media, Inc.
20. Crook, Hal. 1991. *How to improvise. Advance music*.
21. Johnson-Laird, Philip N. 2002. How jazz musicians improvise. *Music Perception: An Interdisciplinary Journal* 19 (3): 415–442.
22. Stone, George Lawrence. 2013. *Stick control: For the snare drummer*. Alfred Music.
23. Shmulevich, Ilya, and Dirk-Jan Povel. Complexity measures of musical rhythms.
24. Lisheng, Xu, David Zhang, Kuanquan Wang, and Lu Wang. 2006. Arrhythmic pulses detection using lempel-ziv complexity analysis. *EURASIP Journal on Advances in Signal Processing* 2006 (1): 1–12.
25. Shmulevich, Ilya, and D.-J. Povel. 1998. Rhythm complexity measures for music pattern recognition. In *1998 IEEE second workshop on multimedia signal processing*, 167–172. IEEE.
26. Shmulevich, Ilya, and D.-J. Povel. 2000. Measures of temporal pattern complexity. *Journal of New Music Research* 29 (1): 61–69.
27. Toussaint, Godfried T., et al. 2002. A mathematical analysis of African, Brazilian, and Cuban clave rhythms. In *Proceedings of BRIDGES: Mathematical connections in art, music and science*, 157–168. Citeseer.
28. Norgaard, Martin. 2011. Descriptions of improvisational thinking by artist-level jazz musicians. *Journal of Research in Music Education* 59 (2): 109–127.
29. Fidlon, James Daniel. 2011. Cognitive dimensions of instrumental jazz improvisation. PhD thesis.
30. Norgaard, Martin. 2016. Descriptions of improvisational thinking by developing jazz improvisers. *International Journal of Music Education* 0255761416659512,.
31. Papadopoulos, George, and Geraint Wiggins. 1999. AI methods for algorithmic composition: A survey, a critical view and future prospects. In *AISB symposium on musical creativity*, 110–117. Edinburgh, UK.

32. Pachet, François, and Pierre Roy. 2011. Markov constraints: Steerable generation of Markov sequences. *Constraints* 16 (2): 148–172.
33. Franklin, Judy A. 2006. Recurrent neural networks for music computation. *INFORMS Journal on Computing* 18 (3): 321–338.
34. Pressing, Jeff. 1988. Improvisation: Methods and models. *Generative processes in music*, Hg. John A. Sloboda, 129–178. Oxford.
35. Norgaard, Martin. 2014. How jazz musicians improvise. *Music Perception: An Interdisciplinary Journal* 31 (3): 271–287.
36. Zen, Heiga, Keiichi Tokuda, and Alan W. Black. 2009. Statistical parametric speech synthesis. *Speech Communication* 51 (11): 1039–1064.
37. Hunt, Andrew J., and Alan W. Black. 1996. Unit selection in a concatenative speech synthesis system using a large speech database. In *ICASSP-96. Conference proceedings, 1996 IEEE international conference on acoustics, speech, and signal processing, 1996*, vol. 1, pp. 373–376. IEEE.
38. Black, Alan W., and Paul A. Taylor. 1997. Automatically clustering similar units for unit selection in speech synthesis.
39. Conkie, Alistair, Mark C. Beutnagel, Ann K. Syrdal, and Philip E. Brown. 2000. Preselection of candidate units in a unit selection-based text-to-speech synthesis system. In *Proceedings of ICSLP*, Beijing.
40. Cope, David, and Melanie J. Mayer. 1996. *Experiments in musical intelligence*, vol. 12. AR editions Madison, WI.
41. Huang, Po-Sen, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, 2333–2338. ACM.
42. Guthrie, David, Ben Allison, Wei Liu, Louise Guthrie, and Yorick Wilks. 2006. A closer look at skip-gram modelling. In *Proceedings of the 5th international Conference on Language Resources and Evaluation (LREC-2006)*, 1–4.
43. Levy, Omer, and Yoav Goldberg. 2014. Dependency-based word embeddings. In *ACL*, vol. 2, 302–308. Citeseer.
44. Huang, Cheng-Zhi Anna, David Duvenaud, and Krzysztof Z. Gajos. 2016. Chordripple: Recommending chords to help novice composers go beyond the ordinary. In *Proceedings of the 21st international conference on intelligent user interfaces*, 241–250. ACM.
45. Stevens, Catherine, Nicole Lees, Julie Vonwiller, and Denis Burnham. 2005. On-line experimental methods to evaluate text-to-speech (tts) synthesis: Effects of voice gender and signal quality on intelligibility, naturalness and preference. *Computer Speech & Language* 19 (2): 129–146.
46. Katayose, Haruhiro, Mitsuyo Hashida, Giovanni De Poli, and Keiji Hirata. 2012. On evaluating systems for generating expressive music performance: The rencon experience. *Journal of New Music Research* 41 (4): 299–310.
47. Iyer, Vijay. 2002. Embodied mind, situated cognition, and expressive microtiming in African-American music. *Music Perception: An Interdisciplinary Journal* 19 (3): 387–414.

Chapter 5
"Be Social"—Embodied Human-Robot Musical Interactions

5.1 Abstract

Embodiment has a significant effect on social human-robot interaction, from enabling fluent turn-taking between humans and robots [1] to humans' positive perception of robotic conversants [2]. In Robotic Musicianship, embodiment and gestural musical interaction can provide social benefits that are not available with standard computer based interactive music [3, 4]. The physical presence of a robot can help facilitate musical leader-follower relationships; music-making gestures can support synchronization by allowing humans to anticipate robotic actions; and ancillary non-music making gestures can convey emotions that affect a human's musical experience. Robotic musicians' embodied social presence could also inspire human musicians to be more engaged in the joint activity and enhance the aesthetic outcome to audience. We start this chapter with a description of embodied social interaction, turn taking, and pattern learning with Haile. We then discuss the effect of Shimon's music making and ancillary gestures on anticipation and audience engagement, Shimi's emotion conveyance through gestures, and end with Shimi's interaction scenarios outside of music performance.

5.2 Embodied Interaction with Haile

We developed several embodied interaction schemes for Haile in an effort to allow the robot to listen to and play along with live accompanying drummers. One of our main goals was to create inspiring human-machine interactions based on our theory of interdependent group interaction in interconnected musical networks [5, 6]. At the core of this theory is a categorization of collaborative musical interactions in networks of artificial and live players based on sequential and synchronous operations with both centralized and decentralized control schemes. In sequential decentralized interactions (Fig. 5.1) , players create their musical output with no

G. Weinberg et al., *Robotic Musicianship*, Automation, Collaboration,
& E-Services 8, https://doi.org/10.1007/978-3-030-38930-7_5

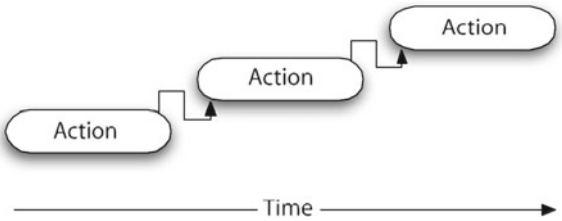


Fig. 5.1 Model of sequential decentralized interaction. Musical actions are taken in succession without synchronous input from other participants and with no central system to coordinate the interaction

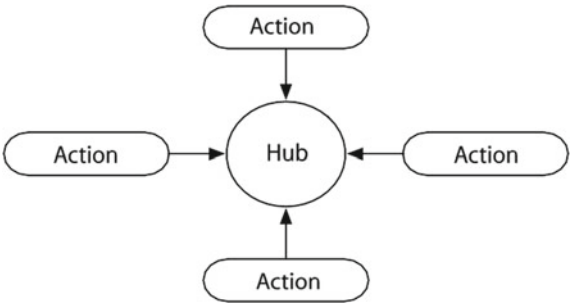


Fig. 5.2 Model of synchronous centralized interaction. Human and machine players take musical actions simultaneously and interact through a computerized hub that interprets and analyzes the input data

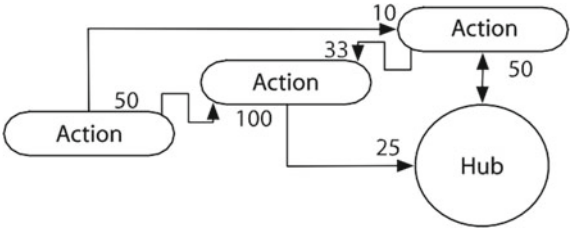


Fig. 5.3 A combination of centralized, decentralized, synchronous, and sequential musical actions in an asymmetric topology with weighted gates of influence

direct influence from a central system or other players. They can then interact with algorithmically generated responses by listening, modifying and sending their new musical materials to other participants in a sequential manner. In comparison, synchronous centralized networks (Fig. 5.2) allow participants, both human and artificial, to manipulate their peers' music in real-time. To facilitate such interactions, the network requires a hub that performs real-time analysis and generative functions. More sophisticated schemes of interaction can be designed by combining centralized, decentralized, synchronous, and sequential interactions in different architectures and trajectories, and by embedding weighted levels of influence between participants (e.g., Fig. 5.3).

5.2.1 Interaction Modes

Informed by these ideas, we developed six different interaction modes for Haile: *Imitation*, *Stochastic Transformation*, *Perceptual Transformation*, *Beat Detection*,

Simple Accompaniment, and *Perceptual Accompaniment*. While Haile is not capable of ancillary gestures aimed at facilitating social cues, we hypothesize that his physical embodiment and the music generating gestures in both arms can help facilitating visual cues that enables turn taking and social interactions, not possible with traditional graphical user interfaces.

In the first mode, *Imitation*, Haile listens to a human musicians, detects onsets and pitches and repeats the sequence as a response. Haile waits for a predefined length of silence after a human plays a rhythm before he imitates it in a sequential call-and-response manner. Haile uses one of his arms to play lower pitches close to the drumhead center and the other arm to play higher pitches close to the rim. In the second mode, *Stochastic Transformation*, Haile improvises in a call-and-response manner based on human players' input. Here, the robot uses stochastic operations to divide, multiply, and skip certain beats in the detected rhythm in an effort to variate the input rhythm while keeping its original feel. Different transformation coefficients could be adjusted manually or automated to continuously control the level of stochastic operation, effectively manipulating the similarity levels between human's input and Haile's responses. In the third mode, *Perceptual Transformation*, Haile analyzes high level musical percepts in the human's rhythm, such as rhythmic stability, and responds by choosing and playing other rhythms from its library that have similar levels of stability to the original input. In this mode Haile automatically responds after a specified phrase length.

The previously described modes, *Imitation*, *Stochastic Transformation*, and *Perceptual Transformation*, formed sequential interactions using decentralized call-and-response routines between human players and Haile. In comparison, the next two modes—*Beat Detection* and *Simple Accompaniment*, facilitated synchronous interaction where humans play simultaneously with the robot. In *Beat Detection* mode, Haile tracks the tempo of and beat of the input rhythm using the Max/MSP object `beat~`. The `beat~` object was design to analyze the beat in prerecorded songs, but in a live setting, when the tempo variates fluently and when human players naturally adjust to the robot's tempo, beat detection becomes unstable and difficult to adjust to. Haile, therefore, uses `beat~` to listen for a short period of times (5–10 s) and then locks the tempo before joining in. The *Simple Accompaniment* mode facilitates simpler yet effective synchronous interaction where Haile plays prerecorded MIDI files, allowing his co-players to interact by entering their own rhythms or by modifying elements such as drum-head pressure to transform timbres in real-time. In this mode composers can feature structured compositions without algorithmic transformation or any symbolic input from co-players. This could be useful, for example, in sections of synchronized unison where humans and Haile play together.

The most sophisticated mode of interaction we developed for Haile is the *Perceptual Accompaniment* mode, which combines synchronous, sequential, centralized, and decentralized operations. In this mode, Haile listens and analyzes human players input while playing along simultaneously. This modes also allowed for local perceptually based call-and-response interactions with human players. The mode employs the amplitude and density perceptual modules described previously, allowing Haile to play short looped sequences (captured during the *Imitation* and *Stochastic Trans-*

formation modes), while listening to and analyzing the amplitude and density curves of human's input in real time. Haile could then transform the looped sequence based on the amplitude and density coefficients of the human players. For example, when the analyzed rhythmic input from the human players was dense, Haile could play sparsely, providing only the strong beats and allowing humans to perform solos with higher level of note density. When humans played sparsely Haile could improvise using high density rhythms that are based on stochastic and perceptual transformations. Haile could also respond directly to the amplitude of the human players so that the stronger humans play, the louder Haile plays to accommodate the human dynamics.

Two compositions were written to explore these interactions modes between Haile and human players, each using a different combinations of perceptual and interaction modules. The first composition, titled *Pow*, premiered at the Eyedrum Gallery in Atlanta as part of the Listening Machines concert in January 2005. The second piece, titled *Jam'aa* [7], was commissioned by and performed at the Hama'abada Performing Art Center. It premiered in Jerusalem, Israel, in March 2006.

5.2.2 Leader-Follower Interaction

In addition to the fundamental interaction modes described above, we developed a turn-taking scheme to facilitate leader-follower interactions with Haile, while applying a beat detection algorithm in the context of a live jam session. When musicians interact in an improvisatory manner, such as in drum circles, they often switch leadership roles organically: at different times during the session, one of the musicians would take the lead using techniques such as playing loudly, densely or in a different tempo or beat. The other musicians would often follow the leader until another musician takes the music in a new direction, as previous leaders become followers. To apply this model to human-robot musical interaction with Haile, we decided to use beat and tempo changes to signify leadership cues. When human co-players change the tempo, Haile identified that human as a leader, and in response entered a "Follower Mode" where it detected the beat, listened to and recorded the human rhythms, while providing accompaniment in synchronous manner. When the human co-players stayed at a steady tempo for a predetermined amount of beats, Haile inferred that it could exit "Follower Mode" and enter a "Leader Mode" where it can lock to the accompaniment tempo in one of its arms and play an improvised solo with the other arm. The improvised rhythm was generated using stochastic transformation of the previously recorded human rhythms. In general, this mode was well received by human co-players, although some criticism was voiced regarding the limited options humans were provided with to enter and leave the different modes. Since tempo transitions were often short in duration, humans could only lead during a short period of time, and were not able elaborate on their rhythms freely.

To address this criticism, we added volume and note density as factors that could help determine leadership, since these musical parameters often serve as an intuitive

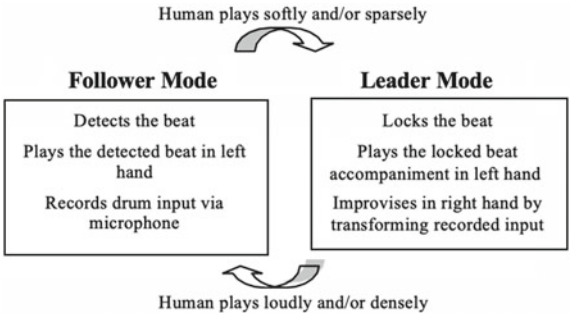


Fig. 5.4 Leader-follower interaction scheme for Haile with volume/density leadership cue

cue in human-human musical interactions. In the revised interaction, when the human drummers plays louder and more dense rhythm, Haile detects the human as the leader, and continues to apply its beat-detection algorithm. Therefore, in the revised interaction scheme, humans can lead for longer periods of time, with more opportunity to play sophisticated rhythms for Haile to build upon. This leader-follower interaction scheme is shown in Fig. 5.4.

5.2.3 Evaluation

In order to evaluate the effect of Haile's embodiment on synchronization, turn taking and learning, we conducted an experiment that assessed how well humans synchronize and learn rhythmic patterns with and without visual cues from the robot. The study showed that visual cues can influence synchronization in tasks that involve difficult rhythmic patterns and that participants showed a tendency to learn new patterns faster while receiving visual cues from the robot.

5.2.3.1 Related Work

Rhythmic synchronization is one of the most fundamental tasks in ensemble playing, as both auditory and visual cues can have a significant affect on the level of synchronization between two or more musicians. Luck and Sloboda [8] studied the synchronization between a human's tapping and visual cues depicting the beats using a point-light representation of a conductor. The researchers examined the effect of various conducting movement rates and showed that the highest correlation between taps and the conductor occurred in correlation to acceleration of the trajectory of the conductor's hand. Similarly, Repp has shown that visual cues assist with auditory cues on synchronization [9]. In a preliminary study we explored synchronization in an ensemble performance and found that for experienced drummers, visual cues helped synchronize gestures, but for novices visual cues decreased synchronization.

However, our sample size of 6 participants was too small, we did not control for the difficulty level of the rhythmic pattern, and did not vary the order of the conditions.

5.2.3.2 Motivation and Hypothesis

The goal of the experiment was to measure the effect of visual modalities on ensemble playing in a controlled manner. The experiment was designed to accurately measure synchronization of rhythmic motifs between a single musician and Haile in an effort to test two hypotheses: 1. participants would synchronize better with visual cues, and 2. participants would learn the patterns faster with visual cues.

5.2.3.3 Method and Stimuli

The stimuli used in the experiment was a common seven 16-beat drum patterns introduced to the drum pedagogy literature by Povel and Essen in 1985 [10]. The patterns were played in a tempo of 240 ms per beat and repeated in a loop for one minute. The duration and onset sequences of the patterns are shown in Fig. 5.5. All patterns contained the same total number of hits and the same number of hits of each duration, but at different temporal positions in a bar. All patterns were new to all the participants. The difficulty level of the patterns were increased with each pattern based on the pedagogy suggested by Povel and Essen.

5.2.3.4 Participants and Procedures

Twenty individuals with different levels of music background participated in the experiment. Participants were randomly divided into Group A and Group B, each

Pattern	Duration Sequence	Onset Sequence
1	1 1 1 1 3 1 2 2 4	
2	1 1 2 1 1 2 1 3 4	
3	1 1 2 1 3 1 2 1 4	
4	1 2 1 1 1 2 1 3 4	
5	1 1 1 1 2 1 2 3 4	
6	1 1 1 2 2 3 1 1 4	
7	1 1 1 2 1 1 3 2 4	

Fig. 5.5 Rhythmic patterns used in the experiment

presented with a different order of pattern playing. For Group A, participants were asked to follow the seven patterns played by Haile twice. During the first round, they could see Haile's movement. During the second round Haile was occluded with a screen so that the participants would play without making visual contact with the robot. Participants in Group B followed the same procedure, but in reverse order. They played the seven patterns without visual cues the first time and then with the visual cues the second time.

5.2.3.5 Equipment

To detect and measure Haile's strike onsets, a piezoelectric sensor was placed on the surface of the drum to detect vibration. An Arduino microcontroller was used as an AC/DC converter, which read the piezo output voltage value and transformed it into digital form. The Arduino software sent a measurement in milliseconds when the voltage exceeded a certain threshold; this threshold could be adjusted to change the level of sensitivity in different experiment environments. Additionally, a microphone was used to record the acoustic sound of the drum as a reference wave file. Human players used an electronic drum pad as the input device to collect onset data as MIDI data, which was connected to ProTools software for recording. See block diagram of the system in Fig. 5.6.

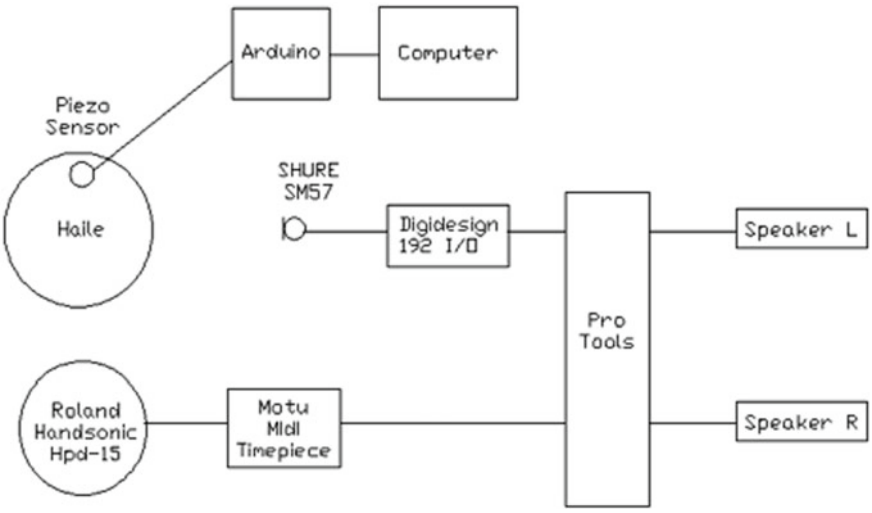


Fig. 5.6 System diagram for the experimental evaluation setup

5.2.4 Data Analysis

5.2.4.1 Noise Data Removal

A threshold of 200 ms was set as a minimum interval between two of Haile's onsets, based on the tempo used for the patterns corresponding to 240 ms per beat. This reduced the possibility of interpreting a single strike with large vibrations as multiple onsets. If the time interval between two note detections was less than 200 ms, the second detected onset was ignored. We chose 200 ms as a threshold to eliminate noise and false onsets as it was short enough to allow for onset shifts due to robot's mechanical friction yet long enough to eliminate most of the double triggered onsets.

We used a similar minimum interval threshold method for removing noise and false onsets from the participants' MIDI data. Noise data could be generated in one of two ways: the elasticity of the drum skin might cause a double hit or the simultaneous strike of two sections of the segmented MIDI drum pad (this would cause two MIDI notes to be recorded at the same time). To account for these possible false onsets we set a minimum interval length threshold of 10 ms.

5.2.4.2 Onset Sequence Matching

In order to measure the synchronization between the robot and the human players, we identified each of their corresponding onset pairs. We regarded the piezo onset sequence played by the robot as a reference and tried to match it with the MIDI sequence played by the humans. To address the matching problem, we devised an algorithm based on the Dynamic Time Warping algorithm (DTW), designed to detect an optimal alignment between two given time-dependent sequences under certain matching restrictions [11].

To determine an optimal path, defined by the matching points between sequences X and Y on the two axes of a matrix grid, one could test every possible warping path between X and Y (see Fig. 5.7). However, this technique would result in a high

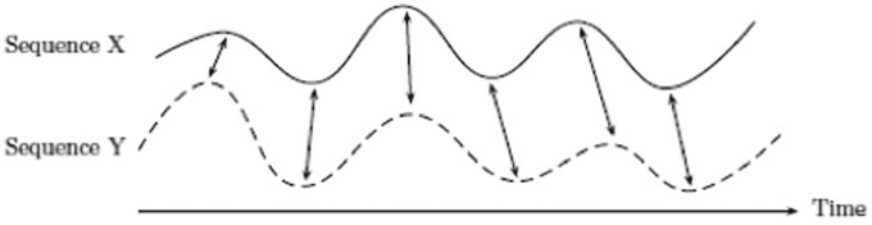


Fig. 5.7 Dynamic Time Warping

computational complexity that is exponential in lengths to N and M. To reduce the complexity, dynamic programming can be applied. We defined the accumulated cost matrix as:

$$D(n, m) = DTW(X(\frac{1}{n}, Y\frac{1}{m})) \tag{5.1}$$

with the cost being:

$$c(X_n, Y_m) \tag{5.2}$$

The resulting accumulated cost matrix D can then be computed by the recursion:

$$D(n, m) = minD(n-1, m-1), D(n-2, m-1), D(n-1, m-2) \tag{5.3}$$

For

$$1 < n < N + 1 < m < M \tag{5.4}$$

Instead of finding all possible routes through the grid that satisfy the constraints, the algorithm works by keeping track of the cost of the best path to each point on the grid. While populating the accumulated cost matrix, any path can potentially be define the lowest cost path. However, the optimal path can be traced back once the entire matrix is populated. Several constraints were applied in the process, such as that multiple MIDI onsets cannot being matched to single piezo onset and one MIDI onset cannot being matched to multiple piezo onsets.

As a penalty, we set

$$c(x_n, y_m) = 500\,\text{ms} \tag{5.5}$$

when no match is found. Based on our observations and analysis of the experiment data, a player was not likely to try to make a MIDI onset that was more than 500 ms away from the piezo onset that he/she intends to follow. Participants either played their onset closer to the piezo onset or missed it altogether. Based on these observations, we set a window with length of 500 ms and only allowed onsets inside the window to be selected as possible matches, which increased computation speed. This algorithm had some advantages compared to simple identification of nearest onsets (see: Fig. 5.8). The upper part of the Figure shows piezo onsets from 40 to 45 s, and the lower one shows MIDI onsets during the same period. For the 3rd red onset R3, it could be assumed that the player intended to catch the 3rd piezo onset B3, but played a little late. R4 is supposed to be matched with B4. With the simple algorithm, R3 could be matched to both B3 and B4 since it is the closest point to both. If only one pair was to be selected, B4 would be matched to R3, thus making the wrong match and abandoning B3 and R4 onsets. However, with the DTW-based algorithm, matches were made correctly.

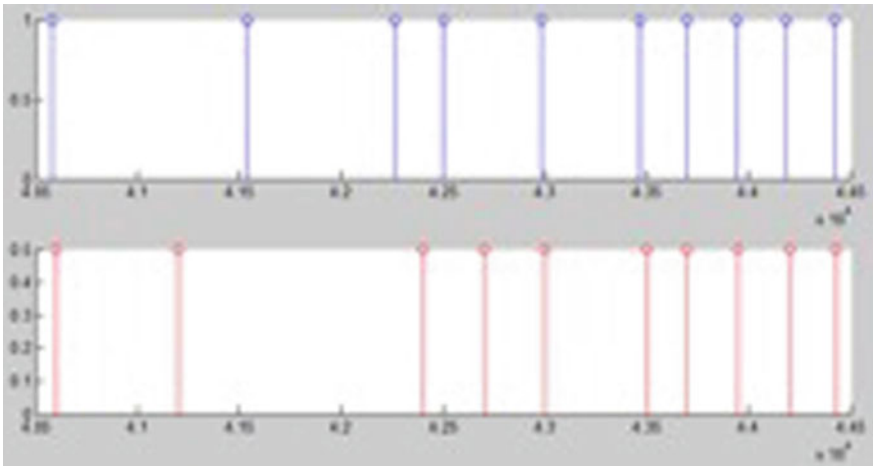


Fig. 5.8 One segment of piezo and MIDI onset sequences

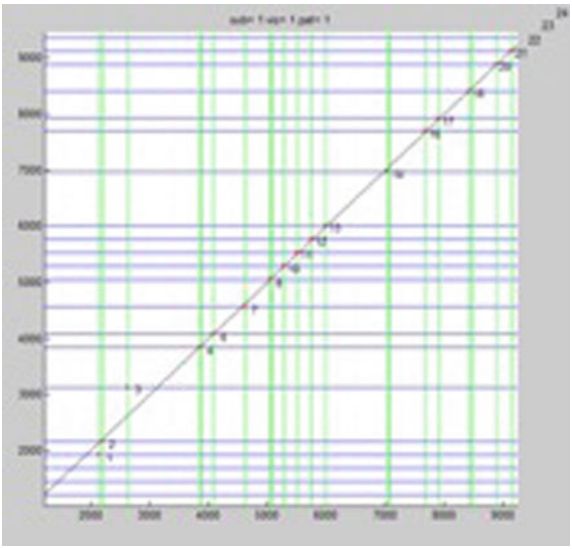


Fig. 5.9 Matching result of first 10 s for one of the participants in the experiment

Figure 5.9 depicts the first 10 s for the first participant, matching results with visual cues for pattern 1. Both x and y axes depict time. Green lines depict the human player's onsets, and blue lines depict piezo onsets. The red dots mark the determined matches. The black line shows y = x as a reference for perfect synchronization. The figure also shows some noisy onsets, which has been filtered off.

5.2.5 *Results*

After matching all the pairs, statistical analysis was conducted to identify the time difference between participants' onset time and piezo onset time for each matching pair. Absolute time difference was used to analyze synchronization, and original time difference was used analyze the prediction/lead-lag effect during learning. We applied 3-way Analysis of Variance (ANOVA) to both versions of time difference, with respect to the parameters: visual cue order (experiment group), visual cue on/off, and pattern. The ANOVA analysis was applied to the full 1 min of each pattern's playtime, as well as to the first 10, 10–20, 20–30 and 30–60 s, in order to observe the performance differences in different learning periods. The analysis of data from the experiment showed the following effects of visual cues (Fig. 5.10):

1. Overall synchronization accuracy: the analysis of absolute time difference shows that two out of seven patterns, (4 and 7), which we labeled as medium and high difficulties, showed significant difference in the synchronization metric between the visual on/off conditions (see Fig. 5.9). The metric for these patterns was worse in the case of visual off condition. Differences were not significant with easier patterns. We infer that visual cues aid synchronization in case of more difficult patterns. Analyzing the data of the relatively steady latter half of the sessions, from 30 to 60 s, we also observed a trend wherein the mean synchronization metric was slightly better for visual-on condition compared to that for visual-off.

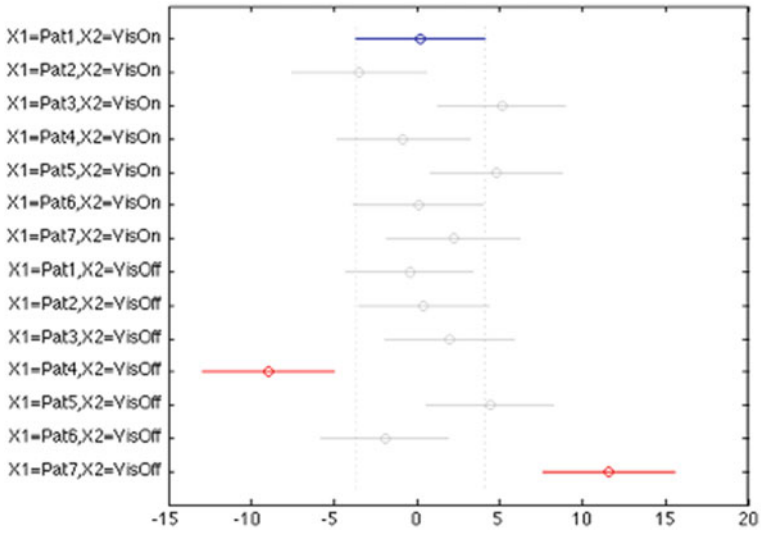


Fig. 5.10 Comparison of synchronization metric for the 7 patterns under visual on/off conditions

2. The data of the relatively steady latter half of the sessions, from 30 to 60 s, indicate a trend wherein the mean synchronization metric was slightly better for visual-on condition compared to that for visual-off.
3. Lead and lag: the analysis on original time difference shows that participants played with a large lag when playing with visual cues at first 10 s, and played better after the first 10 s (Fig. 5.9). The overall performance for 1 min, with presence of visual and without, did not show significant difference. This suggests that players needed some time after seeing the visual cues and only then play the onset during the learning period. They play better with visual cues after learning period.

5.2.6 Conclusion

When learning some rhythmic patterns at a medium tempo, our experiment shows that robotic visual cues can be helpful in facilitating synchronized performance, at least when human players are first introduced to new patterns. Participants also showed a tendency to learn new patterns faster when they had visual cues with the robot. Additionally, they tended to lag with visual cues in the early learning period, but to play better later after learning with visual cues. However, the latter two effects were not statistically significant.

5.3 Synchronization with Shimon's Music-Making Gestures

As discussed in the previous section, a benefit of robotic musicianship in contrast to other computer-generated music lies in the fact that the visual embodied connection between the human and the robotic musician can help synchronize the ensemble's playing.

Most previous works on synchronization between humans and robots in a musical or pseudo-musical setting (e.g. [12, 13]) have been concerned with the synchronization of tapping and drumming, and not with structural and melodic interactions.

To evaluate the benefit of embodiment on melodic robotic musicianship, we ran a study that tried to isolate the embodied and visual parts of synchronization when playing with the Shimon marimba-playing robot (see: Sect. 2.3). We hypothesized that the visual and physical presence of the machine can help human players anticipate the robot's timing, and thus coordinate their playing with that of the robot.

5.3.1 Hypotheses

In particular, we tested the following hypotheses regarding a human musician's ability to synchronize their playing with an artificial (computer or robotic) musician:

H1 Synchronization is enhanced by the physical presence of a computer musician (Embodiment effect)
H2 Synchronization is enhanced by visual contact with an embodied computer musician (Visual contact effect)
H3 The above effects are more pronounced in situations of low accuracy on the part of the computer musician.

5.3.2 Experimental Design

To evaluate these embodiment effects on human-robot musical synchronization, we conducted a 3×2 within-participant study manipulating for level of embodiment and robot accuracy.

Six experienced pianists from the Georgia Tech Music Department were asked to repeat a call-and-response segment a Jazz standard with a robotic musician. The interaction started by the pianist playing a 7-note introductory phrase on a grand piano. The robot detected the tempo and bar synchronization of the phrase and responded in a rhythmic three-chord pattern on the marimba. The pianists were asked to synchronize a single bass note with each of the robot's chord, as best they could.

Each pianist repeated the call-and-response sequence 90 times. They were asked to play at a variety of tempos of their choosing.

The timing of the human's playing was recorded through a MIDI interface attached to the grand piano, and the robot's playing time was also recorded, both to millisecond precision. MIDI delays between the human and the robot were accounted for.

5.3.3 Manipulation I: Precision

In the first half of the sequences (the `PRECISE` condition), the robot was programmed to play its response in the precise tempo and on-beat of the human's call phrase. In this condition, the pianists were informed that the robot will try to match their playing precisely. In second half of the sequences (the `IMPRECISE` condition), the robot was programmed to play its response either on tempo and on-beat to the human's call phrase, slightly slower than the human's introduction phrase (either 50 ms too slow, or 100 ms too slow), or slightly faster (either 50 ms too fast, or 100 ms too fast). The first chord is always "on beat" and only the subsequent chords suffer from accumulative delay.

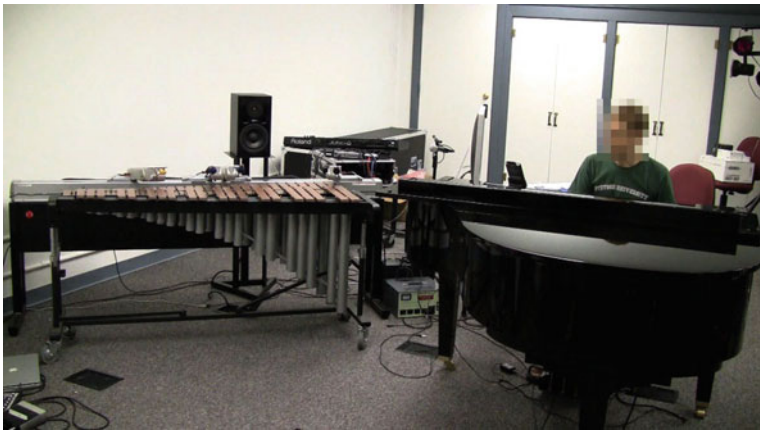


Fig. 5.11 Experimental setup showing the human pianist on the right, and the robotic marimba player *Shimon* on the left

The pianists were informed that the robot might play slightly off their proposed beat, but that its response will be consistent throughout each individual response sequence. Also, the pianists were asked to try to synchronize their playing with the actual notes of the robot, and not with their own "proposed" tempo and beat.

5.3.4 Manipulation II: Embodiment

Within each half of the trials—for a third of the interaction sequences (the `VISUAL` condition), the pianists were playing alongside the robot to their right (as shown in Fig. 5.11), enabling visual contact with the robot. In another third of the interaction sequences (the `AUDITORY` condition), the robot is physically present, but separated from the human musician by a screen. In this condition, the human player can hear the robot move and play, but not see it. In the remaining third of the interaction sequences (the `SYNTH` condition), the robot does not move or play. In this condition, the human player can hear a synthesized marimba play over a set of headphones. The order of the conditions was counterbalanced for each pianist.

In both the `AUDITORY` and the `SYNTH` conditions there is no visual contact with the robot. Furthermore, in both the `VISUAL` and the `AUDITORY` condition there is an acoustic note effect indicating the presence of a physical instrument and a physical player, and in addition, the robot's motor noise can indicate to the pianist that the robot is in motion.[1]

[1] For reference, the motor noises peaked at 51.3 dBA measured at a distance of 1.5 m length and 1.5 m height from the center of the base of the robot. The measurements were made using a calibrated Apex 435 condenser microphone. Measured under the same conditions, without the motors running, the ambient noise in the room was measured at 42.5 dBA.

5.3.5 Results

To account for robot accuracy, and the resulting human uncertainty, we pose three auxiliary hypotheses, differentiating between the three response chords. This is due to the different musical role each chords plays: the first chord occurs an eighth beat after the introductory phrase, so that the pianists can easily synchronize with the robot by simply playing according to their original tempo. The second chord reveals the robot's perceived tempo, and its temporal placement may vary in the `IMPRECISE` condition. Since all three chords play at a fixed tempo, the temporal placement of the third chord can be inferred by the interval between the first and the second chord, in which case the synchronization can, again, be achieved by rhythm alone. We thus pose the following auxiliary hypotheses:

H3a Synchronization in the `PRECISE` condition is higher than in the `IMPRECISE` condition
H3b Synchronization of the first chord is highest
H3c Synchronization of the second chord is lowest.

All three auxiliary hypotheses are supported by our findings.

The offsets in all trials in the `PRECISE` condition are significantly lower than those in the `IMPRECISE` condition: 69.63 ± 130.53 versus 86.91 ± 97.14, $t(1513) = -2.89$, $p < 0.001$***.

Furthermore, as can be see in Fig. 5.12, the offsets (absolute delays) for the first chord are the lowest (49.35 ms), those of the second chord are significantly higher (116.16 ms), and those for the third chord are lower than the second, but not as low as the first (67.79 ms). A one-way ANOVA shows that the three metrics differ significantly ($F(2,1512) = 47.14$, $p < 0.001$***), and pairwise comparison shows that each of them is significantly different from each of the other at $p < 0.001$. We therefore confirm our auxiliary hypotheses **H3a**, **H3b**, and **H3c**, and use these metrics separately in order to evaluate Hypothesis 3.

5.3.5.1 `PRECISE` Condition

We first evaluate hypotheses **H1** and **H2** in the `PRECISE` condition, phrased as follows:

H1a When the robot is following the human lead precisely, synchronization is enhanced by the physical presence of a computer musician (Embodiment effect with precise robot)
H2a When the robot is following the human lead precisely, synchronization is enhanced by visual contact with an embodied computer musician (Visual contact effect with precise robot).

Comparing the offset (absolute value of the delay) between pianist and robot in the `PRECISE` condition, we find a significant difference between the three embodiment

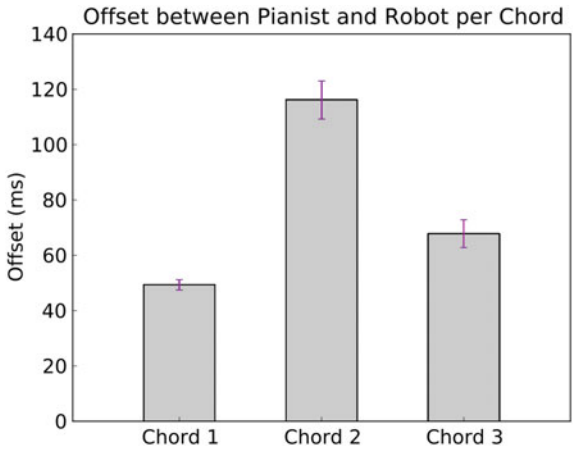


Fig. 5.12 Mean offset in milliseconds between pianist and robot per chord of the response phrase

conditions using one-way ANOVA [F(2,798) = 3.55, $p < 0.05$*] (Fig. 5.13). Post-hoc tests reveal that this is due to the AUDITORY condition being less precise than the other two conditions[T(799) = 2.65, $p < 0.01$**]. We thus find no advantage to either visual contact or physical presence with the robot, refuting both Hypothesis **H1a** and **H2a**. This can be attributed to the fact that since the robot plays precisely according to the pianist's cue, the musicians can use their internal rhythm to synchronize with the robot. For possible reasons for the negative effect of the AUDITORY condition, see our discussion below.

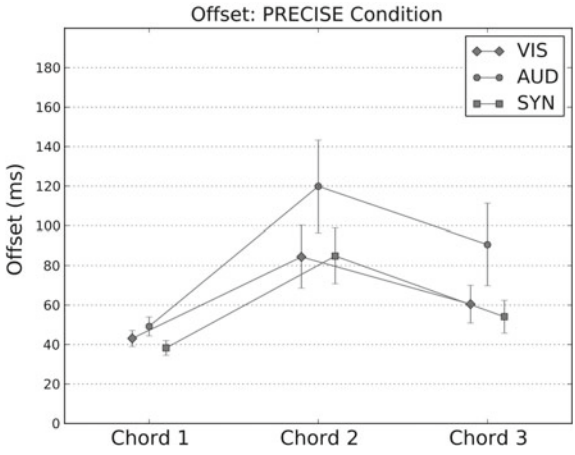


Fig. 5.13 Mean offset in milliseconds between pianist and robot in the PRECISE condition

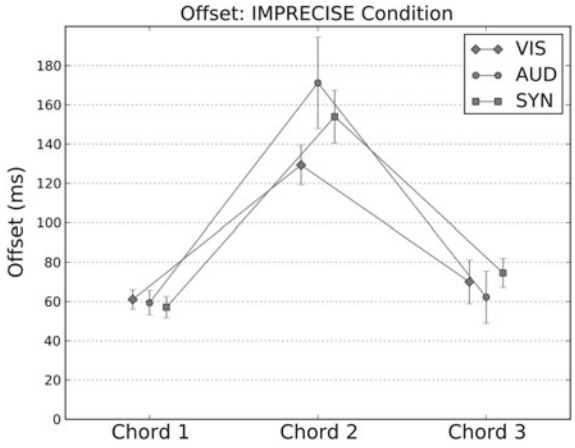


Fig. 5.14 Mean offset in milliseconds between pianist and robot in the `IMPRECISE` condition

5.3.5.2 `IMPRECISE` Condition

When the robot changes the detected tempo of the introductory phrase, we expect to detect more of a difference in synchronization between the human pianist and robot. Specifically, we test:

H1b When the robot is not following the human lead precisely, synchronization is enhanced by the physical presence of a computer musician (Embodiment effect with imprecise robot)

H2b When the robot is not following the human lead precisely, synchronization is enhanced by visual contact with an embodied computer musician (Visual contact effect with imprecise robot)

H3 The above effects are more pronounced in situations of uncertainty.

Figure 5.14 shows the mean and standard error for all trials in the `IMPRECISE` condition. For the first and third chord, we see no difference between the conditions, indicating that, indeed, the human musicians could use the auditory and rhythmic cues to synchronize these two chords. In particular, it is notable that the first two chords are enough for the participants to synchronize the third chord based on the same interval, supporting Hypothesis **H3**.

However, for the second chord—the timing of which has some uncertainty—the offset is smaller for the `VISUAL` condition compared to both non-visual conditions, albeit not significantly: `VISUAL`: 129.32 ± 10.01 ms; other conditions: 162.80 ± 13.62 ms, [$T(182) = -1.66$, $p = 0.09$], suggesting that visual cues may be used to synchronize the relatively unpredictably-timed event, but more study is required to evaluate Hypothesis **H2b**.

The "offset" discussed above is the absolute error between the human's key-hit and the robot's marimba-strike. The effect of visual contact is, however, more apparent, when looking at the sign of the error: evaluating the directional delay, we

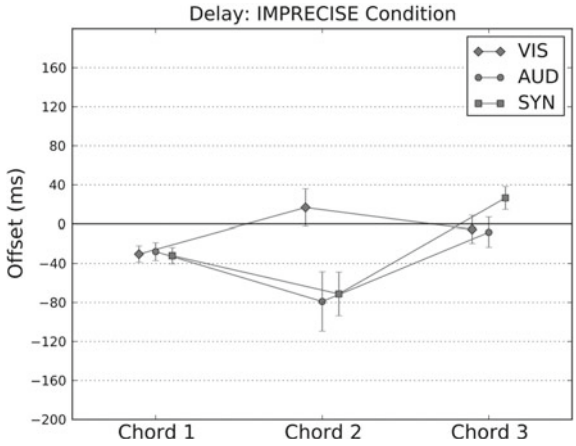


Fig. 5.15 Mean delay in milliseconds between pianist and robot in the `IMPRECISE` condition

find a significant difference between the three conditions on Chord 2 [$F(2,181) = 4.80$, $p < 0.01$**]. Specifically, the `VISUAL` condition is significantly different from other two conditions (Fig. 5.15): `VISUAL`: 16.78 ± 19.11 ms; other conditions: -75.21 ± 18.95 ms, [$T(182) = 3.10$, $p < 0.01$**]. This finding, too, supports Hypothesis **H2b**.

In particular, we find that trials in the `VISUAL` condition to be delayed with respect to the robot, whereas the trials in the non-visual conditions pre-empt the robot's playing, indicating that pianists *react* to the robot's movement when they can see it, but try to anticipate the robot's timing when they cannot see it.

5.3.5.3 Effects of Tempo

As a post-hoc analysis, we also found that the benefits of a visual connection increase at slower playing tempos. Figures 5.16 and 5.17 show the errors for all trials over and under 100 beats per minutes, respectively, showing a larger embodiment effect for slow trials than for fast trials.

While the `AUDITORY` condition is significantly more error-prone in slow trials than in fast trials (234.54 ± 56.25 ms [slow] versus 131.25 ± 10.75 ms [fast]; $T(57) = 2.14$, $p < 0.05$*), the error in the `VISUAL` condition is not affected by the decrease in tempo (138.59 ± 17.62 ms [slow] versus 119.43 ± 11.54 ms [fast]; $T(60) = 0.94$). As above, the effect on the `SYNTH` condition is similar to the `AUDITORY` condition, but less pronounced.

For signed delays, we also find more of the embodiment effect on Chord 2 reported above in slow trials, compared to fast trials (bpm < 100: $F(2,69) = 4.83$, $p = 0.01$; bpm > 100: $F(2,105) = 3.11$, $p = 0.048$).

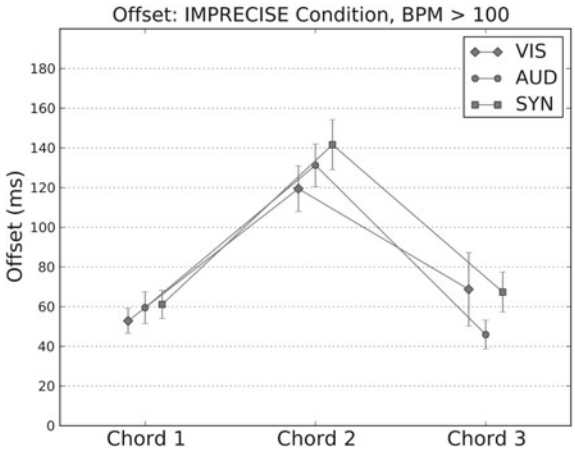


Fig. 5.16 Mean delay in milliseconds between pianist and robot in the `IMPRECISE` condition for trials over 100 bpm

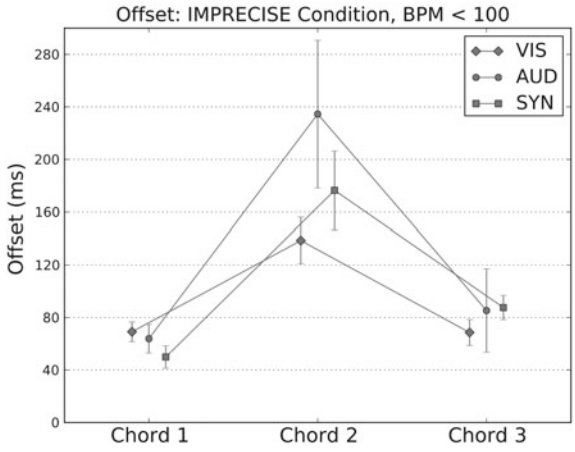


Fig. 5.17 Mean delay in milliseconds between pianist and robot in the `IMPRECISE` condition for trials under 100 bpm

5.3.6 Discussion

In our experiments, we find that visual contact with the robot contributes only partially to the degree of synchronization in a call-and-response interaction. The effect of embodiment without visual contact, compared to a disembodied musician, seems to be even less pronounced, and sometimes detrimental.

In the case where the robot does not intentionally change the tempo, we see no advantage to visual contact or embodiment over a synthesized music player. We believe that this is due to the fact that the robot precisely follows the pianist's rhythm, allowing for perfect synchronization simply by playing along in the same rhythm, without any input from the robot.

In the case where the robot slightly alters the tempo in response to the human's playing, we find that the pianists' ability to synchronize with the robot is significantly

reduced. For the second chord (the only chord in which there is an uncertainty in timing), visual contact reduces the error compared to the auditory and synthesized condition. In particular, visual contact allows the pianists to *react* to the robot instead of preempting the timing of their playing. This indicates that the pianists use the visual cues to time their playing.

By the third chord, the human players seem to be able to achieve a high level of synchronization regardless of the embodiment of the robot. This may indicate that they resort again to a rhythmic cue based on the first two chords.

We also find that visual contact is more crucial during slow trials, possibly suggesting that visual cues are slow to be processed and do not aid much in fast sequences. For example, it may be that during fast sequences, the pianists did not have time to look at the robot. In general, it seems that pianists use visual information when they can, but can resort to rhythmic and auditory cues when necessary and possible.

Interestingly, it seems that the synthesized condition is less error-prone than the present-but-screened (`AUDITORY`) condition in both precise and imprecise playing modes of the robot. This may be due to the fact that the pianists try to use the existing motor noise from the robot as a synchronization signal, but find it to be unreliable or distracting.

5.3.7 Audience Appreciation

We also tested the effects of visual contact and embodiment on audience appreciation. In this experiment, we filmed two pianists playing in three different improvisation settings each with the robot. We wanted to test how embodiment and visual contact affects joint improvisation as judged by an audience. The physical setup was similar to the previous experiment, and the conditions were similar to those in the "Embodiment" manipulation, namely `VISUAL`, `AUDITORY` (present but occluded), and `SYNTH`. The only difference was that, in this experiment, the synthesized sound came out of a speaker behind the robot, instead of through headphones.

In this experiment we tested the following hypotheses:

H4 Visual Contact between a robot and a human musician positively affects audience appreciation of a joint improvisation session

H5 Physical embodied presence positively affects audience appreciation of a joint improvisation between a human and a machine.

5.3.7.1 Experimental Setup

The pianists' sessions were videotaped, and from each session, a 30 s clip was extracted by choosing the 30 s after the first note that the robot or computer played. We posted these video clips onto a dedicated website, and asked an online audience to rate the clips on eleven scales. Each scale was a statement, such as "The robot

played well" (see: Table 5.1 for all scales), and the participants were asked to rate their agreement with the statement on a 7-point Likert scale between "Not at all" (1) and "Very much" (7). Participants watched an introductory clip familiarizing them with the robot, and could play and stop the clip as many times as they liked.

For each pianist, the order of conditions was randomized, but the conditions were grouped for each pianist, to allow for comparison and compensation of each pianist's style. The wording of each scale was matched to the clip, i.e. in the SYNTH condition, the word "robot" was replaced by the word "computer".

We collected 30 responses, out of which 21 were valid, in the sense that the participants rates all three performances of at least one pianist. The reported age of the respondents ranged between 25 and 41, and 58% identified as female.

5.3.7.2 Results

In order to compensate for each pianist's style, we evaluated the difference between conditions for each participant and each pianists. We then combined the results for both pianists across all participants.

Table 5.1 shows the results of comparing the VISUAL condition to the AUDITORY condition, and the comparison of the AUDITORY condition to the SYNTH condition. The first comparison indicates the effect of visual contact between the pianist and the machine, the second comparison indicates the effect of physical co-presence, acoustic sound, and ambient motor noise in the absence of visual contact.

5.3.7.3 Effects of Visual Contact

We found a significant difference in audience appreciation of the improvisation session between the visual-contact and occluded conditions, on all scales but one (overall enjoyment). Even though the robot uses the same improvisation algorithm in all conditions, audiences felt that in the VISUAL condition the robot played better, more like a human, was more responsive, and seemed inspired by the human. In addition, we find that the human player, too, was rated as more responsive to the machine, and as more inspired by the robot. The overall rating of the duo as being well synchronized, coordinated, connected, coherent, and communicating was also significantly higher in the VISUAL condition.

These findings support hypothesis **H4**, indicating that visual contact between human and robot contributes significantly to the audience's appreciation of robotic musicianship.

5.3.7.4 Effects of Embodied Presence/Acoustic Sound

In contrast, we could not support hypothesis **H5**, and found only little significant difference in audience appreciation between both occluded conditions. For most

Table 5.1 Effects of visual contact and embodied presence/acoustic sound on audience appreciation of a number of scales. t numbers indicate 1-sample t-Test with $\bar{x} = 0$ as the null hypothesis. $*p < 0.05$; $**\ p < 0.01$; $***p < 0.001$

Scale	VIS-AUD		AUD-SYN	
	$\bar{x} \pm \sigma$	t(38)	$\bar{x} \pm \sigma$	T(38)
I enjoyed this performance	0.28 ± 1.28	1.36	0.26 ± 1.81	0.87
The robot played well	0.92 ± 1.42	4.00 ***	0.23 ± 1.78	0.80
The robot played like a human would	1.11 ± 1.25	5.37 ***	0.05 ± 1.38	0.23
The robot was responsive to the human	0.54 ± 1.28	2.60 *	0.67 ± 1.68	2.44 *
The human was responsive to the robot	1.08 ± 1.79	3.71 ***	-0.28 ± 1.87	−0.93
The duo was well-coordinated and synchronized	1.00 ± 1.48	4.15 ***	-0.28 ± 2.07	−0.84
The human seemed inspired by the robot	1.13 ± 1.90	3.67 ***	-0.26 ± 1.71	−0.93
The robot seemed inspired by the human	0.67 ± 1.37	3.01 **	0.64 ± 1.70	2.32 *
The two players felt connected to each other	0.97 ± 1.58	3.75 ***	0.24 ± 1.81	0.79
The duo felt like a single unit	0.95 ± 1.63	3.58 ***	-0.08 ± 2.06	−0.23
The duo communicated well	1.11 ± 1.74	3.85 ***	-0.05 ± 1.99	−0.16

scales, there was no difference whether the machine's improvisation came out of the speaker or whether the robot played physically on the keys. The two scales on which there was a significant advantage for the physically embodied/acoustic condition was the robot's responsiveness, and the robot's inspiration by the human player.

Thus, while participants did not rate the robot's playing as better or more human, they did attribute more human-like characteristics to the acoustically playing robot. The robot seemed more responsive and more inspired when it was playing a real acoustic instrument.

Interestingly, occluding the physical robot seems to impair the duo's performance, as in all joint-performance ratings, there is no difference between the occluded physical robot and the synthesized (also occluded) speaker. It is possible that the pianist's engagement drops significantly when there is no visual contact.

We find it surprising that the robot's physical movement and acoustic sound does not contribute to the robot's rated performance quality. However, it is possible that the quality of the video recording and the digital quality of the audio recording are responsible for there being little effect on the audience. Further research could evaluate this question in a live-presence audience study.

5.4 Emotion Conveyance Through Gestures with Shimi

Beyond the benefits of synchronization, a robotic musician's embodiment also provides the ability to express emotion through nonverbal means as an effective tool to interact with people. This ability bears the promise of enhancing and enriching musical emotive communication. In recent years evidence has been mounting suggesting that there is a direct relationship between external physical behavior and emotional states [14]. In fact, Aviezer et al. found that body language broadcast emotional internal states even more reliably than facial expression [15]. Whether unconscious or deliberate, it is clear that humans are able to associate postures and movements with different emotions [16–20], which has been used in the field of Affective Computing to develop emotionally expressive machines [21]. It has been shown that such "Affective Channels" enable communication without increased cognitive function by the user, and help support natural machine-human social interactions.

In human-robot interaction, expressive robotic movements can be used as an additional form of emotive communication and a behavior which can carry influence in a social interactive environment [22]. Robots that respond to people's emotional states with such expressive gesture enable more natural and comfortable human-robotic interactions [23–25], which can help facilitate more engaging musical interactions. An emotionally responsive and expressive robot can be effective in social situations such as in learning and teaching environments by communicating compassion, awareness, accuracy, and competency [26].

Our work in Robotic Musicianship contributes to this research by examining a robot's ability to (1) express emotion using physical body gestures and (2) autonomously generate emotional behaviors. As part of our research we investigate what aspects of musical physical emotional expression can be translated to robotics. As part of the project, we also studied whether emotionally perceptive robots can offer increased levels of engagement in human robotic interactions by responding with expressive movements.

5.4.1 Related Work

5.4.1.1 Emotion Classification

Before describing how emotion display can be generated, it is important to present common methods of emotion classification. Some methods utilize a set of basic discrete emotions (happiness, sadness, fear, etc.) with more complex emotions addressing a combination of two or more of these fundamentals [27]. Other methods utilize a continuous dimensional model to classify emotion. These dimensions include valence, arousal, and less commonly, dominance, and stance [28, 29]. Critics of the theory of discrete emotions rely on brain research which shows that discrete brain regions cannot be mapped to discrete emotion categories [30]. Other studies demon-

strated that basic emotion views can be represented in neural correlated, but these mappings do not constitute one-to-one relationship rather demonstrate complex distributed networks [31, 32]. A newer theory describes dynamical discrete emotions and attempts to address the variability and context-sensitivity of emotions [33]. This approach has been deemed viable by researchers on both ends of the discrete versus continuous emotion debate [34].

5.4.1.2 Emotion in Virtual Agents

Emotionally expressive animations have demonstrated effectiveness by making the agents more relatable and lifelike, which encourages the viewer to empathize with and respond to the virtual agent as if it were human. For the last few decades computer animation researchers have addressed the challenge of turning living objects into expressive beings with rich personality. Lasseter describes the use of pose, motion, and acceleration in animating the iconic Pixar lamp, *Luxo Jr* [35], presenting techniques such as "staging" and "exaggeration" used to create unmistakably clear emotional display. "If a character is sad, make him sadder; if he is bright, make him shine; worried, make him fret; wild, make him frantic." Studies have also shown that these ideas are appropriate for robots as well. Gielnak and Thomaz demonstrate that exaggerated cartoon-like motions of a robot are most effective for yielding the benefits of increased partner engagement and entertainment value [36]. The interactive graphics-based agent, *Rea*, is a virtual real-estate agent that uses gestures, gaze, and facial expressions as an additional communicative and expressive layer to accompany her speech [37]. Nayak and Turk describe how emotional expression in virtual agents is achieved through a combination of facial expressions and body language including torso, shoulder, head, and leg movements [38].

5.4.1.3 Emotions in Robots

Animation techniques can be useful for social robotics [39], although robots may be more challenging than virtual agents as they are often less mobile, have fewer degrees-of-freedom, and can only support slower movement. However, the proximity and presence of a physical robot can have significant benefits for human-machine interaction. In this section, we describe related work in robotic communication through gesture and proximity.

Gesture and Faceless Robots—Gesture is an important facet of human communication and studies have shown that characteristics such as velocity, acceleration, and location (front versus side-oriented) can influence how people respond to robotic movement [40, 41]. Gesture often accompanies speech to help the speaker to formulate ideas and provide additional information to the observer. Salem et al. developed an integrated model of speech-gesture production to address this concurrence [42]. A few robots have been designed to use gesture to explicitly convey posture, motion, and gaze to convey and elicit emotion. Some of these robots are faceless (or facially

expressionless) and must rely only on the physical body and head movements. *RoCo* is a robotic computer stand designed to move in a subtly expressive manner and influence users' affective states by altering its posture [43]. *AUR* is a robotic lamp which utilizes both human-controlled and autonomous modules to generate expressive gestures and behaviors [44]. *Keepon* is a creature-like robot designed to facilitate emotional expression through its gaze and movements without active facial DoFs. It can convey emotions such as pleasure and dislike through bobbing (up and down and back and forth) and vibrating motions [45]. The NAO robot is a humanoid robot which also has a face, but no facial motors. It adjusts its arms, legs, head, and torso to arrange itself in emotionally semantic poses [46]. Similarly, both the DARwIn and Hubo robots have been programmed to utilize head and arm positions to express mood while dancing to music [47].

Presence and Proximity—Presence and proximity influence the way people perceive emotion. A physical presence (as opposed to virtual) can lead to more "altruistic and persuasive" perceptions of the robot [48]. Since spatial features that are used to support social behaviors can be used to trigger levels of comfort and discomfort [49–51] it is important to design mechanisms to control the proximity of the robot to the user. However, Shimi is a stationary robot so our experiment can evaluate presence, but cannot test the effectiveness of a dynamically changing distance between collaborators and how this may affect emotion perception and recognition.

5.4.1.4 Algorithmic Emotion Architectures

Affective Computing researchers have suggested a number of guidelines for the generation of emotional intelligence architecture. Picard describes the constituents of expressive machines as having both instinctual (spontaneous) and communicative (intentional) pathways, and calls for robot designer to develop systems flexible enough to account for both pathways [21]. A well known system for an emotionally expressive robot attempting to address both pathways is Kismet's [52]. Breazeal describes a system which enables facial expression transformation on a continuous *valence*, *arousal*, and *stance* scale. Velásquez describes the system, *Cathexis*, which is based on the six primary emotions (*anger, fear, disgust, sadness, happiness,* and *surprise*) [53]. An implementation of expressive dance motions NAO robot is described in [54] which uses automated scheduling of discrete motion primitives driven by beats and emotion in music. A more recent system designed for human-machine companions is presented by Traue et al. [55]. The system integrates the use of both discrete and continuous emotions. External events are subjected to an appraisal process and interpreted as discrete emotional stimuli. These discrete stimuli are mapped to a continuous space and averaged. The resulting value is used to manipulate the agent's behavior in response to the stimuli. These systems connect the agent's motivations to its affective behavior based on the fact that emotions arise out of one's internal goal [56, 57]. The motivations, or "drives" as they are often referred, may result by a number of factors including the agent's need for energy,

survival, communication, or even dancing. The agent's desire to satiate these goals influences its behaviors and thus emotions are born.

5.4.2 A System for Generating Emotional Behaviors

Our aim is to automatically create movements that correlate with specific emotions using a set of parameters and mathematical functions to define pose and motion. We implement our approach on Shimi, a robotic speaker dock (Sect. 2.4). Initially, we had hand designed gestures and movements that we interpreted as being emotionally communicative. Though effective, designing gestures in this manner is time consuming and only a finite set of movements can be designed. Here, we describe a system to allow Shimi to auto-generate and express varying levels of emotions, in a manner that would resonate with how human perceive such emotions. For example, humans can experience and describe different levels of happiness using language such as joyous, content, exuberant, satisfied, ecstatic, pleased, jubilant, overjoyed, etc. Our hypothesis was that with an algorithmic system that would manipulate and combine different emotion-driven gestures, Shimi could express similar emotive richness. This work is informed by the work of Breazeal [52], where transitions between facial expressions on a continuous scale allowed Kismet the robot to express a wide range of emotions with varying degrees of intensities. Our approach expands on this work by creating a system for expressive emotional behaviors without facial expressions, using only posture and motion, while still allowing to generate the discrete emotive gestures designed in the previous section.

5.4.2.1 Control Variables

In this section, we describe our method for algorithmically generating emotive behaviors. To do so, we first need to understand the variables that constitute emotional behaviors. An additional challenge is to convey varied levels of emotion using only the five DoFs available to Shimi.

Our approach to addressing this challenge was informed by Walbott [58] based of observations from Darwin [59]. Figure 5.18 presents a summary provided Walbott regarding relationships between posture and the movements inherent to different emotions, which indicates which characteristics (such as velocity, head position, etc.) can be used to control motion. Based on this analysis we can then generate different types of behavior. We generalize Darwin's observations as a set of relevant features for differentiating the behaviors.

1. *Posture Height*—representative of the relative distance between the height of a person's chest and height of the waist. In essence, how erect or crouched a person's torso is.

Table 1. Body movements and postures accompanying specific emotions (citations from Darwin, 1872/1965)

Joy	Various purposeless movements, jumping, dancing for joy, clapping of hands, stamping, while laughing head nods to and fro, during excessive laughter whole body is thrown backwards and shakes or almost convulsed, body held erect and head upright (pp. 76, 196, 197, 200, 206, 210, 214)
Sadness	Motionless, passive, head hangs on contracted chest (p. 176)
Pride	Head and body held erect (p. 263)
Shame	Turning away the whole body, more especially the face, avert, bend down, awkward, nervous movements (pp. 320, 328, 329)
Fear/terror/ horror	Head sinks between shoulders, motionless or crouches down (pp. 280, 290) convulsive movements, hand alternately clenched and opened with twitching movement, arms thrown wildly over the head, whole body often turned away or shrinks, arms violently protruded as if to push away, raising both shoulders with the bent arms pressed closely against sides or chest (pp. 291, 305)
Anger/rage	Whole body trembles, intend to push or strike violently away, inanimate objects struck or dashed to the ground, gestures become purposeless or frantic, pacing up and down, shaking fist, head erect, chest well expanded, feet planted firmly on the ground, one or both elbows squared or arms rigidly suspended by the sides, fists are clenched, shoulders squared (pp. 74, 239, 243, 245, 271, 361)
Disgust	Gestures as if to push away or to guard oneself, spitting, arms pressed close to the sides, shoulders raised as when horror is experienced (pp. 257, 260)
Contempt	Turning away of the whole body, snapping one's fingers (pp. 254, 255, 256)

Fig. 5.18 A table presented by Walbott [58] describing observations from [59]

2. *Shoulder Height*—representative of the relative distance between the waist and the shoulders.
3. *Arm Position*—the location of the arms in relation to the rest of the body ("close to sides" vs. "over the head")
4. *Gaze (Head Position)*—the direction and angle at which the head is positioned is indicative of where somebody's attention is and how welcoming or rejecting somebody is
5. *Body Activation*—the type of motion the body and arms exhibit (slow vs. fast) including
    a. *Up and Down Activation*
    b. *Left and Right Activation*
    c. *Rotational Activation*—twisting of the body and arms
6. *Head Activation*—the type of motion the head exhibits is classified into 2 parts
    a. *Positive Head Activation*—head nodding up and down
    b. *Negative Head Activation*—head shaking left and right
7. *Volatility or Periodicity*—a function of variation in the movements in terms of the positions a movement oscillates between and the rate at which this is done. A highly periodic movement would indicate a motion with a low volatility. (This

idea is supported by Sullivan et al. [60] who show anger to be accompanied by more spasmodic movements compared to sadness)
8. *Exaggeration*—the range of motion exhibited by each DoF (smaller vibrations vs. massive convulsions).

Though these parameters were derived from Darwin's observations of humans, we believe that all, or a subset of these parameters, can be useful for designing expressive movement in non-humanoid robots as well. On a robot the *posture, shoulder, arm,* and *gaze* parameters can be considered positional values for the DoFs which correspond to the relevant body parts. The *activation* parameters describe how the DoF positions change in time in relation to their "center values" described by these positional values. *Volatility* and *exaggeration* are modifying parameters which can be used to manipulate both the positional and activation values in time. In the following sections, we describe the system and control parameters in more detail. For Shimi, we use only the parameters relevant to its design and DoFs (see Fig. 5.21) which include posture height, gaze, head activation, volatility, and exaggeration.

5.4.2.2 System Design

Our emotional behavior generative system consists of three main sections: Emotion Tagging, Interpretation, and Expression (Fig. 5.19).

1. *Emotion Tagging* takes in sensor data such as sound or language and represents it as a particular emotion using the continuous approach of valence and arousal values or a discrete approach using specific words such as "anger" or "happy".
2. *Interpretation* maps the emotional tags to each of the control parameters.
3. *Expression* generates Emotive behaviors based on the control parameters using motion and body language.

A growth and decay function and motion primitives were used as input parameters to the system. The growth and decay function allows the robot to fluidly transition between emotional states. A homeostatic state represented by particular control

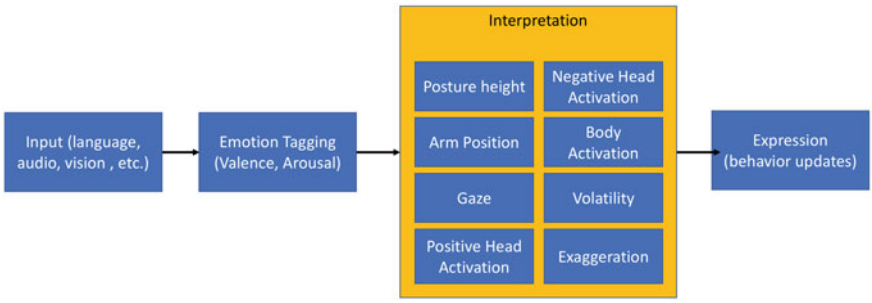


Fig. 5.19 Components of the emotional behavior generative system

parameter values was implemented to describe the emotional behavior that the robot "wants" to be at. For example, If Shimi is to be inherently happy, its homeostatic values would be representative of a happy behavior. If it is to be inherently sad then these values would be representative of a sad behavior. In our implementation we gave Shimi a homeostatic nature of calm and content.

The growth and decay function describes the rate at which the robot "grows" to a new behavior (defined by the control parameters) and "decays" back to its homeostatic state. This behavior is based on human behavior where it is not likely for people in a joyous and excited state to immediately change their mode to a calm and still state, unless triggered by some external force. The decay function allows Shimi to naturally make these transitions. The system supports multiple growth and decay functions in different situations. For example, Fig. 5.20a represents an immediate transition from the robot's current state to the new emotional state with an exponential decline back to the homeostatic state. This function can be used to represent abrupt physical action such as for expressing surprise or startled fear. Figure 5.20b can be used to express an emotion such as disappointment where transition into the emotional state is slower. The growth and decay function can be described by

$$f(P_i) = decay(H, S, T, t_i - t_0) \tag{5.6}$$

where P_i is the set of n control parameters at current time i such that $P_i = [p_{i0}, p_{i1}, \ldots, p_{in}]$, t_i is the current time, $t_0$is the initial time of activation, T is the total time necessary to decay back to the homeostatic state, H is the set of control parameters for the homeostatic state such that $H = [h_0, h_1, \ldots, h_n]$, and S is the set of received control parameters from the interpretation phase such that $S = [s_0, s_1, \ldots, s_n]$.

The total time, T, is determined by the particular situation. For example, the human expression of laughter can be thought of as a series of contiguous chuckle motions. The physical traits between a discrete segment of laughter and a chuckle are similar, but where laughter is repetitive and lengthy a chuckle is fleeting. The same notion

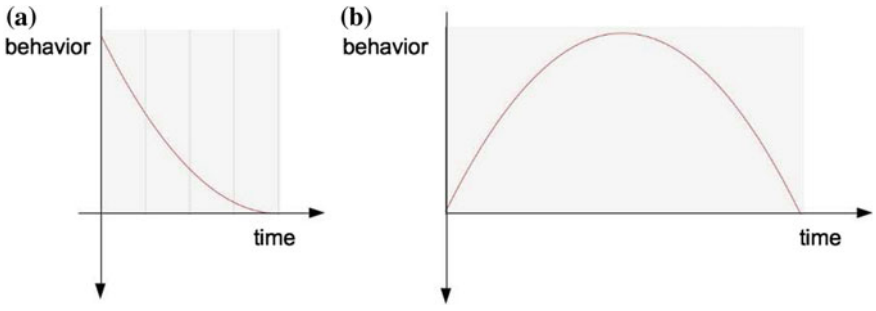


Fig. 5.20 Example decay rates where y = 0 is the homeostatic state and y = 1 is the new emotional state. **a** Shows an immediate transition and exponential decay, **b** shows a growth and decay

applies to the temporal length of growth and decay. In our current implementation we have a fixed T for different emotions. For future work an algorithmically determined T can be determined by a function of the relevant social display norms.

The expression system is driven by a timer which updates at a given interval, α where $\alpha = t_{i+1} - t_i$. The timer triggers an update of equation (1) every α seconds. This interval can be adjusted based on the robot update frequency. For smaller α values the transitions between emotional states will more closely follow the contours of the decay function. A larger α will ease computation expense on the expanse of transition smoothness.

Since Shimi is designed for musical applications we use α to represent the concept of a "beat", an essential musical parameters that signifies the perceptible pulse exhibited by rhythm. When Shimi dances to music it is aligns its motions with this pulse, by setting α to the temporal interval which signifies the length of a beat.

Shimi's movements are defined by a couple of basic motion primitives—head nod (up and down) and head shake (left and right). Figure 5.21 shows the range of motion for each of Shimi's DoFs. The Head Nod primitive actuates the head up/down and neck up/down motions, while Head Shake primitives actuates the head left/right motions. Motion primitives performance is defined by a subset of the control parameters such that:

$$P_{nod} = [p_{posture}, p_{gaze}, p_{posHeadActivation}, p_{volatility}, p_{exaggeration}] \tag{5.7}$$

$$P_{shake} = [p_{gaze}, p_{negHeadActivation}, p_{volatility}, p_{exaggeration}] \tag{5.8}$$

These parameters control the rate at which the primitive is performed, the maximum range the motion exhibits, and the center location of the motion.

In an effort to introduce variation into the repetitive actions, we added a volatility parameter. We control volatility by manipulating a value based on a Gaussian distribution about p_η where p_η is some control parameter. The Gaussian probability density function is the statistical distribution:

$$f(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{5.9}$$

where $\mu = mean$, and $\sigma^2 = variance$. An increase in the value $p_{volatility}$ results in an increase of variance. The volatility manipulation function is thus defined as:

$$volatility(p_\eta) = f(x; p_\eta, p_{volatility}) \tag{5.10}$$

where x is a random number. The rate, range, and location of the motion primitives are defined as functions of the remaining original control parameters and the volatility manipulated control parameters.

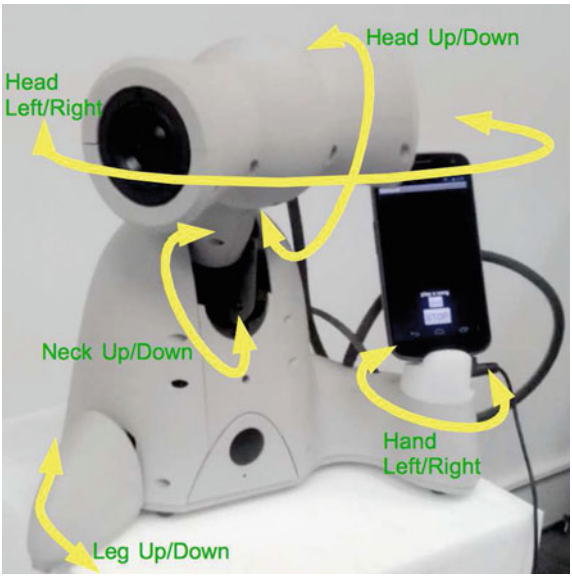


Fig. 5.21 Shimi's five degrees of freedoms. For our experiment we use the two head DoFs and the neck DoF

$$rate = g(volatility(p_{headActivation})) \tag{5.11}$$

$$range = g(volatility(p_{exaggeration})) \tag{5.12}$$

$$location_{nod} = g(p_{gaze}, p_{posture}) \tag{5.13}$$

$$location_{shake} = g(p_{gaze}) \tag{5.14}$$

where *rate* is always a whole number multiple of α in order to keep the motions synchronized to the beats. Figure 5.22 gives an overview of the architecture for the expression portion of the system.

5.4.2.3 Experiment Procedure

We conducted an experiment to measure the significance of our control parameters as emotional contributors. As part of the experiment, a generative emotional system was implemented on Shimi in order to determine the correlation between each parameter and participants' perceptions of emotions. For this purpose, ten participants (7 male) were asked to watch six 45 s performances of Shimi generating motions. Shimi had five emotional state updates for each performance using random values for the control parameters (values were randomly generated using Java's `Random.nextFloat()` function). The real time decay function output values for each parameter were recorded. The α update rate for the decay function was 200 ms, therefore, parameter values were recorded every 200 ms.

Participants were tasked with rating the level of a particular emotion in each performance by moving a slider up and down on a MIDI control board. Slider values

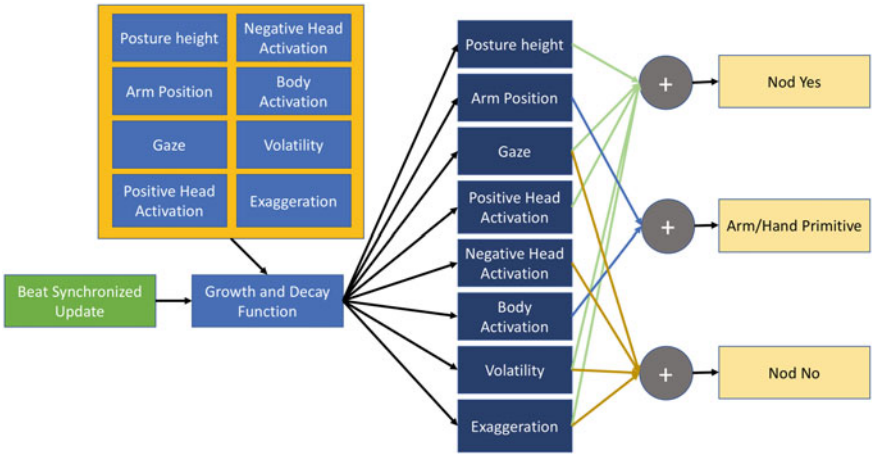


Fig. 5.22 The initial control parameters are received by the growth and decay function. This function allows for different types of transitions between emotional states (such as slow and smooth vs abrupt) and outputs new parameters as a function of time and the homeostatic state. The homeostatic state represents a set of constant parameters which the robot will always come back to as determined by the decay function. The function is run at intervals determined by an arbitrary timer in the beat synchronized update. This allows a dancing robot to align its movements with the perceived beat of music. Finally, motion primitives are performed in a manner determined by the outputs of the decay function

were recorded every 200 ms. Participants were told that a slider in its lowest position indicated the absence of the emotion. Higher values indicated emotions while level described how well the emotion was being represented. Before each performance participants were told which particular emotion they would be evaluating. Each participant moved the slider a total of six times rating each of the six fundamental emotions. Our random parameter generator assured that there were no identical performances. A message was sent over a network in order to synchronize the recording of Shimi's parameter values and the participant slider values.

5.4.2.4 Hypotheses

We made several hypotheses based on Darwin's observations and our empirical findings regarding dynamic affective expression design from the previous experiment:

H8 (Posture / Valence) Posture height will be positively correlated with positive valence emotions and negatively correlated with negative valence emotions.

H9 (Head Nodding / Arousal and Valence) Positive head activation (increased nodding rate) will be positively correlated with emotions of positive arousal and positive valence (happiness and surprise) and negatively correlated for the other emotions.

H10 (Head Shaking / Arousal and Valence) Negative head activation (increased shaking rate) will be positively correlated with emotions of positive arousal and negative valence (anger, fear, and disgust) and negatively correlated for the other emotions.

H11 (Gaze / Arousal and Valence) Gaze will be positively correlated with emotions of positive valence and positive arousal (happiness and surprise) and negatively correlated for other emotions.

H12 (Volatility / Emotions with "Purposeless Movements") Volatility will be positively correlated with emotions which Darwin describes as being expressed with "purposeless movements" (happiness and anger).

H13 (Exaggeration / Arousal) Exaggeration will be positively correlated with emotions which have a positive arousal level (anger, happiness, disgust, surprise, fear) and negatively correlated with emotions which have a negative arousal level (sadness).

5.4.2.5 Results and Discussion

The results were evaluated by calculating the correlation coefficients between the slider values and control parameters for each of the emotions (see Fig. 5.23). The p-values for the majority of the coefficients were below 0.05. The complete correlation values are shown in Table 5.2. The coefficients with $p > 0.05$ are shown in red.

The results largely supported our hypotheses, providing additional evidence that specific characteristics of body language are indicative of specific emotions. Specifically, posture height and gaze demonstrated direct correlations with an emotion's valence. Similarly to our first experiment, Fear, was and exception, showing weak positive correlations for both of these parameters despite its negative valence. Nodding and shaking of the head demonstrated direct correlations with an emotion's valence and arousal. Here too fear demonstrated a negative correlation with head shaking, contradicting to our hypothesis. Volatility, or the amount of variation exhibited in a behavior, showed positive correlations with anger and disgust and negative correlations with the other four emotions. Finally, the range of motion exhibited by a movement, defined as exaggeration, showed positive correlations with all of the emotions. These results can only offer broad affirmation concerning the relationship between the control parameters and the emotions. Additional studies and analyses can be helpful in describing the co-variance between multiple control parameters and the emotion ratings.

5.4.3 Shimi Interactive Applications

In addition to an underlying emotional intelligence, an embodied robotic musical companion affords several interactive applications. In this section we list several interactive applications implemented using the Shimi robot taking advantage of the

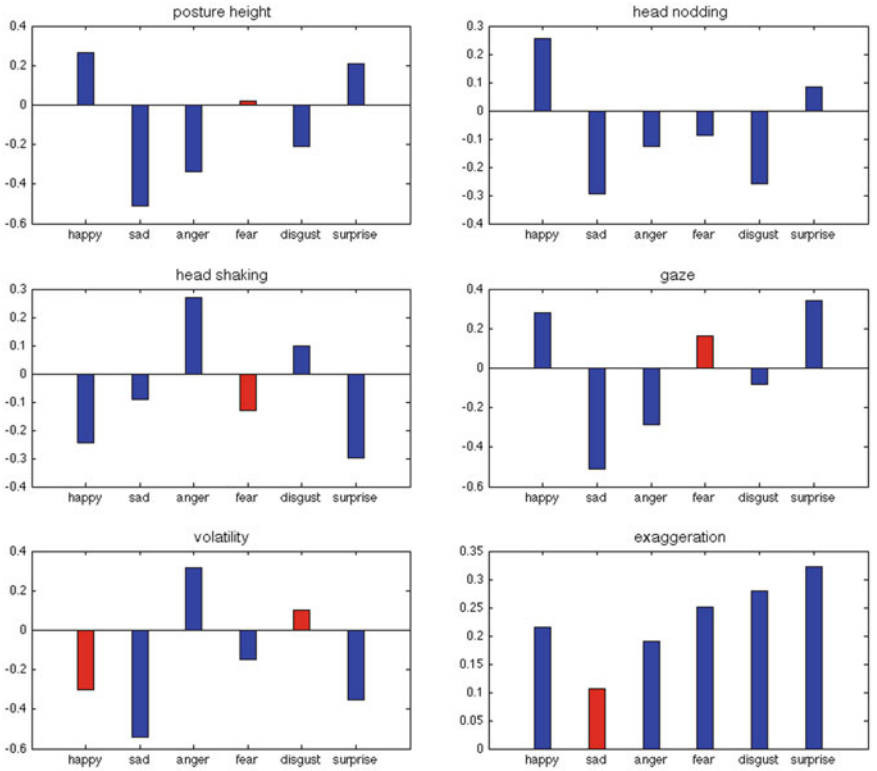


Fig. 5.23 Correlation coefficients for the parameter control variables and each of the six fundamental emotions. Blue bars indicate results supporting our hypotheses and red bars indicate results contrary to our hypotheses

Table 5.2 Parameter and emotion correlations: correlation values for each parameter where values in red indicate $p > 0.05$

Parameter	Happy	Sad	Anger	Fear	Disgust	Surprise
Posture height	0.26	−0.51	−0.34	0.02	−0.21	0.21
Head nodding	25	−0.30	−0.13	−0.09	−0.26	0.08
Head shaking	−0.25	−0.09	0.27	−0.13	0.1	−0.30
Gaze	0.28	−0.51	−0.29	0.16	−0.08	0.34
Volatility	−0.3	−0.54	0.32	−0.15	0.1	−0.35
Exaggeration	0.21	0.11	0.19	0.25	0.28	0.32

smart phone's microphone and camera and the built-in Android face and speech recognition service.

5.4.3.1 Shimi as an Interactive Speaker Dock

The built in speakers and smart phone docking station make Shimi ideal for serving as a speaker dock. However, rather than using a graphical user interface (GUI), the user can interact verbally and nonverbally with a physically expressive entity. In this mode, Shimi accesses the musical library on the device or uses online music sources such as Spotify. The Echo Nest (TEN) [61] provides metadata information about the music. We present three song query methods as initial proposals for how a robotic companion can enhance the speaker-dock experience. These methods are: query-by-natural-language, query-by-tapping, and query-by-dancing.

In 'query by natural language' (QbNL), natural langauge processing (NLP) is incorporated to support general music query and recommendation applications. Shimi responds to queries such as "do you have [a specific artist or song]?" or "can you play something [with a specific genre or mood]?" Perceptron models are trained with frequently used query phrases and word spotting is used to map the user's requested parameters to features made available by TEN. This includes specific artists, song titles, genres, moods, energy, and danceability. Essentially, Shimi supports a natural language interface for TEN's query methods.

Another method of interaction includes 'query by tapping' (QbT). This music retrieval system is similar to 'query by humming' (QbH) in which songs are chosen based off of their distinct melodies and how they compare to user input humming or singing [62]. In QbT users tap or clap a rhythmic motif and Shimi finds a song from its library which is best represented by that specific motif. Currently, our system can only function accurately on a much smaller scale (roughly 20–30 songs) than what has been successful with QbH. However, this method still holds potential for effective interaction. Interfaces such as mobile phones or keyboards require users to use their hands. A robotic interface such as Shimi frees users of the necessity of holding a device or pressing buttons, thus, permitting one's hands to be used for other purposes, like clapping or tapping a rhythm.

QbT works by matching rhythmic patterns to manually tagged rhythms which are considered representative of a piece. This is done using a kNN classifier based on a dynamic time warp (DTW) distance metric. At this stage, the method is a proof-of-concept as the classifier scales poorly. In addition not all songs can be successfully characterized by a single rhythmic motif. However, we believe the idea is worth pursuing as such functionality has proven popular through observations during showcases and demos.

QbT may be more successfully generalized by querying for classes of songs defined by genre and tempo rather than individual pieces. An example of such a broad classification approach has been implemented for Shimi in a system referred to as 'query by dancing' (QbD). QbD uses computer vision to track a user's movement based on Shi and Tomasi's 'Good Features to Track' corner detection algorithm [63].

Optical flow is computed for these good features and values for periodicity (via autocorrelation) and acceleration (mean absolute value of the first-order difference) are calculated. These values are mapped to two musical parameters available by TEN: (1) tempo and (2) energy. Using motion, users are then able to query for songs through TEN's search function using tempo and energy as parameters.

Identifying and playing the appropriate songs for these types of queries is essential in order for Shimi to exhibit a higher level musical intelligence. However, choosing the song is only the first step. Demonstrating an understanding of other low level qualities describing the music is addressed once the song begins to play.

Shimi's five DoFs provide the mobility necessary to enable dancing. Genre, beat, and section locations are features extracted from TEN that provide informational and musical cues describing how to dance. Shimi's generative motion functions enable beat synchronized movements. Shimi has a library of expressive dance moves which were specifically designed for different styles of music and can be performed at any tempo (though if a beat's duration is too short the move may not be fully completed). As a section change occurs (e.g., verse to chorus) Shimi switches dance moves as well.

Shimi also serves more practical applications beyond the entertainment that dancing provides. Two of the speakers are located on either side of Shimi's head. Using Google's face detection API it is possible to track the user's position. Given this position, Shimi moves it's head so that it remains "looking" in the direction of the user. This allows the listener to be optimally located within the stereo field even as he or she moves. This functionality is described in more detail in [64].

5.4.3.2 Shimi as a Listener

It is important for a robotic musical companion to exhibit intelligence regarding the music it is generating. It is also important for such a robot to similarly exhibit intelligence and responsiveness to music with which it is being presented. Additional capabilities were implemented to achieve this by enabling Shimi to listen and respond to a live performer (Fig. 5.24).

From a technical standpoint, this music analysis task proves difficult because analysis must occur in real-time using only the resources available to the phone. It is also difficult to distinguish important musical content from the noise generated from Shimi's own motors. Applying a low pass filter helps to reduce some of the high frequency content Shimi produces as it moves.

Determining which characteristics in the music Shimi should listen for and respond to provides another question. As a speaker dock, we found that when responding to the genre, beat location, and section location features of a song Shimi is able to demonstrate a knowledge of both higher and lower level musical parameters through dancing. In listening mode, however, we assume the live performer will take on a more improvisatory role and features describing the more immediate and subtle nuances of a performance will be necessary. We chose features to provide rough esti-

Fig. 5.24 Shimi listens to solo performances and responds with movements influenced by audio features including note density, pitch, loudness, and beat

mates of note density, pitch, loudness, and periodicity (i.e. is there a strong presence of a beat).

Analysis of the audio occurs over a three second window. Note density is calculated by counting the number of onsets per window where onsets are detected using a Q-bounded analysis based on the Pd/MSP `bonk~` object [65]. Fundamental frequency is calculated using a maximum likelihood estimation matching predefined windowed impulse trains to an FFT of the input signal. Though individual pitches can be useful for calculating several high level features, currently we are measuring how pitch changes over time enabling Shimi to respond to rising and falling melodic lines. Therefore, we take the first order difference of the detected fundamentals and calculate the average and standard deviation for each three second window. Average and standard deviations of the absolute magnitude spectrum is used to describe the loudness of the performance.

Auto- and cross-correlation methods described by Davies and Plumbley [66] are used for detecting the presence of periodicities in the performance (i.e. does the music have a beat). During solo improvisation performers often play more freely without adhering to a strict tempo. Empirically we have found that it is very noticeable when Shimi moves periodically to a beat which the listener does not also hear. To limit these false positives Shimi only aligns with a tempo if the measured beat confidence exceeds a certain threshold. Otherwise, Shimi's movements are influenced more by the other features (note density, pitch, and loudness).

5.4.3.3 Shimi as a Practice and Compositional Aid

Visual cues help to increase performance of synchronization tasks in music [67]. The manner in which something moves can be manipulated to provide additional benefits. For example, visual metronomes which exhibit acceleration and deceleration can be more helpful than those which move at a constant velocity [68]. The controller for Shimi's movements was designed to support such oscillating motions and other velocity contours. This allows for better perception of Shimi's movement-beat synchronization while dancing to a song and also means Shimi can be used as a visual metronome.

A drum sequencer with a natural language interface was built for Shimi. The user does not place the individual notes in the sequencer, but rather verbally asks Shimi to play drum patterns of particular styles such as, "Shimi, can you play a latin beat?" The user can then tap or clap the desired tempo. Shimi plays back the drum beat at the specified tempo and synchronizes it's dance gestures. This provides the user the benefits of both auditory and visual metronomic cues.

Using natural language as an interface has both advantages and disadvantages. It is an intuitive method of communication but it may not provide the desired low level control and might be frustrating if a request is interpreted incorrectly. Some functions are relatively simple to interpret through NLP such as those regarding tempo ("play it [in half time, in double time, 10 bpm slower, etc]"). However, other requests can have a degree of subjectivity which can be difficult to quantify, for example, "play something a little more funky."

We built default behavioral responses for many of these more subjective requests, but there remains a significant likelihood that the default response will not meet the user's wishes. To address this we use a stochastic sequencer allowing the user to request for increased or decreased probability that a particular drum is used at specific time steps. The sequencer starts with predefined probabilities based off a specific style template (i.e. rock, swing, samba, bossa-nova, etc.). Then a request can be made to modify the template in some way. For example, a request to "use more bass drum and less hi-hat" will result in a increased probability of a bass drum being hit and decreased probability of a hi-hat being hit. Because the sequencer is probabilistic Shimi's patterns will change over time. Shimi remembers the most recent 64 drum strikes so if the user hears something he or she is particularly fond of, then a request can made such as, "I like that pattern, repeat it."

5.4.3.4 Shimi as an Emotionally Intelligent Musical Linguist

Machines which demonstrate a sensitivity to people's emotional states by responding with understandable and relatable behaviors enable more natural, engaging, and comfortable human-robot interactions [23–25]. The benefits of emotionally intelligent robots have been exhibited in many interactive social situations [26]. This is because information such as compassion, awareness, and competency can be communicated through a non-conscious affective channel [21], resulting in lower cognitive load

for the user. Here, we describe a functionality which enables Shimi to detect sentiments and topics in language and generate an appropriate response using music and physical gestures.

Emotion can be synthesized by machines through facial expressions [52], physical movement [45], and speech synthesis [69]. Though emotionally expressive physical behaviors can be synthesized and interpreted with success, emotional speech synthesis has not been as successful. We propose using music as an alternative to speech synthesis. Though it may be possible to use generative music methods, in this first attempt at constructing such a system we use samples of prerecorded music. Such a method allows the system to leverage the many of songs which exist representing a plethora of topics and emotions. Music is an inherently descriptive art form with songs and lyrics capable of conveying complex emotions, topics, and stories. Using audio from the user's musical library as an alternative to speech synthesis allows Shimi to take advantage of music's expressive nature.

The steps for constructing such communicative functionality can be categorized into two parts (1) sentiment and topic detection in speech and (2) tagging portions of songs with particular emotions and topics. In order for Shimi to demonstrate an understanding of human emotion we trained neural networks to classify phrases with particular emotions and topics. Though sentiment and topic labeled corpi exist for NLP, our experience finds that these corpi are not optimal for the interactions we anticipate one to have with a robotic musical companion. Therefore, we developed a dataset specifically for our purposes using Twitter. Each spoken phrase is then classified by the perceptrons as belonging to one of the six fundamental emotions (happy, sad, surprise, fear, anger, and disgust) and 20 topics ranging from love and relationships to weather, money, and aging.

The second step requires the system to generate an appropriate response. Though automatic mood labeling for entire songs has been performed with marginal success, instantaneous emotion and topic labeling within a song has not been performed and no dataset exists for training. Therefore we hand-labeled a library of short audio samples to be used in the system. This allows us to use the quintessential musical representations (based on our own judgment) for each emotion and topic. We are currently working on automating this process, but using hand-labeled data enables us to experiment with the HRI component and determine which aspects of the music and lyrics are important for establishing useful communications.

The final interaction consists of the user speaking and Shimi responding by classifying the content and choosing an appropriate sample. For example, a user might say, "I'm so sad my girlfriend broke up with me" and Shimi will detect a sad sentiment and a topic of love. This will result with Shimi playing an audio clip of a sad love song.

In addition to using music to demonstrate empathy, music can also be used for Shimi to convey information about itself. For example, Shimi can reply to "yes" and "no" questions with music representing positive or negative feelings and play back music indicative of its affective state. When asked "how are you feeling," Shimi responds with a happy and positive song when its components are operational and it is satisfied with the functioning of its sensors. However, Shimi will have a frustrated

or angry response when something is wrong such as not being able to find the user's face with the camera. This may occur if the lighting conditions are not optimal or the person is simply out of view. These musical communicative functions help to socially engage the user and have practical applications in HRI.

5.4.3.5 Shimi as a Performer

Many of the interactions with Shimi were designed to explore the musical experience outside of traditional music performance settings. Here, we describe applications in which Shimi becomes a performer for an audience. Many of the functionalities previously described can be extended to work in a performance setting. In one performance Shimi plays the backing track as two vocalists sing and rap. Shimi dances along to the music while tracking the vocalists' movements. A combination of dance moves and emotionally expressive gestures are used to demonstrate excitement to build energy and engage the other performers and audience (Fig. 5.25).

Simply dancing to a backing track can be entertaining and considered a performance in itself. However, it is important for a robotic musician to address some of the interactive designs and functionalities inherent to other robotic musicians. This means implementing a methodology for Shimi to synthesize its own sounds and play an instrument. Shimi cannot play an acoustic instrument, but it has the ability to play samples or generate sounds using different synthesis methods.

Choosing the types of sound to generate is predominantly an aesthetic choice by the composer and programmer, however, the sounds and synthesis methods should

Fig. 5.25 Shimi performs with two vocalists by providing the backing track and tracking and responding to their movements

also be informed by how they can be connected to Shimi's movements. Up until now we have discussed Shimi's ability to provide visual cues in the context of periodic sounds and movements. Motion is generated in a beat synchronized fashion and the inherent musical pulse is visualized. In performance with an acoustic instrument visual cues are connected to each individual note or sound as it is played. There is no constraint to moving in a beat synchronized fashion. It is necessary for Shimi to manifest a similar sonic visual relationship with its generated sounds and movements in order to create the appearance of playing an instrument.

We built an architecture which enables Shimi to imitate two different methods in which sound is produced with acoustic instruments. The first method simulates the relationship between sound and gesture inherent to striking a percussive instrument. In this situation the sound is preceded by the motion of moving towards the resonator. Sound is produced once the motion has reached the surface of the instrument followed by an additional rebounding or bounce motion moving away from the surface. Shimi emulates this by choosing a target position for one or more degrees of freedom (DoFs) and accelerates to this position playing a sound only once it's reached the destination. Different sounds or notes can be mapped to different gestures and DoFs to provide consistency reinforcing the perception that Shimi is playing an instrument.

The second method simulates the act of producing sound in a similar style to bowing a violin or cello. In this scenario the sound is linked to continuous motion. When using samples playback speeds are adjusted to fit the duration of a particular motion. For synthesized sounds specific parameters controlling characteristics such as the pitch, volume, or timbre of the sound are mapped to DoF positions. Such a mapping allows the sound to change with the velocity contours of the movements creating a stronger contingency between motion and sound.

Coupling Shimi's motion controllers and sound generators produces useful visual cues. However, by doing this Shimi becomes constrained by its mechanics in the same way acoustic instrument playing robotic musicians are. Additionally, maintaining a motion and sonic link and preventing confusion for the listener restricts the composer from using too many types of sounds. One way to overcome this is to use multiple Shimi robots in which each robot has it's own set of sounds that contribute to the overall sound and structure of the music. A performance using three Shimi robots was written. In this scenario the robots communicate with each other (using the phone's networking capabilities) to synchronize their movements and sounds to create a choreographed performance. The piece uses a predefined set of samples and synthesizer parameters. It functions as a stand alone robotic performance without any human interaction. However, a mode allowing users to record their own sounds to be used in the piece was developed permitting additional human input and providing variety to the music.

5.5 Conclusion

We showed how embodiment can support synchronization, player coordination, and audience engagement. An embodied robotic musical companion can also express emotions and play a role in a range of applications. This survey suggests that robotics have a place in the many scenarios in which music is experienced. Though several challenges must be addressed and additional research is necessary, novel and entertaining forms of music consumption can arise as a result of designing embodied robotic musicians and robotic musical companions inspired by underlying musical intelligence and ambition.

References

1. Kidd, Cory D., and Cynthia Breazeal. 2004. Effect of a robot on user perceptions. In *Proceedings of 2004 IEEE/RSJ international conference on intelligent robots and systems, 2004(IROS 2004)*, vol. 4, 3559–3564. IEEE.
2. Bainbridge, Wilma A., Justin Hart, Elizabeth S. Kim, and Brian Scassellati. 2008. The effect of presence on human-robot interaction. In *The 17th IEEE international symposium on robot and human interactive communication, 2008. RO-MAN 2008*, 701–706. IEEE.
3. Gil, Weinberg, and Driscoll Scott. 2006. Toward robotic musicianship. *Computer Music Journal* 30 (4): 28–45.
4. Weinberg, Gil, Beck Andrew, and Godfrey Mark. 2009. Zoozbeat: A gesture-based mobile music studio
5. Gil, Weinberg. 2005. Interconnected musical networks: Toward a theoretical framework. *Computer Music Journal* 29 (2): 23–39.
6. Weinberg, Gil. 1999. Expressive digital musical instruments for children. PhD thesis, Massachusetts Institute of Technology.
7. Weinberg, Gil, Scott Driscoll, and Travis Thatcher. 2006. Jam'aa-a middle eastern percussion ensemble for human and robotic players. In *International computer music conference*, 464–467.
8. Luck, Geoff, and John A. Sloboda. 2009. Spatio-temporal cues for visually mediated synchronization. *Music Perception: An Interdisciplinary Journal* 26 (5): 465–473.
9. Repp, Bruno H., and Amandine Penel. 2004. Rhythmic movement is attracted more strongly to auditory than to visual rhythms. *Psychological Research* 68 (4): 252–270.
10. Dirk-Jan, Povel, and Essens Peter. 1985. Perception of temporal patterns. *Music Perception: An Interdisciplinary Journal* 2 (4): 411–440.
11. Müller, Meinard. 2007. Dynamic time warping. *Information Retrieval for Music and Motion* 69–84.
12. Komatsu, Tomoaki, and Yoshihiro Miyake. 2004. Temporal development of dual timing mechanism in synchronization tapping task. In *RO-MAN 2004. 13th IEEE international workshop on robot and human interactive communication (IEEE Catalog No. 04TH8759)*, 181–186. IEEE.
13. Crick, Christopher, Matthew Munz, and Brian Scassellati. 2006. Synchronization in social tasks: Robotic drumming. In *ROMAN 2006-The 15th IEEE international symposium on robot and human interactive communication*, 97–102. IEEE.
14. Inderbitzin, Martin, Aleksander Väljamäe, José Maria Blanco Calvo, Paul F. M. J. Verschure, and Ulysses Bernardet. Expression of emotional states during locomotion based on canonical parameters. In *Ninth IEEE international conference on automatic face and gesture recognition (FG 2011)*, Santa Barbara, CA, USA, 21–25 March 2011, 809–814. IEEE.
15. Hillel, Aviezer, Trope Yaacov, and Todorov Alexander. 2012. Body cues, not facial expressions, discriminate between intense positive and negative emotions. *Science* 338 (6111): 1225–1229.

16. de Gelder, Beatrice. 2006. Towards the neurobiology of emotional body language. *Nature Reviews Neuroscience* 7: 242–249.
17. Nele, Dael Marcello Mortillaro, and R. Scherer Klaus. 2012. The body action and posture coding system (bap): Development and reliability. *Journal of Nonverbal Behavior* 36: 97–121.
18. Mark, Coulson. 2004. Attributing emotion to static body postures: Recognition accuracy, confusions, and viewpoint dependence. *Journal of Nonverbal Behavior* 28 (2): 117–139.
19. Krauss, Robert M., Palmer Morrel-Samuels, and Christina Colasante. 1991. Do conversational hand gestures communicate? *Journal of Personality and Social Psychology* 61 (5): 743.
20. Kipp, Michael, and J.-C. Martin. 2009. Gesture and emotion: Can basic gestural form features discriminate emotions? In *3rd international conference on affective computing and intelligent interaction and workshops, 2009. ACII 2009*, 1–8. IEEE.
21. Picard, Rosalind W. 1995. Affective computing.
22. Frijda, N.H. 1987. *The emotions*. London: Cambridge University Press.
23. Kozima, Hideki, and Hiroyuki Yano. 2001. In search of otogenetic prerequisites for embodied social intelligence. In *Proceedings of the workshop on emergence and development on embodied cognition; international conference on cognitive science*, 30–34.
24. Cynthia, Breazeal, and Aryananda Lijin. 2002. Recognition of affective communicative intent in robot-directed speech. *Autonomous Robots* 12 (1): 83–104.
25. Castellano, Ginevra, Iolanda Leite, André Pereira, Carlos Martinho, Ana Paiva, and Peter W. McOwan. 2010. Affect recognition for interactive companions: challenges and design in real world scenarios. *Journal on Multimodal User Interfaces* 3 (1): 89–98.
26. Scheutz, Matthias, Paul Schermerhorn, and James Kramer. 2006. The utility of affect expression in natural language interactions in joint human-robot tasks. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on human-robot interaction*, 226–233. ACM.
27. Laurence, Devillers, Vidrascu Laurence, and Lamel Lori. 2005. 2005 special issue: Challenges in real-life emotion annotation and machine learning based detection. *Neural Networks* 18 (4): 407–422.
28. Albert, Mehrabian. 1996. Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in temperament. *Current Psychology* 14 (4): 261–292.
29. Russell, James A. 2009. Emotion, core affect, and psychological construction. *Cognition and Emotion* 23 (7): 1259–1283.
30. Lindquist, Kristen A., Tor D. Wager, Hedy Kober, Eliza Bliss-Moreau, and Lisa Feldman Barrett. 2012. The brain basis of emotion: A meta-analytic review. *Behavioral and Brain Sciences* 35 (03): 121–143.
31. Katherine, Vytal, and Hamann Stephan. 2010. Neuroimaging support for discrete neural correlates of basic emotions: A voxel-based meta-analysis. *Journal of Cognitive Neuroscience* 22 (12): 2864–2885.
32. Stephan, Hamann. 2012. Mapping discrete and dimensional emotions onto the brain: controversies and consensus. *Trends in Cognitive Sciences*.
33. Giovanna, Colombetti. 2009. From affect programs to dynamical discrete emotions. *Philosophical Psychology* 22 (4): 407–425.
34. Barrett, Lisa Feldman, Maria Gendron, and Yang-Ming Huang. 2009. Do discrete emotions exist? *Philosophical Psychology* 22 (4): 427–437.
35. John, Lasseter. 1987. Principles of traditional animation applied to 3D computer animation. *Computer Graphics* 21 (4): 35–44.
36. Gielniak, Michael J., and Andrea L. Thomaz. 2011. Anticipation in robot motion.
37. Cauell, Justine, Tim Bickmore, Lee Campbell, and Hannes Vilhjdlmsson. 2000. Designing embodied conversational agents. *Embodied Conversational Agents* 29.
38. Nayak, Vishal, and Matthew Turk. 2005. Emotional expression in virtual agents through body language. *Advances in Visual Computing* 313–320.
39. Salem, Maha, Stefan Kopp, Ipke Wachsmuth, and Frank Joublin. 2010. Generating robot gesture using a virtual agent framework. In *2010 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 3592–3597. IEEE.

40. Riek, Laurel D., T.-C. Rabinowitch, Paul Bremner, Anthony G. Pipe, Mike Fraser, and Peter Robinson. 2010. Cooperative gestures: Effective signaling for humanoid robots. In *2010 5th ACM/IEEE international conference on human-robot interaction (HRI)*, 61–68. IEEE.
41. Moon, A.J., Chris A.C. Parker, Elizabeth A. Croft, and H.F. Van der Loos., 2013. Design and impact of hesitation gestures during human-robot resource conflicts. *Journal of Human-Robot Interaction* 2 (3): 18–40.
42. Maha, Salem, Kopp Stefan, Wachsmuth Ipke, Rohlfing Katharina, and Joublin Frank. 2012. Generation and evaluation of communicative robot gesture. *International Journal of Social Robotics* 4 (2): 201–217.
43. Breazeal, Cynthia, Andrew Wang, and Rosalind Picard. 2007. Experiments with a robotic computer: body, affect and cognition interactions. In *2007 2nd ACM/IEEE international conference on human-robot interaction (HRI)*, 153–160. IEEE.
44. Hoffman, Guy, and Cynthia Breazeal. 2008. Anticipatory perceptual simulation for human-robot joint practice: Theory and application study. In *Proceedings of the 23rd national conference on artificial intelligence—Volume 3*, AAAI'08, 1357–1362. AAAI Press.
45. Michalowski, Marek P., Selma Sabanovic, and Hideki Kozima. 2007. A dancing robot for rhythmic social interaction. In *2007 2nd ACM/IEEE international conference on human-robot interaction (HRI)*, 89–96. IEEE.
46. Monceaux, Jérôme, Joffrey Becker, Céline Boudier, and Alexandre Mazel. 2009. Demonstration: First steps in emotional expression of the humanoid robot nao. In *Proceedings of the 2009 international conference on multimodal interfaces*, 235–236. ACM.
47. Grunberg, David K., Alyssa M. Batula, Erik M. Schmidt, and Youngmoo E. Kim. 2012. Synthetic emotions for humanoids: Perceptual effects of size and number of robot platforms. *International Journal of Synthetic Emotions (IJSE)* 3 (2): 68–83.
48. Kidd, Cory David. 2003. Sociable robots: The role of presence and task in human-robot interaction. PhD thesis, Massachusetts Institute of Technology.
49. Walters, Michael L., Kerstin Dautenhahn, René Te Boekhorst, Kheng Lee Koay, Dag Sverre Syrdal, and Chrystopher L. Nehaniv. 2009. An empirical framework for human-robot proxemics. *Procs of New Frontiers in Human-Robot Interaction*.
50. Takayama, Leila, and Caroline Pantofaru. 2009. Influences on proxemic behaviors in human-robot interaction. In *IEEE/RSJ international conference on intelligent robots and systems, 2009. IROS 2009*, 5495–5502. IEEE.
51. Mead, Ross, Amin Atrash, and Maja J. Mataric. 2011. Recognition of spatial dynamics for predicting social interaction. In *Proceedings of the 6th international conference on human-robot interaction*, 201–202. ACM.
52. Breazeal, C. 2003. Emotion and sociable humanoid robots. *International Journal of Human-Computer Studies* 15: 119–155.
53. Velásquez, Juan D. 1997. Modeling emotions and other motivations in synthetic agents. In *Proceedings of the national conference on artificial intelligence*, 10–15. Citeseer.
54. Xia, Guangyu, Roger Dannenberg, Junyun Tay, and Manuela Veloso. Autonomous robot dancing driven by beats and emotions of music. In *Proceedings of the 11th international conference on autonomous agents and multiagent systems—Volume 1*, AAMAS '12, 205–212; Richland, S.C. 2012. International foundation for autonomous agents and multiagent systems.
55. Traue, Harald C., Frank Ohl, André Brechmann, Friedhelm Schwenker, Henrik Kessler, Kerstin Limbrecht, Holger Hoffmann, Stefan Scherer, Michael Kotzyba, Andreas Scheck, et al. 2013. A framework for emotions and dispositions in man-companion interaction. *Coverbal Synchrony in Human-Machine Interaction*, 99.
56. Frijda, N.H. 1995. Emotions in robots. *Comparative approaches to cognitive science*, ed. by H.L. Roitblat and J.-A. Meyer, 501–516.
57. Rolls, E.T. 2005. *Emotion explained*. USA: Oxford University Press.
58. Walbott, H.G. 1998. Bodily expression of emotion. *European Journal of Social Psychology* 28 (6): 879–896.
59. Darwin, Charles. 1916. *The expression of the emotions in man and animals*. D. Appleton and Co., New York. http://www.biodiversitylibrary.org/bibliography/4820.

60. Sullivan, Jean, Linda A. Camras, and Michel George. 1993. Do infants express discrete emotions? Adult judgments of facial, vocal, and body actions. *Journal of Nonverbal Behavior* 17: 171–186.
61. The echo nest. http://echonest.com/, 2014.
62. Ghias, Asif, Jonathan Logan, David Chamberlin, and Brian C. Smith. 1995. Query by humming: Musical information retrieval in an audio database. In *Proceedings of the third ACM international conference on multimedia*, 231–236. ACM.
63. Shi, Jianbo, and Carlo Tomasi. 1994. Good features to track. In *1994 IEEE computer society conference on computer vision and pattern recognition, 1994. Proceedings CVPR'94*, 593–600. IEEE.
64. Hoffman, Guy. 2012. Dumb robots, smart phones: A case study of music listening companionship. In *2012 IEEE, RO-MAN*, 358–363. IEEE.
65. Puckette, Miller S., Miller S. Puckette Ucsd, Theodore Apel, et al. 1998. Real-time audio analysis tools for Pd and MSP.
66. Davies, Matthew E.P., and Mark D. Plumbley. 2004. Causal tempo tracking of audio. In *ISMIR*.
67. Sun, Sisi, Trishul Mallikarjuna, and Gil Weinberg. Effect of visual cues in synchronization of rhythmic patterns.
68. Albin, Aaron, S. Lee, and Parag Chordia. 2011. Visual anticipation aids in synchronization tasks. In *Proceedings of the 2007 society for music perception and cognition (SMPC)*.
69. Burkhardt, Felix. 2005. Emofilt: The simulation of emotional speech by prosody-transformation. *INTERSPEECH* 509–512.

Chapter 6
"Watch and Learn"—Computer Vision for Musical Gesture Analysis

6.1 Abstract

In the previous chapter we showed how human musicians can benefit from visual and physical cues that are afforded by robotic musicians. Similarly, robotic musicians can benefit by augmenting their own abilities through analyzing visual cues by humans. Like humans, robotic musicians can use vision to anticipate, coordinate and synchronize their music playing with human collaborators. For example, a robotic drummer could maintain visual connection with a human guitar player to anticipate and synchronize the ending of a song; it could detect a turn-taking gesture by a human, signifying it is time for it to improvise; or it could reinforce its tempo and beat detection by analyzing the bobbing head of a human. Therefore, one of our principal guidelines calls for robotic musicians to not only listen to musical input, but also to use visual input to inform their musical decisions. In this chapter we describe a few projects developed for Shimon and Shimi which utilize visual information to assist with synchronization, song selection, and composition.

6.2 Robotic Musical Anticipation Based on Visual Cues

6.2.1 Introduction

Visual cues-based anticipation plays an important role in human-to-human interaction, and in particular, in time-demanding scenarios such as group music playing. One example that demonstrates the importance of visual cues in human-to-human interaction is an orchestra that prepares to begin playing by anticipating the end of the introductory gesture by the conductor. Similarly, band members often watch the head movement of a lead singer, which can indicate whether to switch to a new section or to end a song. In a jazz quartet performance, exaggerated gestures can be used to cue the beginning and ending of a section. Anticipation can also be used in

G. Weinberg et al., *Robotic Musicianship*, Automation, Collaboration, & E-Services 8, https://doi.org/10.1007/978-3-030-38930-7_6

a more subtle manner to predict stylistic features. In jazz improvisation, for example, accompanying musicians can complement and ornament the lead improviser by reacting to their body gestures. Collaborating musicians can anticipate the volume and intensity of an improvising marimba player by noticing the height of a mallet before it strikes. Likewise, a pianist's torso and head movements can vary depending on the intensity of their playing. Many instruments are physically structured so that pitch gradually increases from left to right, or top to bottom. Anticipating a specific note or a pitch range can be achieved by observing the location of a musician's hands or mallets. Improvising musicians might not be consciously aware of their movements, but these natural gestures are essential for the collaborating musicians as they anticipate stylistic changes and develop appropriate responses. Informed by these observations, we were interested in studying the role of anticipatory visual gestures in music interaction between humans and robotic musicians. In particular, we focused on a case study where a robotic marimba player responds to a human playing a percussion instrument using computer vision to anticipate human gestures.

6.2.2 Related Work

Recent works have demonstrated the importance of visual cues-based anticipation for human-robot interaction in a variety of contexts, including musical interaction and musician-robot interaction. For example, it has been shown that for robots, the ability to anticipate the behavior of other robots can be advantageous in tasks such as shifting places with one another while navigating through an area with or without obstacles [1]. Eyssel et al. showed that gesture-based anticipation in human-robot interaction increases anthropomorphic inferences and acceptance of the robots by humans [2]. Gielniak and Thomaz observed that anticipatory variants allow humans to discern motion intent sooner than motions that do not have anticipatory cures and that humans are able to reliably predict motion intent before a symbol frame [3]. Hoffman conducted an experiment in human-robot interaction, showing that anticipation can be helpful in a call-and-response scenarios [4]. In [5], a visual-based anticipation system is incorporated in a robot table tennis player. The system enables the robot to react earlier while the opponent is performing the striking movement. It is shown that the learned reaction policies can significantly improve the performance of the robot player.

(a) Shimon setup for drums (b) Shimon setup for Marimba

Fig. 6.1 Two setups for the Shimon robot used in the experiment (a) Drumming setup (b) Marimba-playing setup

6.2.3 Motivation and Approach

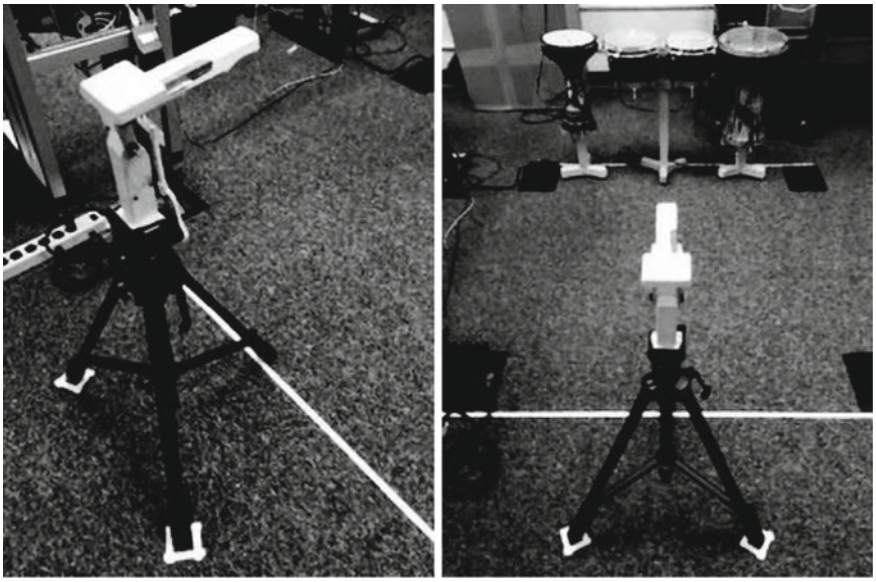
In this work we aim to provide Shimon, the robotic marimba player, with anticipatory capabilities in an effort to improve its synchronisation with humans. We hypothesize the human interaction with Shimon will become more natural and engaging for both human co-players and for the audience if the robot embodies features inherent to human-human musical interactions such as gesture-based anticipation.

We explore a particular case in which a percussionist is playing a four-piece percussion set and Shimon is playing either the marimba or a three-piece percussion set (Fig. 6.1). Anticipation in this project is based on computer vision: the two-dimensional position of the mallets used by the human percussionist is tracked using infrared light-emitting diodes (LEDs) and a camera (Figs. 6.2 and 6.3). The movement is anticipated using uniformly accelerated trajectory prediction.

To address our motivation, we developed two anticipatory algorithms. The first can predict the strike ~10 ms before it occurs and the second ~100 ms before occurrence. Using the second algorithm, we conducted a user study that shows that on average, synchronization was performed better by the anticipating robot than by humans. We also show that providing time in advance of action helps humans synchronize the strike with a reference player. However, the benefit of effect decreases after a certain point.

6.2.4 Method

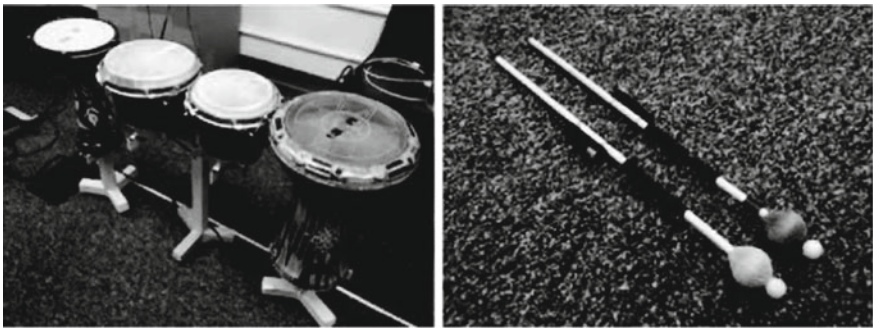
We studied anticipation in the context of a percussionist playing a set of four drums using mallets, as shown in Fig. 6.1. For computer vision, we used the Nintendo Wii Remote controller to track the bi-dimensional coordinates of the mallets on the scene. Infrared LEDs powered by 1.5-V batteries were attached to the tip of each mallet.



(a) the Nintendo Wii Remote controller is used as an infrared camera

(b) the camera points perpendicularly to the line of drums

Fig. 6.2 Sensor configurations



(a) A closeup of the four drums

(b) A closeup of mallets, showing the infrared LEDs attached to the tips and the batteries attached tothe sticks

Fig. 6.3 Drums and mallets setup

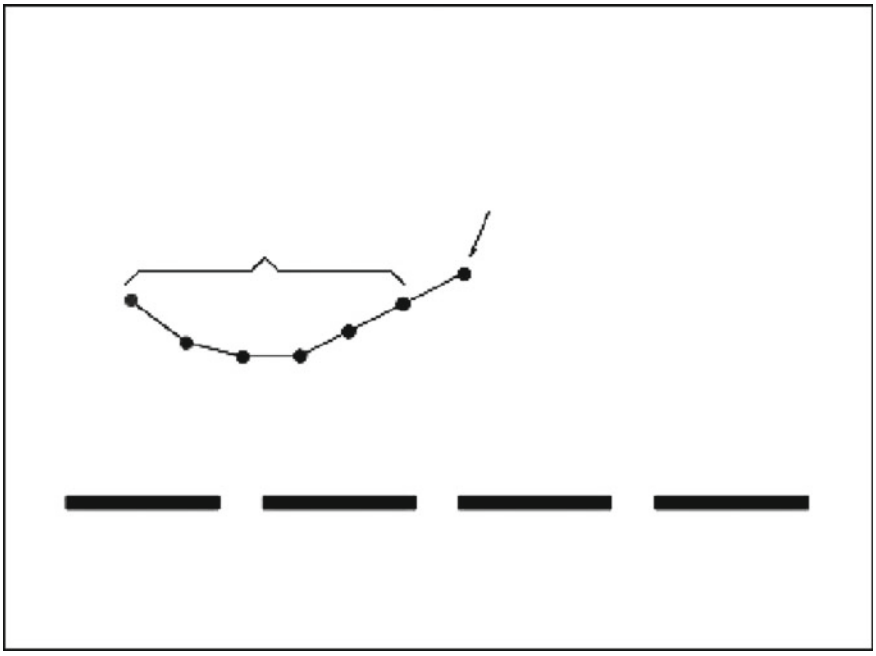


Fig. 6.4 The scene as viewed by the infrared sensor

Small opaque spheres were placed around the LEDs so that the emitted light spread in a cone (LEDs are usually nearly unidirectional). The Wii Remote can track up to four infrared lights at a sample of roughly 100 frames per second.

Figure 6.4 shows the scene as viewed by the infrared sensor, focusing on the mallet position. As the main gesture performed by the human was the movement of their arm from the moment they start lifting the mallet to the moment that sound is produced, we were interested in detecting when such a gesture starts so that we can predict which key should be hit and at what time in the future the sound should be produced. We tested two different approaches. First, anticipation measurement was conducted when the mallet started the descent toward the drum. Second, anticipation was done for the mallet upward movement, right before the descent prior to the strike.

Because the mallet must descend toward the drum at a certain minimum speed to generate the sound, the time of anticipation achieved with the first approach was not sufficient to circumvent the delay between the instruction to play and the robotic playing action itself. We concluded that human percussionists tend to strike too quickly for the first approach to be effective. Nonetheless, we decided to keep the description of the approach in this text, as it might be of interest to the reader, depending on the hardware properties of their application.

Shimon's arms are usually controlled by receiving a MIDI note number and velocity. Shimon's action starts after a fixed 500-ms delay from the time the message is

received. For this experiments, however, we kept the arms fixed and dealt only with the delay related to triggering Shimon solenoids, which is ~90 ms.

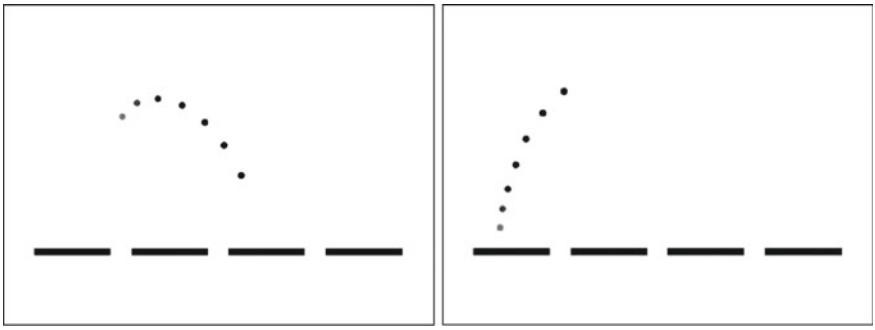
The protocol for the first executed experiment consisted of putting piezoelectric sensors in one of the drums shown in Fig. 6.3 as well as in a test drum under one of Shimon's solenoids. The time difference between the strike of the player and the strike of the robot was measured by observing the response of the piezo sensors. Two different cases were observed: first, when the strike of the robot was triggered by the anticipation algorithm (the one that observes the upward movement of the mallet) and second, when the strike was triggered by the piezoelectric sensor installed in the player's drum.

A similar setting was also used to test the anticipation ability of human players. For this purpose piezo sensors were installed on two different drums (one for the reference player and the other for the test player). This setting was further used to evaluate how significant it was for a human player to see a short or long gesture before the strike of another percussionist when anticipating the strike.

6.2.5 Algorithm

Given the features of the instruments being played, we assumed that the trajectory of each mallet would be nearly parabolic, described by a polynomial of the second degree, when moving from one drum to another. We found that the same anticipation technique could be applied when the same drum was struck consecutively. Figure 6.6 shows a few different samples of mallet trajectories (Fig. 6.5).

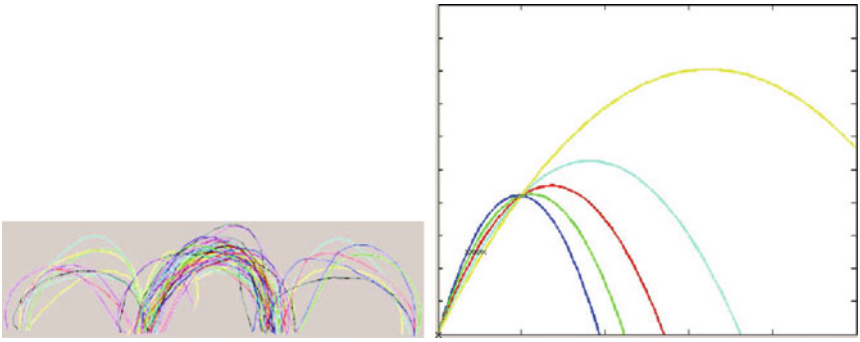
In the first implementation, we anticipated the strike position and time when the mallet starts the downward trajectory, as shown in Fig. 6.5. We kept track of the



(a) The depiction of different sets of points used for anticipating the strike at the downward movement

(b) The depiction of different sets of points used for anticipating the strike at the upward movement before the strike

Fig. 6.5 Anticipation curves for downward and upward mallet movements



(a) The samples of mallettrajectory curves from a drum to its closest neighbor

(b) The computation of the point of strike from three points sampled in the upward movement is not robust, as small changes in one of the points lead to big differences in the resulting position

Fig. 6.6 Mallet trajectories

vertical differences between consecutive capture points. Therefore, in this case, we detected that a strike was about to happen when a certain number of consecutive negative vertical differences occurred and the magnitude of the sequence of vertical differences increased. We observed when the velocity was negative (downward movement) and the mallet was accelerating using only vertical difference information. Therefore, we did not use time measurements that were required for velocity and acceleration, avoiding the extra noise that occurs when adding time information. Once we detected that a strike is about to happen, we fitted a parabola was to the sequence of points and calculated the intersection of the proper branch of the parabola with the straight line at the level of the drums. This provided us with a prediction of the actually drum to strike. We used the OpenCV linear algebra functions to solve the linear system that led to the parameters of the equation we were interested in [6].

To predict the impact time, a uniformly accelerated movement was supposed, and the time of strike was calculated based on the acceleration and velocity computed using the observed curve. We recall that the displacement and velocity equations for the uniformly accelerated movement are

$$S_f = S_i + v_i t + at^2 \tag{6.1}$$

and

$$v_f = v_i + a \tag{6.2}$$

respectively, where s denotes position, v denotes velocity, t denotes time interval, and a denotes acceleration. The subscripts f and i stand for final and initial, respectively.

In the second case study, we searched for an upward movement, i.e., a sequence of consecutive positive velocities. A tail of at least three sampled points was observed, although more were used when the sequence of points was contained in the upward section of a strike. If the last point of the sequence passed a vertical threshold (set according to the average high a mallet reaches before a strike), a strike was detected and defined.

Due to measurement noise and variability in player movements, fitting a parabola for anticipating where the mallet stroke did not provide reliable results. When a strike was detected, we computed the inclination of the line connecting the first and last points of the observed sequence to infer which drum would be struck. This depended on the distance between the drums and the movements of the player.

Similarly to the previous case, a uniformly accelerated movement was supposed for time prediction. However, here the acceleration was set beforehand and treated as gravity, while the mallet was regarded as a body in free fall. Vertical velocity was computed between the time and position measurements of the first and last points in the tail of observed points. In the first setting, an anticipation of ~10 ms was obtained, while for the second setting ~100 ms was used supposing an acceleration of −250 SI for mallets coordinates ranging from zero to one.

Although we developed two algorithms with differing prediction times, we found that the first case (predicting a strike 10 ms beforehand) was not useful for our application due to the 90-ms delay between the moment Shimon's solenoid sent a message to strike and the actual striking time. Therefore, we did not test this algorithm in our user study but instead used only the one that used a 100-ms prediction time. To calculate the delay, we used piezoelectric sensors attached to the drum heads to measure the onsets of a strike. We measured the time difference based on the computer's system clock, using Puckette's onset detection object *bonk* ~ [7], which uses a Q-bounded analysis of the incoming sound to detect attacks.

6.2.6 Results and Discussion

In our first experiment, we measured the time difference between the human player's strokes and Shimon's strokes in two different cases: (1) when the robot strike was triggered by the anticipation algorithm and (2) when the strike was triggered by a piezoelectric sensor installed in the player's drum. In addition, we repeated the part of this experiment that tested anticipation with human players replacing the robot. All of the human participants were Georgia Tech students over 18 years of age. They were all musicians enrolled in the music technology masters program. Musical performance experience ranged from three to 15 years on various musical instruments.

For each participant, ten comparisons were made, with the reference drummer striking with an upward plus downward mallet movements. The results are shown in Table 6.1, with the mean and standard deviation in milliseconds. For ten tries, the robot tried to anticipate the strike of the reference player and to play in

Table 6.1 Statistical measurements of time differences between strikes of a human player and the robot (first two lines), and another human player (remaining lines)

Experiment	Mean	Standard deviation
Shimon (anticipation)	19.4	16.2
Shimon (piezo)	79	4.8
User A	18.9	15.7
User B	34.2	26.6
User C	41.1	33.5
User D	51.6	26
User E	35.1	16.5
User F	28.7	21.9
User G	47.8	34.4

synchronization. On average, the time difference between strikes was 19.4 ms, with a standard deviation of 16.2 ms. As can be seen, on average, the robotic anticipation algorithm outperformed the humans in synchronizing its strike with the human.

We also evaluated the relationship between synchronization level and the advance time length provided to users, as can be see in Fig. 6.7. The chart depicts the result of an experiment where subjects were asked to strike the drum in 3 different manners: (1) use only downward movement, (2) use upward and downward movements and (3) produce an anticipatory gesture followed by upward and downward movements (the gesture consisted of a left-right movement). The results show that the synchronization did not improve significantly when including an anticipatory gesture previous to the upward movement. However, using an upward movement before the strike was significantly more helpful for producing better synchronization.

It is worth mentioning that all time measurements for the experiments were performed by comparing the responses of two different piezo sensors, detecting the impact of the strike. Therefore, the delay related to computer vision apparatus was included. Observing that a strike was about to happen involved detecting an upward or downward movement, which was negligible in computational terms. Likewise, the computation that led to the parameters describing the striking time and the drum to be hit were negligible.

6.2.7 Conclusions

In this project, we explored the effect of anticipatory gestures for synchronization between humans and robotic musicians using computer vision. Two different algorithms for anticipation were developed, obtaining predictions of ~10 and 100 ms before the strike event. A user study was conducted showing that, on average, synchronization was performed better by the robot using the anticipatory cue than by

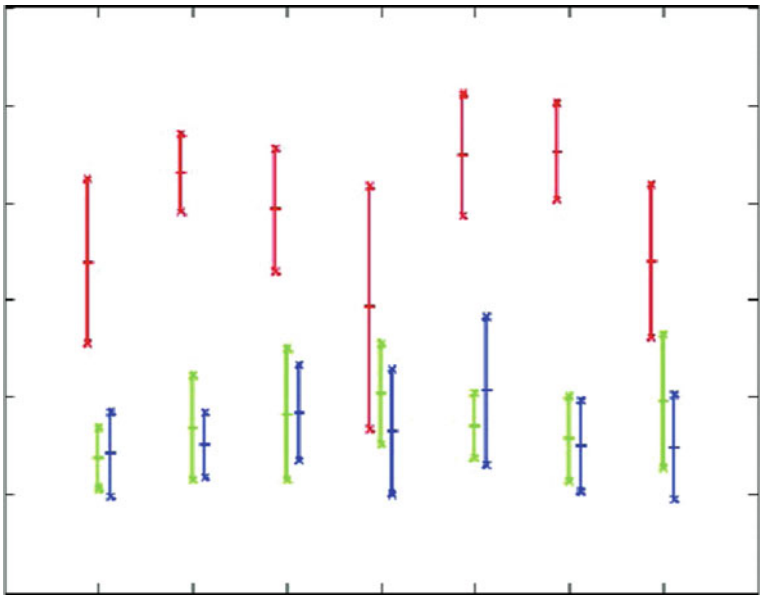


Fig. 6.7 The means (centers of vertical lines) and standard deviations (half lengths of vertical lines), in milliseconds, of the differences between the times that the reference player and the tested user strike for seven users (x-dimension) for three degrees of difficulty: only downward movement (red lines), upward and downward movements (green lines), and gesture plus upward and downward movements (blue lines). For instance, user number 1 (three leftmost vertical lines) takes ~120 ms to strike his drum after he sees a downward movement of the reference player (red line) and ~20 ms when he sees either an upward and downward movement or a gesture plus upward and downward movement

humans. The presented implementation highlighted the potential of anticipation in human-robot musical ensembles. We find that the application of visual-cues-based anticipation algorithms is of significant importance in human-robot interaction, particularly in the realm of musical performance.

6.3 Query by Movement

In the previous section we described an approach for programming a robotic musician to interpret humans' musical intent by analyzing their physical gestures. There, Shimon used analysis of gestural information to anticipate the following human gestures, which improved synchronization as part of the musical human-robot interaction. In this section we present a different approach for interpreting humans' musical intent by analyzing their gestures. Here, Shimi, the musical robotic companion, detects beat and tempo from a human's head movements, and uses this analysis to retrieve relevant songs from the robot's musical database.

6.3.1 Motivation and Related Work

There are a number of related projects, both commercially and in academia, which attempt to allow users to interact with their musical library in an intuitive user-friendly manner. Some systems utilize automatic playlist generation based on mood or similarity, while others allow users to query general types of music rather than specific songs. A few more sophisticated systems such as Query by Humming, allow users to search for specific songs by humming it when the name or lyrics of the songs are unknown. The system then uses machine learning to find the right matches using music information retrieval techniques [8]. In this chapter, we introduce a new music retrieval system which is based the connection between music and dance, allowing users to query for music based on their physical action rather than what they may be able to articulate through words or singing. This gestural system was particularly designed for a robotic speaker dock, such as Shimi, allowing the listener and the speaker dock to connect using gestures. We title this new system Query by Movement (QBM).

We propose that dancing and rhythmic body movements can serve an effective and intuitive means for music retrieval. Studies suggest a strong cognitive link between music and body motion [9]. Music is shown to promote motor responses in both children and adults [10, 11], while the connections between music and dance can be recognized by observers [12], even when music and movement are temporally separated [10]. Sievers' findings suggest that "music and movement share a common structure that affords equivalent and universal emotional expressions" [13]. Gazzola observed that the neurons responsible for firing when performing an action also fire when hearing an audio representation of that action [14]. The connection is further supported by the historical interdependent development of music and dance [15], and is most often utilized when dancing while listening to music.

Some projects have explored reverse relationship between music and dancing, where dancers generate analogous music through movement. This paradigm has been investigated using wearable sensors as well as external sensors such as computer vision [16–20]. Samburg et. al furthered this investigation in a social music consumption context [21].

The aforementioned projects have been able to report "rich mappings that directly reflected the movement of the dancer" [16] because music and dance share a vocabulary of features. These include concepts such as rhythm, tempo, energy, fluidity, dynamics, mood, and genre. We believe that by using the congruent features of dance to query for music, we can leverage the cognitive link between music and dance and bypass the deciphering and articulation necessary to query used by other methods.

6.3.2 Approach

In order to leverage the congruent features of dance to allow a robotic musician to query music, we first have to extract and demonstrate a correlation between these features in both motion and audio domains. A few candidate data acquisition systems were vetted from related works, including wearable sensors, external sensors, and computer vision. We chose a computer vision implementation on an iPhone because of its ubiquity and relevance to our robotic musician, Shimi, which uses a smartphone for all of its sensing and higher order computation [22].

6.3.2.1 Data Acquisition: Computer Vision

Motion tracking in computer vision can be performed in a number of ways. Most approaches involve a trade-off between computational requirements and degrees of freedom. For example, a powerful means of motion tracking involves object recognition, in which a classifier is trained to detect features, such as a face, hands, or facial features. This could be leveraged to track and consider different body parts individually. Conversely, a simple means of motion tracking is frame differencing, in which pixels in corresponding frames are subtracted from one another, and their difference is summed. This algorithm is inexpensive and robust, but can only supply a generalized view of the movement. We attempted to optimize this trade off by leveraging sparse optical flow, the tracking of a number of arbitrary pixels from frame to frame. This algorithm is cheaper computationally than object detection and tracking, but allows more degrees of freedom than frame differencing, should we need them.

We used the Open Source Computer Vision (OpenCV) library's implementation of the sparse pyramidal version of the Lucas-Kanade optical flow algorithm, as detailed in [23]. Because of the limitations of the iPhone's hardware, an optimization must be made between frame rate and the number of image features we can track with optical flow. As a result, we initialized our image processing by running Shi and Tomasi's "Good Features to Track" corner detection algorithm [24] to identify the 100 strongest corners in the first frame.

We compute the optical flow for these 'good features' on each subsequent frame and generate candidate feature vectors:

$$\sum_{i=1}^{N} |s[i]| \tag{6.3}$$

$$\sum_{i=1}^{N} |s_x[i]| \tag{6.4}$$

$$\sum_{i=1}^{N} |s_y[i]| \tag{6.5}$$

$$\sum_{i=1}^{N} s_x[i] \tag{6.6}$$

$$\sum_{i=1}^{N} s_y[i] \tag{6.7}$$

6.3.2.2 Query

The Echo Nest [25] was a music intelligence platform whose API grants access to over a billion MIR data points on millions of songs. Furthermore, the platform allowed an application to query for music by defining specific ranges of these data points. Many of these calculated features were analogous to dance related concepts, including genre, mood, tempo, and energy. Given our data acquisition system, we did not believe we could extract genre and mood from the optical flow of arbitrarily tracked pixels in a deterministic manner. Therefore, we selected tempo and energy as a 2-D feature space to map from movement to music query.

6.3.3 User Study

We designed a user study to validate our feature selection and optimize the algorithms used to compute them. The study aimed at demonstrating a correlation between calculated data and Echo Nest supplied labels.

6.3.3.1 Method

We used the Echo Nest to query for nine songs at each of three tempos: 90, 120, and 150 bpm. The songs varied in energy. We attempted to control variables that may also contribute to higher-energy dancing by holding constant other Echo Nest-supplied proprietary features such as danceability, artist familiarity, and song "hotttnesss." We allowed genre to be randomly selected. The research subjects were asked to bob their head to 9 short song segments in front of the iPhone's camera. Three songs of varying energy were randomly selected per tempo, and the three tempo classes were presented in a random order. For each dance, we computed the optical flow of their movement and generated the five candidate data vectors detailed in Sect. 2.1.

6.3.3.2 Results

Twenty Georgia Tech students participated in the study. We analyzed results for both the tempo and energy features.

We based our study based on the assumption that tempo correlates strongly and explicitly between music and dance. Extracting the periodicity from a time-varying signal is a well-defined research problem [26]. We found that combining the data vectors (1) (2) and (3) minimized the error rate.

We performed an autocorrelation on these vectors and peak-picked the three strongest correlations per vector. This provided the lag indices which could be multiplied by the mean time elapsed between frames to arrive at a candidate time per beat. We then converted this to beats per minute and scaled the value to fall within the range of 77–153 bpm, and finally rounded it to the nearest 10 bpm. Each data vector yielded 3 tempo candidates, and the mode of the candidates was selected as the tempo induction of a dance signal.

Using this method, we were able to detect the correct tempo in 88.33% of dances from the user study. Of the 11.77% error, half were off by just 10 bpm.

Energy was a more difficult feature to extract in both the movement and audio domains. Energy is a proprietary Echo Nest feature which quantifies the following: "Does [the song] make you want to bop all over the room, or fall into a coma?" [27]. We hypothesized that people tend to move in quicker, more jagged movements when dancing to high-energy music. We expected to see smoother accelerations and decelerations in lower—energy movement. Grunberg found a similar correlation between movement and emotion. When dancing to angrier music, gestures tended to "only take up about two-thirds of the beat" [28].

We found our hypothesis to be represented in the data. Figure 6.8 depicts a subject's movement magnitude vector when dancing to a low energy song versus a high energy

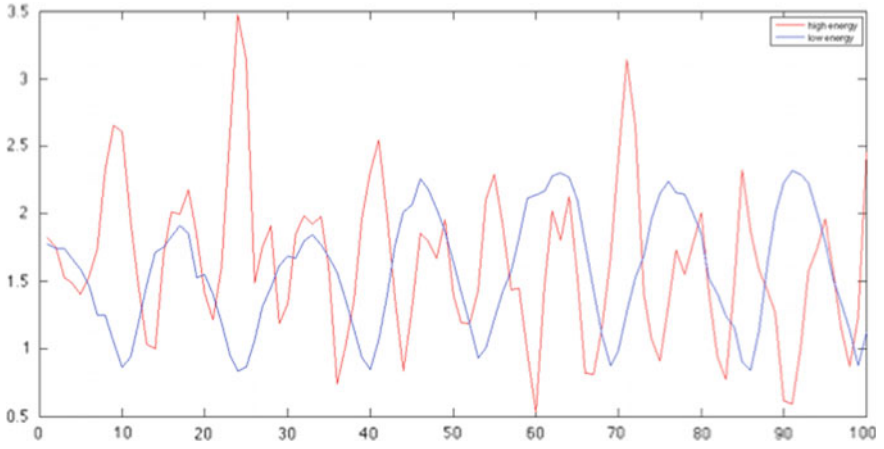


Fig. 6.8 Movement vector for a high and low energy song at the same tempo

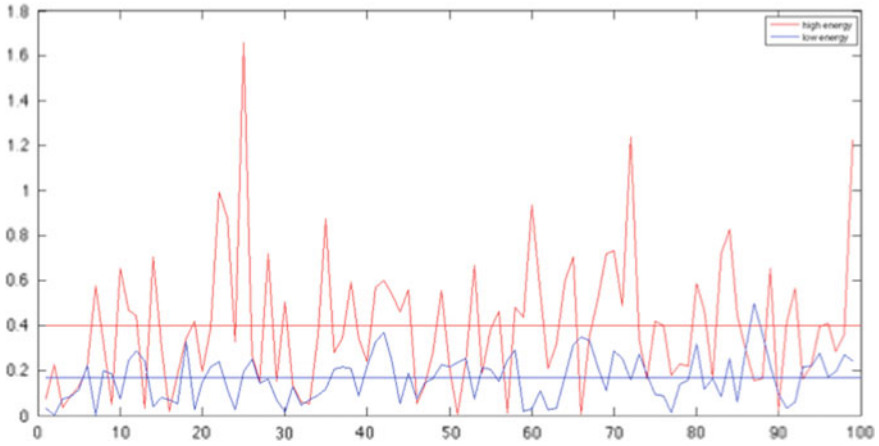


Fig. 6.9 Mean absolute value of the first-order difference of the data in Fig. 6.8

song at the same tempo. Notice the tall, thin spikes in velocity in the high-energy movement as compared to the wider, more gradual velocity fluctuations in the low energy song.

We found that energy is proportional to the mean absolute value of the first order difference of our movement vector. Figure 6.9 depicts the absolute value of the first-order difference of the above data. The mean is superimposed on top and represents our final calculated energy value.

The study results indicated a significant correlation between calculated energy and Echo Nest labeled energy at each tempo, as illustrated in Fig. 6.10. When considering each individual subject's data separately, the correlations became more significant, as illustrated in Fig. 6.11. On a related note, we also found that the research subjects generally responded to an increase or decrease in song energy with a corresponding increase or decrease in their dance energy. This correlation was strong and significant, as illustrated in Fig. 6.12.

6.3.3.3 Discussion

The results of the study suggest an explicit correlation of tempo in the audio and movement domains. While the energy results suggest a correlation too, the stronger individual correlations may suggest that scaling among individuals could be better addressed. For example, one user's medium energy movement may be similar to another user's high-energy movement.

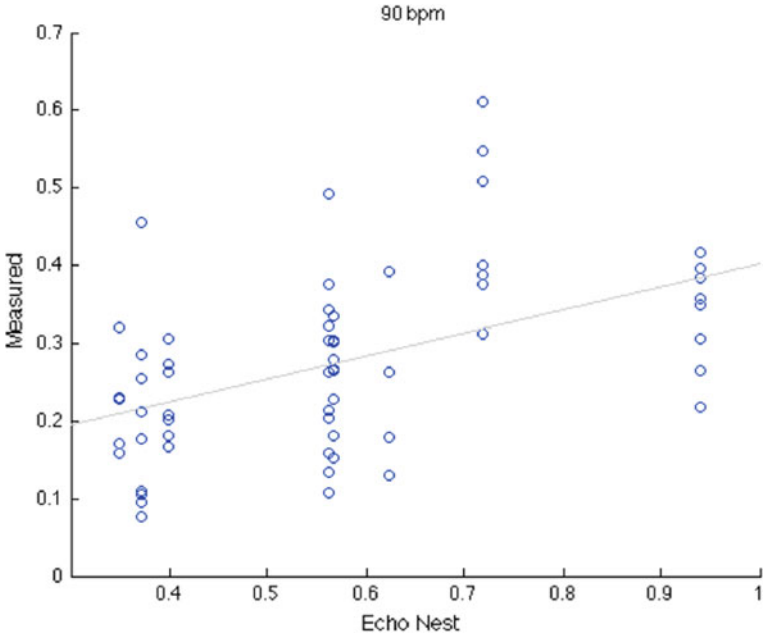


Fig. 6.10 Measured energy versus Echo Nest supplied energy at 90 bpm. r = 0.4784, $p < 0.001$

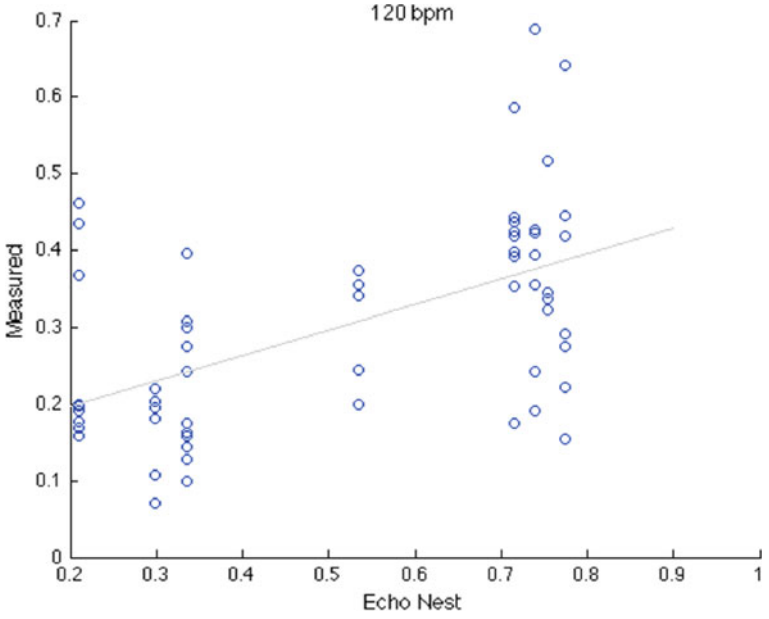


Fig. 6.11 Measured energy versus Echo Nest supplied energy at 120 bpm. r = 0.4845, $p < 0.001$

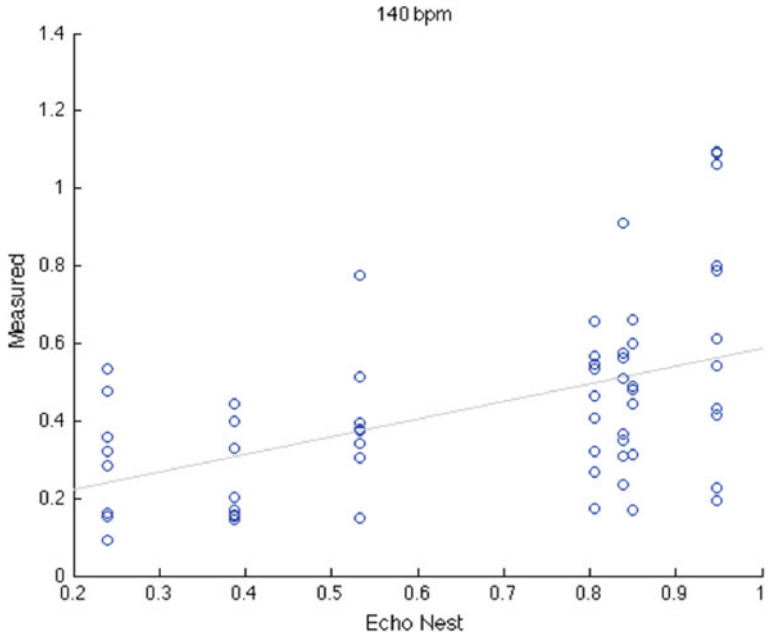


Fig. 6.12 Measured energy versus Echo Nest supplied energy at 140 bpm. r = 0.5484, $p < 0.001$

6.3.4 Implementation

Query by Movement was implemented as an application for the iPhone 5. It used the front-facing camera to capture frames at 30 fps and leveraged the OpenCV implementation of the sparse pyramidal Lucas-Kanande optical flow algorithm [29] initialized with strong corner detection in order to generate movement vectors. We extracted energy and tempo information from these vectors using the algorithms optimized in the first study.

We used this two-dimensional feature space to query for music using *libEchoNest*, Echo Nest's iOS library for the API. Specifically, we queried for songs within (±) 5 bpm of our extracted tempo and within (±) 0.1 of our extracted energy (out of 1). Our query also included a minimum danceability—an Echo Nest feature which is closely related to beat strength and tempo stability—of 0.5 (out of 1). Finally, minimum artist familiarity and "song hotttnesss" were set at 0.5 (out of 1) under the assumption that the user would want to query for something familiar.

The Echo Nest query returned an array of songs which matched the query. The array contained information about each song, including a Spotify URL. We selected a song at random and played it using the *libSpotify* API.

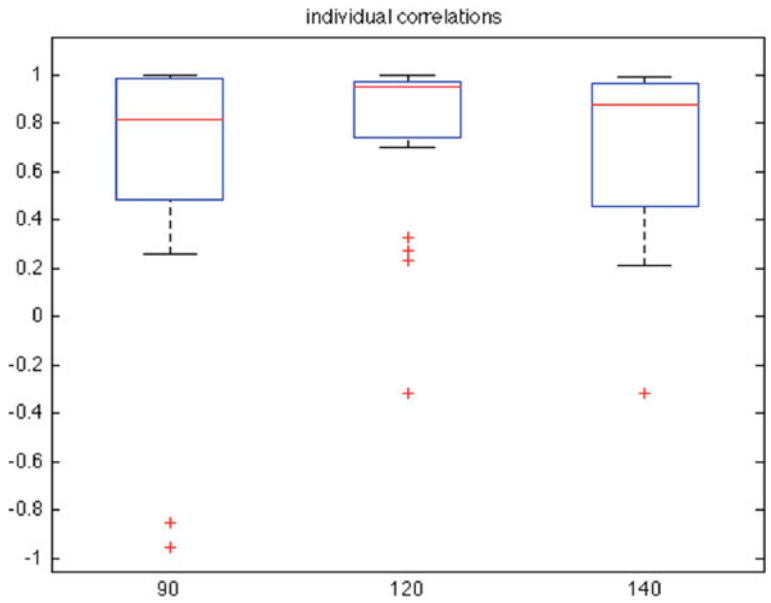


Fig. 6.13 Individual correlations

6.3.5 Evaluation

We validated our feature selection and our system by performing a subjective user study. Research subjects were asked to use the Query By Movement system to query for music. After extracting tempo and energy, the system performed each query in one of four ways:

1. Query using detected tempo and detected energy
2. Query using random tempo and detected energy
3. Query using detected tempo and random energy
4. Query using random tempos and energies

Each research subject queried 8 songs and the system responded in each of the four ways listed above in a random order. After hearing the system's response, the subject recorded a rating on a scale of 1–5, with 5 being the most appropriate. Each subject experienced all query conditions twice.

6.3.6 Results

Ten graduate Georgia Tech students participated in the study. A one-way between subjects analysis of variance (ANOVA) was conducted to compare the effect of the

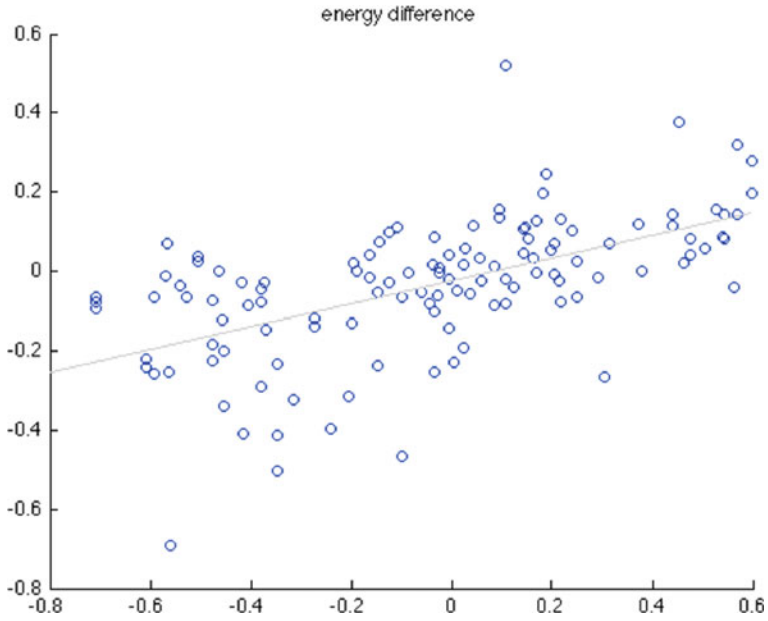


Fig. 6.14 Labeled versus extracted changes in energy. r = 0.57, $p < 0.001$

different testing conditions. The standard query performed significantly better than the randomized queries, as depicted in Fig. 6.15.

6.3.7 Discussion

The results from this user study indicate that our calculated features contribute to an appropriate query using our Query By Movement system. Because appropriateness is a measure related to expectation, our results may suggest that our features were transparent across the audio and movement domains. We also speculate that with more study participants, we may find that energy more strongly contributes to an appropriate query than tempo does.

6.3.8 Future Work and Conclusions

A natural and logical continuation of investigating a Query By Movement system is the construction of a higher-dimensional feature space. In order to achieve this, new features must be able to be extracted from both the movement and audio domains. Features such as mood and genre may contribute to a more compelling system.

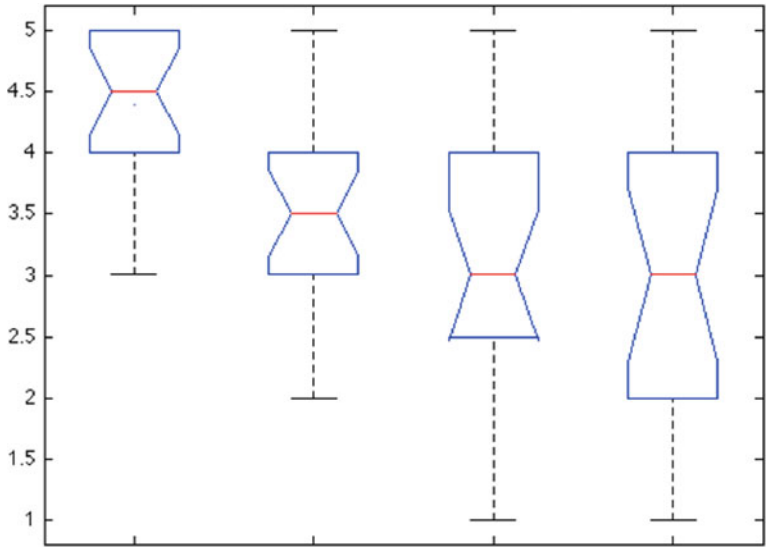


Fig. 6.15 ANOVA of the 1–4 testing conditions on x-axis and the user rating on y-axis

The Microsoft Kinect, for example, is capable of sophisticated video feature tracking while remaining robust and somewhat ubiquitous. Tracking of individual body parts could possibly grant enough degrees of freedom to begin considering machine learning as a means to classify mood or genre in dance.

6.4 A Robotic Movie Composer

The last project described in this chapter is more artistic in nature, rather than scientific. Here, we extended our work in computer vision for robotic musicianship to allow Shimon to analyze movie imagery and compose a sound track based on its analysis. As part of the project, Shimon watched, analyzed and composed a marimba track in real-time for a movie titled *The Space Between Fragility Curves* by video artist Janet Biggs. Shimon was designed to act like a traditional silent film composer. The original one channel video, set at the Mars Desert Research Station in Utah, premiered on the 17th of May 2018, at the Festival International de la Imagen in Manizales, Colombia. The two channel version, including footage of Shimon interspersed amongst the footage from the Mars Research station, premiered on the 14th of November, May 2018 at the Museo de la Ciencia y el Cosmos, in Tenerife, Canary Islands.

Fig. 6.16 Shimon practicing the space between fragility curves (Photo by Janet Biggs)

6.4.1 Visual Analysis

The visual analysis focused on extracting key elements from the visual elements that a human may track within this specific film. Custom analysis tools were built in MaxMSP's Jitter, reading a JSON file generated by Microsoft's Video Insights. The JSON file included identified faces and their time and location on screen. It also included objects and a location analysis, including labels defining if the scene was indoors or outdoors as well as the surrounding landscape. Jitter was used to extract director aesthetic choices, defined as conscious film choices that set the tone, style and pace of film. These tracked aesthetic choices included panning, zoom, similarity between camera angle changes, character movement and coloration. Parameters were then used to create a meta analysis of the overall pacing of the movie.

6.4.2 Music Generation

A musical arc was set by the director, dividing the film into four minutes of character and object driven musical generation, with two segments given independent algorithmic processes. At the beginning of each run, four melodies are generated using a Markov Model that was trained on melodies from Kubrick film scores. A Markov Chain is a probability based model that bases future choices on past events. In this case we referred to three past events, using a third order Markov Model for pitch and a separate fifth order model for rhythm, both trained on the same data. Two melodies were assigned to characters, with the other two melodies set for indoor and outdoor

scenes. Melodies were then used throughout the film, blending between each other dependent on what was occurring on screen. Melodies were varied based on movement of the chosen characters on screen, their position and external surroundings.

The first of the separate sections was centered inside an air chamber with director requesting a claustrophobic soundtrack. For this scene an embodied approach was used with Shimon. Shimon's design encourages the use of semitones, as both can be hit without the arms moving. Chords were then built around a harmonic rule set. The second section was the conclusion of the film, which used Euclidean rhythms [30], commonly found in algorithmic composition. The scene features one of the main characters riding an ATV across the desert. The number of notes per cycle was set based upon the movement of the ATV and position on screen.

6.4.3 Informal Feedback

For this system we did not conduct a user study. We considered it an artistic piece, which can serve as a prototype to generate ideas for a future more complete system. The film premiered in the Canary Islands and has had multiple other showings and was well received by audiences and critics. Comments from the film director demonstrated the importance of the link between visuals, "I love how clear it is that Shimon is making choices from the footage".[1] Follow up emails also suggested that the generative material was effective however the overall musical structure could be improved "I watched the demos again and am super happy with the opening segment. The only other note that I would give for the second half is to increase the rhythm and tonal (tune) quality".[2]

6.4.4 Discussion

After positive feedback from the director and informal viewings, we reviewed the key concepts that emerged from the project. Extracting director aesthetic choices such as movement on screen and panning allowed an instant level of feedback that helped align the score with visuals. To some degree this naturally creates musical arcs matching the film's arc, however with only this information the music is always supporting the visuals and never complementing or adding new elements to the film. Likewise, character-based motives were received positively by audiences, yet without intelligently changing these elements based on the character performance they might also fall into a directly supporting role. Most significantly we came to believe that there was no reason for future systems to work in real-time. Shimon composing a new

[1] Biggs, Janet. "Re: Demos" Message to Richard Savery, 26 October, 2017. Email.

[2] Biggs, Janet. "Re: Demos" Message to Richard Savery, 6 November, 2017. Email.

version for the film through each viewing provided a level of novelty but in reality the film will always be set beforehand and live film scoring is a vastly different process to the usual work flow of a film composer.

References

1. Johansson, Birger, and Christian Balkenius. 2006. An experimental study of anticipation in simple robot navigation. In *Workshop on anticipatory behavior in adaptive learning systems*, 365–378. Springer.
2. Eyssel, Friederike, Dieta Kuchenbrandt, and Simon Bobinger. 2011. Effects of anticipated human-robot interaction and predictability of robot behavior on perceptions of anthropomorphism. In *Proceedings of the 6th international conference on Human-robot interaction*, 61–68. ACM.
3. Gielniak, Michael J., and Andrea L Thomaz. 2011. Generating anticipation in robot motion. In *2011 RO-MAN*.
4. Hoffman, Guy. 2010. Anticipation in human-robot interaction. In *2010 AAAI Spring Symposium Series*.
5. Wang, Zhikun, Christoph H. Lampert, Katharina Mulling, Bernhard Scholkopf, and Jan Peters. 2011. Learning anticipation policies for robot table tennis. In *2011 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 332–337. IEEE.
6. Bradski, Gary, and Adrian Kaehler. 2008. *Learning OpenCV: Computer vision with the OpenCV library*. O'reilly.
7. Puckette, Miller S., Miller S. Puckette Ucsd, Theodore Apel, et al. 1998. Real-time audio analysis tools for Pd and MSP.
8. Ghias, Asif, Jonathan Logan, David Chamberlin, and Brian C. Smith. 1995. Query by humming: Musical information retrieval in an audio database. In *Proceedings of the third ACM international conference on Multimedia*, 231–236. ACM.
9. Lewis, Barbara E. 1988. The effect of movement-based instruction on first-and third-graders' achievement in selected music listening skills. *Psychology of Music* 16 (2): 128–142.
10. Mitchell, Robert W., and Matthew C. Gallaher. 2001. Embodying music: Matching music and dance in memory. *Music Perception* 19 (1): 65–85.
11. Phillips-Silver, Jessica, and Laurel J. Trainor. 2005. Feeling the beat: Movement influences infant rhythm perception. *Science* 308 (5727): 1430–1430.
12. Krumhansl, Carol L, and Diana Lynn Schenck. 1997. Can dance reflect the structural and expressive qualities of music? A perceptual experiment on Balanchine's choreography of Mozart's divertimento no. 15. *Musicae Scientiae* 1 (1): 63–85.
13. Sievers, Beau, Larry Polansky, Michael Casey, and Thalia Wheatley. 2013. Music and movement share a dynamic structure that supports universal expressions of emotion. *Proceedings of the National Academy of Sciences* 110 (1): 70–75.
14. Gazzola, Valeria, Lisa Aziz-Zadeh, and Christian Keysers. 2006. Empathy and the somatotopic auditory mirror system in humans. *Current Biology* 16 (18): 1824–1829.
15. Grosse, Ernst. 1897. *The beginnings of art*, vol. 4. D. Appleton and Company.
16. Paradiso, Joseph A., and Hu, Eric. 1997. Expressive footwear for computer-augmented dance performance. In *First international symposium on wearable computers, 1997. Digest of papers*, 165–166. IEEE.
17. Paradiso, Joseph, and Flavia Sparacino. 1997. Optical tracking for music and dance performance. *Optical 3-D measurement techniques IV*, 11–18.
18. Camurri, Antonio, Shuji Hashimoto, Matteo Ricchetti, Andrea Ricci, Kenji Suzuki, Riccardo Trocca, and Gualtiero Volpe. 2000. Eyesweb: Toward gesture and affect recognition in interactive dance and music systems. *Computer Music Journal* 24 (1): 57–69.

19. Aylward, Ryan, and Joseph A. Paradiso. Sensemble: A wireless, compact, multi-user sensor system for interactive dance. In *Proceedings of the 2006 conference on new interfaces for musical expression*, 134–139. IRCAM–Centre Pompidou.
20. Winkler, Todd. 1998. Motion-sensing music: Artistic and technical challenges in two works for dance. In *Proceedings of the international computer music conference*.
21. Samberg, Joshua, Armando Fox, and Maureen Stone. 2002. iClub, an interactive dance club. In *ADJUNCT PROCEEDINGS*, 73.
22. Bretan, Mason, and Gil Weinberg. 2014. Chronicles of a robotic musical companion. In *Proceedings of the 2014 conference on new interfaces for musical expression*. University of London.
23. Jean-Yves Bouguet. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation*, 5, 2001.
24. Shi, Jianbo, and Carlo Tomasi. 1994. Good features to track. In *Proceedings CVPR'94, 1994 IEEE computer society conference on computer vision and pattern recognition*, 593–600. IEEE.
25. Jehan, Tristan, Paul Lamere, and Brian Whitman. 2010. Music retrieval from everything. In *Proceedings of the international conference on Multimedia information retrieval*, 245–246. ACM.
26. Gouyon, Fabien, Anssi Klapuri, Simon Dixon, Miguel Alonso, George Tzanetakis, Christian Uhle, and Pedro Cano. 2006. An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Audio, Speech, and Language Processing* 14 (5): 1832–1844.
27. Sundram, J. 2013. Danceability and energy: Introducing echo nest attributes.
28. Grunberg, David K., Alyssa M. Batula, Erik M. Schmidt, and Youngmoo E. Kim. Affective gesturing with music mood recognition.
29. Baker, Simon, and Iain Matthews. 2004. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision* 56 (3): 221–255.
30. Toussaint, Godfried. 2005. The Euclidean algorithm generates traditional musical rhythms. In *BRIDGES: Mathematical connections in art, music and science*, 1–25.

Chapter 7
"Wear it"—Wearable Robotic Musicians

7.1 Abstract

Recent developments in wearable technology can help people with disabilities regain their lost capabilities, merging their biological body with robotic enhancements. Myoelectric prosthetic hands, for example, allow amputees to perform basic daily-life activities by sensing and analyzing electric activity from their residual limbs, which is then used to actuate a robotic hand. These new developments not only bring back lost functionalities, but can also provide humanly impossible capabilities, turning those who were considered disabled to become super-abled. The next frontier of Robotic Musicianship research at Georgia Tech focuses on the development of wearable robotic limbs that allow not only amputees, but able-bodied people as well, to play music like no human can, with virtuosity and speed that are humanly impossible. This chapter addresses the promises and challenges of the new frontier of wearable robotic musicians, from a Robotic Prosthetic Drumming Arm that contains a drumming stick with a "mind of its own", to a "Third Arm" that augments able-bodied drummers, to the Skywalker Piano Hand that uses deep learning predictions from ultrasound muscle data to allow amputees to play the piano using dexterous and expressive finger gestures.

7.2 The Robotic Drumming Prosthetic Arm

The principal objective of the Robotic Drumming Prosthetic Arm (see Fig. 7.1) was to simulate the expressive role that fingers play in drumming. This primarily includes controlling the manner in which the drum stick rebounds after initial impact. To achieve this goal, we used a DC motor driven by a variable impedance control framework in a shared control robotic prosthetic arm system. The user's ability to perform with and control the prosthesis was evaluated using a musical synchronization study. A secondary objective of the prosthesis is to explore the implications of

G. Weinberg et al., *Robotic Musicianship*, Automation, Collaboration,
& E-Services 8, https://doi.org/10.1007/978-3-030-38930-7_7

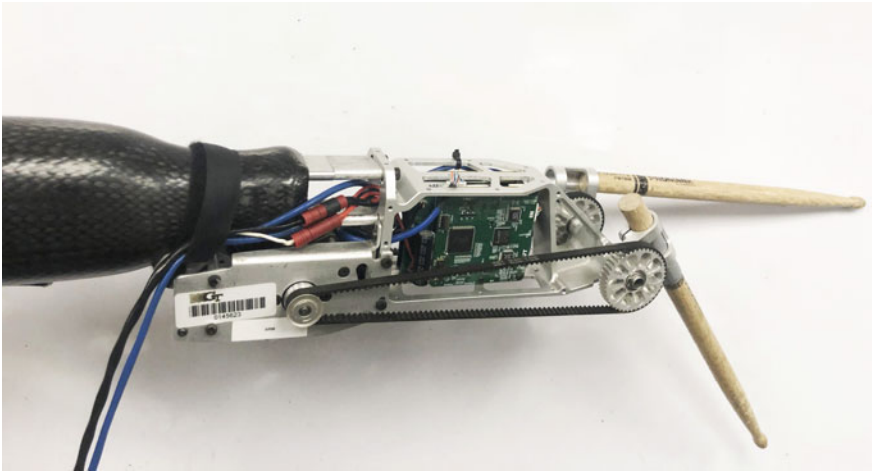


Fig. 7.1 The drumming robotic prosthesis

musical expression and human-robotic interaction when a second autonomous stick is added to the prosthesis. This wearable "robotic musician" interacted with the user by listening to the music and responding with different rhythms and other musical behaviors. We recount some empirical findings based on the user's experience of performing under such a paradigm.

7.2.1 Background

As robots become more pervasive in everyday life and begin to function outside of purely industrial environments, it is essential for methods that support fluid and intuitive human-robotic interactions to be designed and implemented. Such methods must enable a natural flow of information between human and robot in both the cognitive and physical domains.

Wearable Robotics, such as prosthetic, orthotics, and exoskeletons, provide a unique and promising platform for research in robotic interaction and control because of the intimate user interface they offer. These systems may extend, enhance, or replace limbs [1] and in some cases may even equip the user with an additional limb (such as a third arm or additional fingers) [2]. Typically, wearable robots attempt to understand a person's intent through some means of sensing including button presses, electromyography (EMG), and electroencephalography (EEG), allowing it to behave according to the user's desires. These robotic behaviors and functions assist the user by providing capabilities that may have been lost due to amputation or were never present as a result of natural human anatomy.

In this section we describe a wearable robotic prosthetic designed for an amputee drummer. Electromechanical prostheses are composed of sensing and control modules and are designed to recreate natural human function. Dellon and Matsuoka describe three areas of challenge when attempting to replicate human function using prosthetics: electromechanical implementation, extraction of intent, and interface design or usability [3]. The demands and difficulties inherent to these areas are significantly amplified when designing a device that enables performance on a musical instrument. The relationship between a musician and their musical instrument is one of extreme intimacy and requires a level of chemistry that is not commonly apparent or necessary in other human-interface interactions.

Musical virtuosity is dependent on increased dexterity, accurate timing, and subtle control. Permitting an amputee the ability to attain such virtuosic control utilizing the current state-of-the-art sensing and mechanical technology is a challenging task. As a result, our first attempt at developing a robotic drumming arm falls short of a perfect emulation of a human drumming arm, however, we see many successes in our methods and great promise to expand upon and improve the technology. Additionally, our wearable robotic system not only recreates, but also enhances natural human ability through aspects such as increased speed and endurance. The potential for new and interesting music that leverages such augmented abilities can lead to novel ways for musicians (disabled and able-bodied alike) to incorporate robotics in performance.

In the first part of this section we describe our motivation, design, and evaluation of the robotic drumming arm. The primary objective is to recreate the functionality lost due to amputation. The system employs a variable impedance control framework that requires the human and robot to work together to achieve some desired musical output. We evaluate the system through experiments requiring performance and completion of musical synchronization tasks.

Much of our research focuses on robotic musicianship—the design and construction of robots capable of understanding and performing music in interactive scenarios such that the human and machines share control over the final musical output. Later in this chapter we describe the incorporation of an additional stick to the prosthetic. The second drumstick functions as a robotic musician and responds to the music and improvises based on computational music analysis.

While the amputee drummer does not have full control over the second stick, he can react and collaborate with it, creating novel interactive human-robot collaboration. In designing the algorithms controlling both drum sticks, we have taken into consideration the shared control aspects that the human and the machine have over the final musical result. We also consider the physical interdependencies that may occur due to the fact the robotic musician is an extension of the human's own body. While the artificial intelligence is designed to generate different stroke types and rhythmic patterns, the human drummer can influence the final musical result through his own motion and manipulation of the distance between the autonomous stick and the drum. Concurrently, the autonomous stick has the opportunity to complement, contrast, or disrupt the drummer's own patterns.

Such a wearable system yields several research questions: What type of interaction between the human and robot is optimal for achieving fluid and comfortable playability that encourages musical cooperation, yet remains non-restrictive? What type of sensing by the robot is necessary to support creative musicianship? Is it possible for other musicians in the group to interact with and influence the robotic musician?

7.2.2 Related Work

According to Schirner et al. most robots can be categorized into one of two categories determined by the level of human-robot interaction involved: (1) Robots that are fully autonomous performing specific tasks, such as industrial robots working on factory assembly lines and (2) robots that are tele-operated with minimal artificial intelligence capabilities that are considered to be passive tools entirely controlled by humans [4]. These categories describe the two extremes of which humans and robots interact, however, in many cases a more interdependent relationship between human and machine is desired and a shared control paradigm is adopted. Under a shared control framework a synergistic collaboration between the human and the robot exists in order to achieve a goal or complete a task in such a way that the cognitive and physical load on the human is significantly reduced.

Shared control robotic systems are adopted in several domains including assistive technology, teaching environments, and even more artistic and creative enterprises. Some human-wheelchair systems employ such a shared control framework to safely navigate an obstacle filled environment while accounting for variability in user performance [5]. Tele-operated surgical robots can be used as training tools for novice surgeons through a paradigm in which the control is shared between mentor and trainee. The haptic feedback forces provided to each surgeon are inversely proportional to the control authority. The relative expertise levels also determine the degree to which the motion of the robot is affected when the individual commands are issued, thereby, reducing the chances of an inadvertent error [6]. Robotic musicianship research focuses on exploring new avenues of musical expression through human-robotic interaction [7]. The artistic premise being robots that perform and generate original music can push humans to unknown musical white spaces inspiring creativity and even the development of new genres.

Within the shared control paradigm itself two categories [8] are usually identified based on the type of human robot interaction involved: (1) *Input mixing shared control* in which the user provides input to the robot and the robot provides input to the underlying control system and (2) *haptic shared control* in which the user and robot directly provide input to the control system. The system we present in this work incorporates elements of both paradigms, allowing the user to influence the behavior of the robot as well as directly influencing the final musical product.

A common feature found among different shared control systems is the presence of a closed feedback control loop. The feedback can take a number of forms including

visual [9], haptic [6], and auditory [10, 11]. Often, a system incorporates multiple modes of feedback [12, 13]. State-of-the-art prostheses generally provide visual and haptic feedback to the user [14]. However, despite the advantages of shared control in these applications, it is important for the users to have the capability to initiate a passive mode during situations when the robot behavior becomes intrusive or if the goal state is not reached successfully or changes midway [9].

Advances in areas of material sciences, microprocessor technology, and pattern recognition have encouraged the development of reliable and easy to use upper extremity prosthetic technology [15]. In such technology a shared control framework is used to reduce the amount of attention required of the user [14]. Much of the current research in upper-limb prosthetics focuses on improving the performance of grasping [16]. Though several types of control mechanisms exist for assistive technologies such as brain-machine interfaces [17, 18] and body powered prostheses, almost all of the state-of-the-art commercially available hand prostheses use EMG signals from the residual amputation stump for control of electromechanical joint movement (ProDigits and iLimb from Touch Bionics, Otto Bock myoelectric prosthesis). EMG signals are considered to be the most physiologically natural and the greatest advantage of the myoelectric prosthesis is the increased grasp strength that it provides [1].

Wearable robotics are not only used for amputees or the physically impaired, but are also used for augmenting the abilities of natural human anatomy. Supernumerary limbs and fingers help to reduce load on the body and can provide additional support in completing specific tasks [19, 20]. Unlike most commonly seen wearable prostheses, which exhibit an input mixing shared control paradigm, these supernumerary robotic limbs (SRLs) exhibit a haptic shared control paradigm [2]. The person and robot coordinate their efforts while both directly influencing the interface.

The objectives of our work require functionalities that support both shared control paradigms. In the remaining sections we describe these objectives and our methods for attaining them.

7.2.3 Motivation

Prostheses come in many different forms depending on the users' needs. When developing a prosthetic, designers consider the location of the amputation, the necessary functions the device should be able to perform, the ideal communication interface (such as EMG, EEG, potentiometers, buttons, etc.), and any physical constraints that if failed to be addressed may render the device to be more hindering rather than assisting (such as weight, size, shape, and latency). In this section we discuss our drumming prosthesis in detail.

Jason Barnes is a drummer whom, due to an unfortunate accident, required a transradial amputation (below the elbow through the forearm). This means the degree-of-freedom (DoF) provided by the elbow is present, but the DoFs offered by the wrist and fingers have been lost. The nature of such an amputation requires a novel

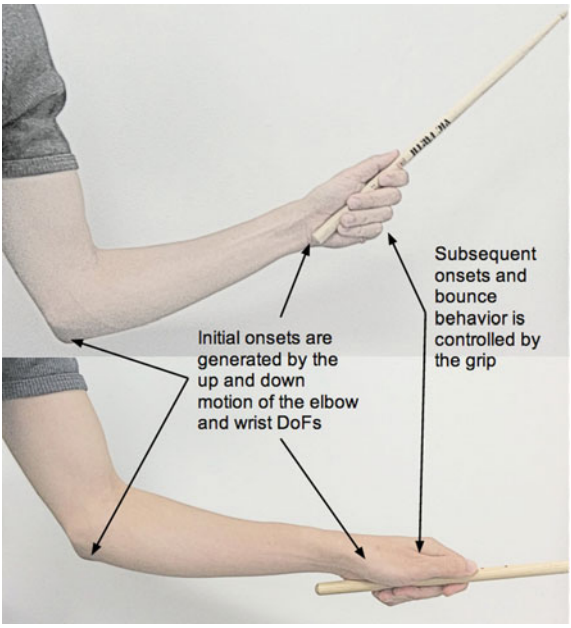


Fig. 7.2 Creating multiple bounce drum strikes

prosthetic design that provides a functionality that is not typically addressed by designers of stand-alone robotic percussionists.

The methods and strategies of current robotic drummers are not applicable for Jason because most robotic drumming machines use solenoid based systems that only provide the necessary vertical motion for striking a drum [10, 11, 21, 22]. In these systems a single DoF is sufficient for a wide range of musicality and the speeds at which the solenoids are capable of moving greatly exceed that of a human drummer. This method is equivalent to a human using only his elbow to generate each drum strike. In reality, human drummers use a combination of several DoFs provided by the elbow, wrist, and fingers allowing for many types of strokes producing different sounds and speeds (see Fig. 7.2).

For two hits separated by a large temporal interval a drummer uses a single-stroke in which the elbow moves up and down for each strike (similar to a single DoF system employed by robots). For two hits separated by a very small temporal interval a drummer may use a *double-stroke* in which the elbow moves up and down once for the first strike and the second strike is achieved by adjusting the fingers' grip on the drumstick to generate a bounce. Single-strokes generated by solenoids and motors can move at such great speeds that it is not necessary to create double-stroke functionality. Though a double-stroke is considered a fundamental rudiment of drumming it is a challenging task to recreate in robotics. Using fast single-strokes with solenoids simplifies the engineering, while achieving similar results to double-strokes.

It is possible to use the more common single solenoid system allowing us to disregard bounce, however, in order to leverage Jason's previous knowledge of drumming and take advantage of his residual elbow DoF it is important to model natural human function as close as possible. The system should be designed in such a way that it allows Jason to make use of his elbow in a manner in which he is familiar and that is typical of able-bodied drummers. Therefore, the primary objective is to develop a prosthetic that enables the initial onset of a strike to be generated using the elbow DoF and then simulates the role of the fingers to control the rebound for subsequent strikes.

7.2.4 Design

A number of considerations needed to be addressed in designing the drumming prosthesis. The device had to be robust enough to handle frequent large impact loads inherent to drumming. Furthermore, in order to reduce unnecessary muscle fatigue the prosthesis needed to be as light as possible with a center of gravity close to the elbow to reduce rotational inertia. An original non-anthropomorphic design allowed us to reduce the amount of hardware necessary and control for the center of mass. Since the arm had to support the physically and cognitively demanding activity of drumming, it was crucial to find a balance between lightweight design and high performance so that the cognitive and physical loads of operating the device were manageable. Therefore, we attempted to create finger and grip functionality using a single motor that would drive tension, and therefore the rebound behavior, of a mounted drum stick (see Fig. 7.3).

A large multi-stage gearbox, typical of robotic actuators, was deemed to be too complex and massive for our purposes. Therefore, brushless gimbal motors were

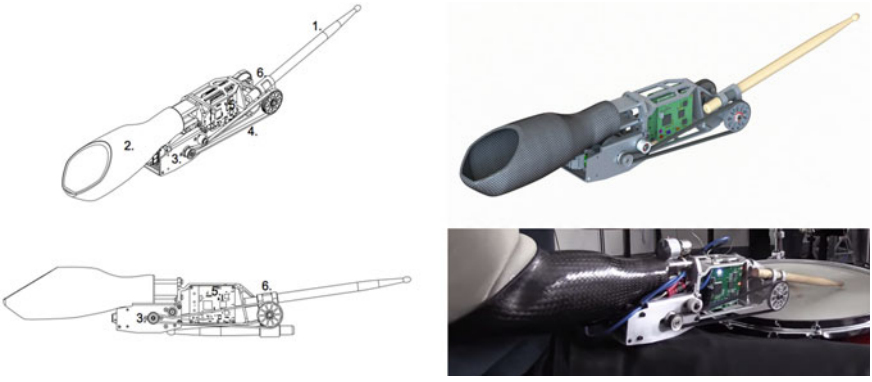


Fig. 7.3 **1.** Drum stick **2.** socket **3.** casing for motors **4.** belt connecting motor to drum stick mount **5.** processor **6.** drum stick mount

chosen for the device. They exhibit a low kV rating and are more capable of providing higher torques at low speeds, compared to standard brushless motors used in robotics, which run at high speeds and require significant gear reduction for usable output torque and speeds. The gimbal motors also provided rapid acceleration, an added bonus for the augmented human ability of extremely fast drumming.

A single stage timing belt drive was chosen for the prosthetic as it could withstand the repeated shock loading common during drumming. Additionally, a belt drive added a compliant element between the output and the motors. It was important to have low to zero backlash to facilitate smooth drumming and precise control. Having a belt drive allowed the motors (the heaviest components of the system) to be placed close to the elbow, thereby, reducing fatigue and serving as a mechanical fuse. Although belts suffer wear and tear and might fail over time due to extended use, they can easily be replaced and have a low gear ratio. The final gear ratio of the device was less than 3:1.

The primary structure of the arm was a unibody frame machined from a single billet of aluminum. Diagonal supports were cut into the frame, which helped in removing as much material as possible without sacrificing structural rigidity. A structure cut out of a single piece of aluminum has the advantage of having less weight than one which requires several parts and is held together by heavy steel fasteners. Furthermore, no extra hardware was required to hold the frame together. The frame had an i-beam profile that was closed on each end for stiffness. The control electronics were integrated into the main structure with cavities and mounting features in the middle of the frame. Cutouts along the top and bottom gave access to connectors for sensors, power, and control bus. The motors were located in a custom sheet metal enclosure at the back of the main structure, tucked underneath the carbon fiber socket and slightly offset downwards.

The main microprocessor boards used in the device were proprietary boards from *Meka Robotics*, mounted directly to the main structure. Advanced high precision motor control was facilitated by using high resolution optical encoders mounted to the rear of the motors. The boards were equipped with analog inputs for taking input from various sensors (EMG, potentiometers) and came with an on-board 9-axis accelerometer. The communication between the host computer and the board was done over a high speed *EtherCat* based control bus. The development platform (*MATLAB/Simulink*) was extremely flexible and could be used to rapidly employ different types of control schemes.

The primary method of sensing was electromyography. Two sets of muscles on the residual limb were chosen as the principal sensory inputs to the prosthesis. The action of playing a drumset involved several muscle groups throughout the entire body. We selected the extensor carpi ulnaris and flexor carpi radialis since these muscles do not play a significant role in the initial onset of a stroke (the up-down motion of the hand), thus, limiting cross-talk with other muscles used during drumming.

These muscles were used to generate control signals that vary the parameters of the PID control system for the motors. Therefore, it became important that the system was fairly robust against false positives, otherwise, unwanted changes in the stick rebound could affect the quality of the musical output.

For each muscle group, a bipolar EMG recording paradigm was adopted. Surface electrodes were used with the ground electrode attached to the bony region directly below the elbow. It is typical of commercial prostheses to use hardware based differential amplifiers for bipolar EMG recordings. However, due to weight and general bulkiness of such amplifiers we perform differential amplification in software. The EMG electrodes are connected to a computer using a standard audio interface with built in pre-amplification. Following pre-amplification differential amplification, full wave rectification, and a gain multiplication are performed on the raw EMG input (see Fig. 7.4).

An RMS-based moving average filter (MAF) is applied on the amplified signal in order to smooth the signal by removing unwanted high frequency noise content. This stage is followed by three bi-quad filters whose coefficients are set in such a way that they act as a low pass filter (with cutoff of 520 Hz), a notch filter (with center frequency around 180 Hz and a Q-factor of approximately 1.5) and a high pass filter (with a cutoff frequency of approximately 20–30 Hz). The EMG muscle activations typically do not contain frequencies above 500 Hz and the notch filter is used to reject the power-line interference. Although this may degrade signal quality, we are not as interested in power spectral density measures and the degradation has little effect on the onset detection methods. In addition to the standard power-line interference, electromagnetic noise due to the motors and on-board electronics also poses a big challenge for clean signal acquisition. The high pass filter is used to eliminate motion artifacts that arise due to the electrodes moving with respect to the skin or cable motion. A noise gate is used after this stage to remove low level residual noise as it further helps in improving the robustness of the onset detection system.

The real-time onset detection algorithm employed in this system is based on a bounded-Q filter-bank decomposition of the incoming signal and this has far superior performance than traditional envelope follower based onset detection methods [23]. This method is useful for percussion-like transients which do not lend themselves

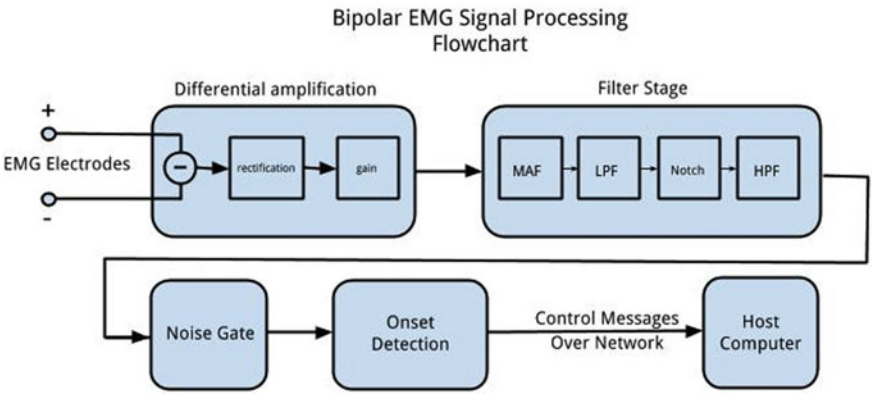


Fig. 7.4 EMG signal flow and processing

to sinusoidal decomposition. Various parameters such as the minimum velocity and the minimum temporal distance between the onsets can be adjusted as required. The detected onsets are mapped to various functionalities such as triggering single strikes of the stick, incrementally varying the parameters of the PID motor control system, etc. Other features, such as the slowly evolving envelope contour of the EMG signal can easily be extracted, but due to the non-stationary nature of bio-signals in general, a meaningful mapping of continuously varying features to low-level parameters of the system is difficult.

We evaluated the EMG filtering and onset detection system by constructing a labeled dataset based on Jason's muscle activations. Time series recordings from the two principal muscle locations were collected. When Jason contracted one of the muscles he provided a verbal cue indicating a perceived onset had occurred. These cues were used to label the recordings, which were then used for an objective evaluation. We achieved an F-measure of $F1 = 0.91$. Additionally, Jason provided subjective validation by designating the detection accuracy as sufficient for his needs.

For the purposes of the experiment that was designed to evaluate the entire prosthesis system, the onsets detected were used to change the proportional gain (kp) of the PID controller system in an incremental fashion. The useful range of the proportional gain parameter was determined empirically and the EMG onsets were used to change the parameter in a smooth linear fashion within this range.

7.2.5 Evaluation

The efficacy and success of a shared control system is typically evaluated by comparing the amount of effort required of the user with and without the system. For example, one may measure the time taken to complete a task [24] or quantify the task difficulty and cognitive load involved using questionnaire forms [9]. We evaluate our system by measuring Jason's performance of a musical synchronization task. Our hypothesis is that by using a variable impedance system to control the stick tension, Jason will be able to achieve the different stroke types required of accomplished drummers. In this situation the initial onsets are achieved through the physical up and down motion of Jason's elbow. The bounce behavior, as a result of the stick tension, is controlled by the motor system and PID controller. Jason is able to manipulate the PID parameters using his muscles through EMG control.

To test our hypothesis Jason was asked to play several common rhythmic motifs that require different types of strokes and bounce behaviors using the prosthetic. The objective is not to examine Jason's proficiency as a musician, but rather whether the electromechanical prosthetic design is sufficient for the needs of a drummer. In other words, does it effectively simulate the function of a drummer's fingers and grip in order to generate different stroke types? Additionally, is Jason capable of setting the proper PID parameters in order to generate the desired stroke type using the EMG interface? Therefore, the rhythmic patterns Jason was asked to play were patterns with which he was very familiar and had practiced for many hours both before

and after his amputation (using a spring loaded drumming prosthetic he designed himself).

A 2×1 within subject study was conducted. Jason performed the study under two conditions: (1) using his own spring-loaded prosthetic and (2) using the new electromechanical prosthetic with variable impedance. The spring-loaded prosthetic functions similarly to our electromechanical prosthetic, however, instead of a motor simulating the grip to achieve a bounce, a spring is used. This prosthetic is the primary drumming tool Jason currently uses and has been using since his amputation. We chose to compare our prosthetic to the spring-loaded prosthetic as opposed to comparing the results to what other able-bodied drummers are able to achieve because a between subjects test would make it difficult to differentiate between the affects of the prosthetic and differences in participants' musical proficiencies. Likewise, comparing results between Jason's biological left arm and our new prosthetic may also simply distinguish the natural strengths and weaknesses between his two arms. Drummers often have a dominant hand, which can outperform the other.

Therefore, we chose to compare the electromechanical prosthetic directly to Jason's spring-loaded prosthetic. We are confident that in doing this we can directly evaluate the effect our prosthetic has on Jason's performance abilities. Additionally, Jason is already considered a proficient drummer when using his spring-loaded prosthetic setting the musical baseline of our electromechanical design at a performance level similar to professional musicians. Despite Jason's skill with his spring-loaded prosthetic, there are limitations of using a spring to generate bounce, hence, our motivation for a new design. Any improvements as a result of our prosthetic might be small, but are essential for becoming a more well-rounded and accomplished musician.

The prime issue with Jason's prosthetic is that the spring's tension cannot be manipulated, thus, only a single bounce behavior is possible. Variable bounce behaviors will allow for double strokes with varying intervals making our prosthetic more amenable to a wider range of tempi. To test this hypothesis Jason completed rhythmic synchronization tasks based on the common rhythmic drumming patterns. The notion is that Jason will be able to more accurately and consistently play rhythmic motifs if the rebound of the stick behaves in the appropriate manner. This will result in better synchronization between Jason's play and the perfectly timed reference audio. The rhythmic motifs were generated using a single audio drum sample that was arranged according to the appropriate temporal patterns using the computer's internal clock. Five patterns were chosen and each was played at tempi starting from 90 to 210 bpm increasing at 10 bpm intervals. The procedure was as follows:

1. Listen to computer audio of rhythmic motif for two measures
2. Perform four measures of the motif while trying to sync perfectly with the audio file (and record audio of performance)
3. Repeat steps 1 and 2 for a wide range of tempos
4. Repeat steps 1–3 for each of the five motifs.

This procedure was performed under two conditions: (1) using the electromechanical prosthetic and (2) using the spring-loaded prosthetic.

7.2.6 Results

We analyzed the audio recordings to each of the rhythmic motif performances for the varying tempi. In order to evaluate synchronicity we used a dynamic time warping (DTW) algorithm. DTW is an algorithm that measures similarity between two time series. We compared the audio of Jason's playing with that of the computer generated audio track. The resulting cost of alignment calculated by DTW provides a distance-like metric. For two identical time series this distance is zero, but as the sequences vary the distance increases. Therefore, DTW can be used as a tool for measuring synchronization [25] (Fig. 7.5).

Synchronicity was measured for each of the rhythmic motifs for all the tempos. The normalized costs were averaged across the five motifs for each of the 12 tempi. The figure shows these averaged results from 90 to 210 bpm. *T*-tests were used to test for significant differences for each tempo and a Bonferroni correction of the *p*-values was performed to control for multiple comparisons. Table 7.1 shows the complete results.

As predicted the synchronization improved over a range of tempi using the electromechanical prosthetic. The absence of significant differences from 90 to 140 bpm is to be expected because these are speeds at which doubles strokes are unnecessary

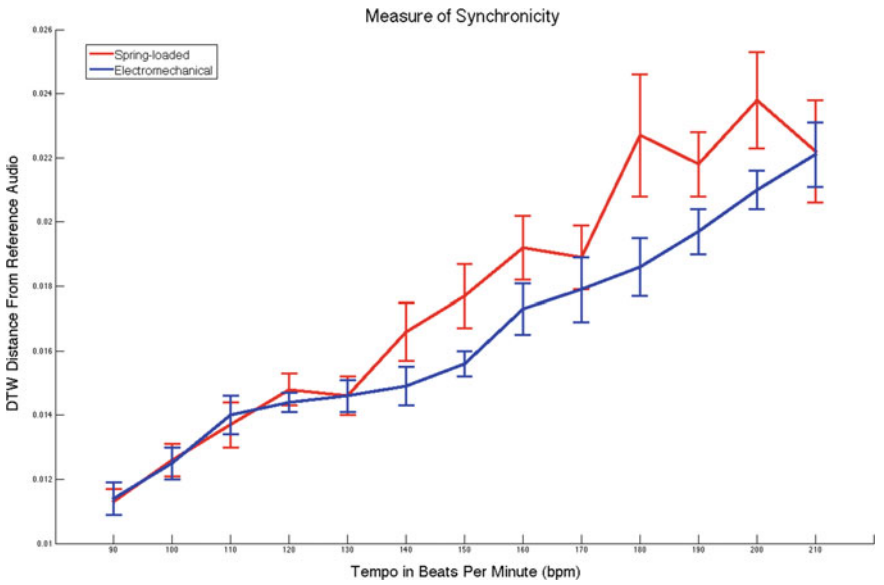


Fig. 7.5 A comparison of the ability to synchronize with a reference audio using the spring-loaded and electromechanical prostheses over a range of tempi. Dynamic time warping is used to measure the distance between the reference and recorded performance. A smaller DTW distance suggests the performance and reference are more closely aligned

Table 7.1 Average distances between the live performances using the spring-loaded and electromechanical prostheses and reference audio for each tempo based on a DTW metric. A *t*-test was used to test for significant differences and is indicated at the $p < 0.05$ level. A Bonferroni correction was used to control for multiple comparisons

BPM	Spring-loaded	Electromechanical	95% Confidence
90	0.0113	0.0114	
100	0.0126	0.0125	
110	0.0137	0.0140	
120	0.0148	0.0144	
130	0.0146	0.0146	
140	0.0166	0.0149	*
150	0.0177	0.0156	*
160	0.0192	0.0173	
170	0.0189	0.0179	*
180	0.0227	0.0186	*
190	0.0218	0.0197	*
200	0.0238	0.0210	*
210	0.0222	0.0221	

allowing Jason to use single strokes. In other words, the speed at which Jason can move his elbow up and down is sufficient to achieve decent synchronization.

At rates faster than 140 bpm double strokes become necessary to accurately play the pattern and this is where the performance of the electromechanical and spring-loaded prostheses begin to diverge. The decrease in synchronization remains roughly linear for the electromechanical prosthetic over the course of all the tempi. The increased performance at 210 bpm for the spring-loaded prosthetic is most likely due to the spring being naturally sufficient for double bounce intervals at that tempo. Though, as is evident, variable spring tension is necessary for sufficient double bounces of other intervals.

The results are promising and we believe more improvement can be expected. Jason had roughly 20 h of playing time with the electromechanical prosthetic over the course of four months prior to this study. This is compared to hundreds of hours of playing time over the course of two years with the spring loaded prosthetic. Additionally, this study required Jason to play only four measures, thus, not allowing for a decent evaluation of endurance. We predict that a better ability to perform these motifs for longer periods of time would also become apparent if measured, as double strokes allow drummers to conserve energy and prevent muscle fatigue.

7.2.7 The Second Stick

In addition to providing increased musical virtuosity for an amputee, the advent of a wearable electromechanical device also permits the exploration for other musical endeavors. One such endeavor is the notion of a wearable robotic musician. In this section we discuss the addition of a second stick to the prosthesis that behaves autonomously as if it were an individual agent. The second stick follows the same hardware blueprint of the primary stick, however, it is programmed to behave in a much different manner. We discuss the interesting implications on both human-robotic interaction and musical performance that such a design engenders by describing several behaviors developed for the agent (Fig. 7.6).

Robotics in music have primarily come in two forms: (1) independent and stand-alone agents [26] and (2) augmented instruments in which mechanics are mounted to traditional instruments resulting in increased or modified functionality [27, 28]. Human musicians interact with systems of the first form similarly to how human musicians interact with each other within an ensemble. There's a significant degree of independence, however, there is a shared and agreed upon musical context (such as tempo, time signature, key signature, and genre) that each musician has a responsibility to adhere to or at least address. Each musician may also have a specific role (such as keeping the beat, providing harmonic foundation, or improvising) that allows them to contribute to the higher-level musical product of the group.

The interaction between a person and an augmented robotic musical instrument may support standard playing technique or may require the person to approach the instrument in a non-traditional manner. Both paradigms present an interesting interaction model resulting from a shared acoustic interface. As the person performs on the instrument, the robotic components generate a response based on something it senses about the music, person, or environment. The robot may be programmed to behave similarly to that of stand-alone musical agents, however, the shared interface allows the person to influence the robot's final sonic result by physically manipulating certain aspects of the instrument (such as muting resonators). Likewise, the robot may similarly influence the person's sound and playing through analogous physical means.

Fig. 7.6 A second stick is included that has a "mind of its own" and creates rhythmic responses to Jason's and other musicians' playing

A wearable robotic musician extends the notion of a shared interface to that of a shared physical actuator and manipulator. Not only is the instrument shared, but also one or more of the DoFs necessary for actuating the instrument is shared. In the case of this prosthetic, the second stick can be used in two ways. It may behave similarly to the primary stick in that it bounces in a particular manner based on autonomously set PID settings. In this situation the user must trigger the initial onset. Subsequent onsets, resulting from the stick's bounce behavior, are controlled by the artificial intelligence (AI). The AI does not attempt to produce the desired effect expressed by the user, but rather an effect it computes to be appropriate based on what it senses about the music and the person's playing.

A second manner in which the robotic musician may behave is by generating its own strikes. In this scenario, the motor is used to both drive initial onsets (functioning like an elbow) and control the bounce behavior. However, the user's elbow DoF controls the distance between the sticks and the drum-head surface. This means the user has control over volume and whether contact is made with the drum at all. From an aesthetic and musical standpoint this scenario is much more interesting because the AI can generate rhythms and timbres that play a larger role on the final musical output than that of merely bounce behavior. This interaction paradigm is an example of haptic shared control in which the user and robot directly provide input to the music (see Fig. 7.7).

Several behaviors were designed for the autonomous secondary stick that enabled different interactive schemes between human and robot such as turn-taking and

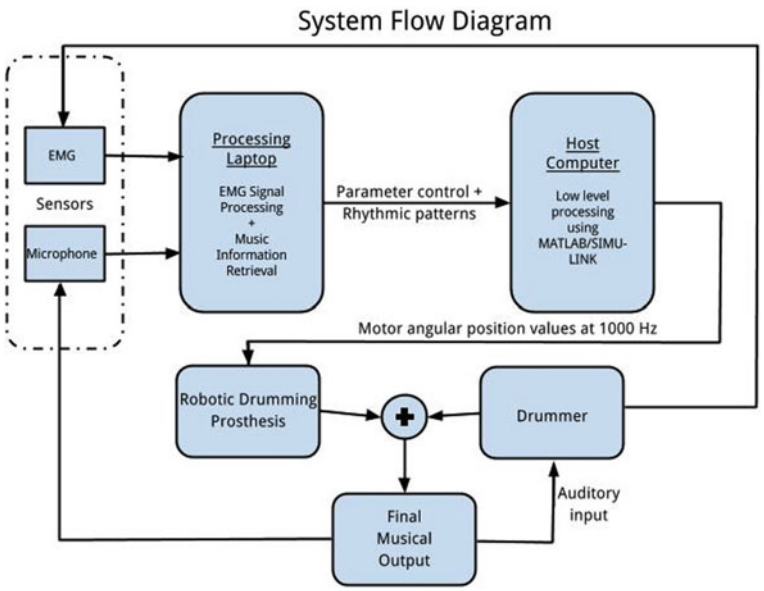


Fig. 7.7 System flow diagram supporting shared control paradigm for musical output

accompaniment. These behaviors were designed based on aesthetics and what we thought had the potential to create interesting musical results. These behaviors were implemented and showcased at various locations around the world. We invite readers to view a concert conducted at the Bailey Performance Center at Kennesaw State University as part of the Atlanta Science Festival featuring many of these behaviors at https://www.youtube.com/watch?v=dlSZCu5FAVM.

One behavior uses a note generative model based on rhythms provided by other musicians. The system analyzes the audio offline and extracts patterns based on rhythm and dynamics. Specifically, we used patterns extracted from excerpts of a percussion performance of renowned physicist, Richard Feynman. The secondary stick then generates these rhythmic sequences and the user can interact with them by layering the rhythms with accompanying sounds using his other limbs as well as providing alternative accent placements by manipulating distance between the electromechanical arm and drum surface.

A second behavior is based on inter-musician communication. The autonomous stick listens to the playing of other musicians in the group and generates responses according to their play. Specifically, the stick listens to the chords played by a guitarist and adjusts its speed based on the chords' tonal centers. The stick is capable of generating strikes at a rate of up to 20 Hz. When using such high speeds the acoustic response perceived by the ear is more timbral in nature rather than rhythmic (akin to different sounding buzz rolls). As the guitarist changes chords the strike frequency changes within the range such that the acoustic timbre is manipulated.

A third behavior uses EMG and allows the user to control higher-level parameters regarding the stick's musical response. One higher-level parameter we explored is note density (the number of onsets within a time frame). In this behavior, the user provides a "seed" rhythm by flexing his muscle rhythmically. The system uses the onset detection method described earlier to learn the temporal sequence and quantizes it based on the desired tempo. The robot then stochastically adds additional notes to the rhythm to increase note density based on the user's control. In order to control the note density the user flexes continuously in which a stronger amplitude is mapped to a higher note density.

Robotic musicianship behaviors such as these are difficult to evaluate objectively. Often, a more subjective overview of the challenges and procedures used to create interesting behaviors is more helpful for those developing generative algorithms. As such, we have empirically created a list of considerations we believe can be useful for other designers developing such a wearable robotic musician paradigm based on Jason's and other musician's experience of performing and interacting with the system.

Do not replace the human drummer It is important that the robotic musician does not cause the human drummer to relinquish the role of his arm entirely to the AI. Rather, design algorithms that supplement or complement the drummer's play.

Consider the natural abilities of the drummer Generative algorithms based on outputs that are outside the reach of the drummer are preferred. The stick's ability of moving at speeds greatly exceeding that of natural human ability and performing

complex rhythms accurately should be leveraged. For example, allowing the robot to perform the complex Feynman rhythms enabled Jason to use his other limbs to create a richer musical outcome.

Use the robot sparingly It is important to find the appropriate balance such that the drummer does not find the robotic musician to be an interference, but rather an inspiration to his and other musician's playing in the group.

Consider cognitive load on the drummer If the drummer is controlling aspects of the robot's behaviors the mappings should be based on higher-level musical parameters (such as note density).

These are a few considerations we found necessary to address in working with Jason and the prosthetic. However, we have only begun to explore the possibilities of wearable robotic musicians and believe there is great potential for further research and development in this area.

7.2.8 Conclusion

In this section we presented a drumming prosthesis for a musician with a transradial amputation. The mechanical design and underlying computational architecture created an effective simulation of finger and grip function during drumming applications. The system was validated qualitatively through discussions with the user and objectively through a user study involving a musical synchronization task.

Developing prostheses that enable people to play musical instruments is very challenging because the necessary level of dexterity, control, and feel afforded to the user is enormous. A perfect replication of natural wrist and hand anatomy is ideal, however, alternative methods can be leveraged to circumvent the obvious difficulties of achieving this. A shared control framework is helpful because it allows the user to achieve responses and behaviors in the device without having to cognitively address each individual parameter or DoF. Instead, the user is able to provide a higher level input, which the robot interprets and uses to manipulate parameters in order to generate the desired response.

The prospect of a wearable robotic musician was also discussed in this section. We explored different interactive scenarios in which musicians perform with the robot enabling shared control over the final artistic output. Our findings are not only useful for composers and musicians, but also for researchers developing wearable artificial supernumerary limbs. Understanding how one interacts with a wearable autonomous agent is important for optimizing coordination tasks and user experience. Supernumerary limbs have the potential to augment human anatomy in several ways. This may include the ability to surmount physical limitations or, as in our case, to encourage creativity and decision processes by presenting the user with new ideas, interpretations, or solutions.

7.3 The Third Drumming Arm

7.3.1 Introduction

While the Robotic Drumming Prosthetic Arm allowed an amputee to retrieve key elements of the musical expression that had been lost, it also enhanced his musical skills with new mechanical abilities, supporting unmatched levels of speed and virtuosity. In an effort to extend this research to a wider user base, we have developed a supernumerary robotic limb (SRL) titled the Third Drumming Arm (TDA), which was designed to be used by able-bodied drummers. The goal of the TDA was to augment musician physical abilities by providing humanly unmatched power, speed, and dexterity. Here too, one of our main challenges was to bring robotic musician's capabilities into the human body in a comfortable and easy to use manner, that will enhance, rather than impair musical expression.

Most SRLs that are not designed for playing music have passive compliance that can result in unwanted oscillations that exert cyclic loads on the user's body and could negatively affect the physical comfort of the user. Moreover, drum playing can lead to unpredictability of the vibrations created, which adds cognitive load on the user and negatively affects the user's musical expression and creativity.

The goal of this project was to address these challenges by utilizing a novel computationally inexpensive input-shaping method, allowing us to study the concept of augmentation and shared control in musical human-robot interaction. To achieve this goal we designed a TDA that can be attached to a drummer's right shoulder, allowing them to control an additional drum stick (see Fig. 7.8). We used robustness analysis to test the effectiveness of the input-shaping method in controlling the residual vibrations of the arm. In addition, we conducted user studies, which demonstrated that our method increased user satisfaction and comfort.

7.3.2 Related Work

In recent years, wearable robotics and Supernumerary Robotic Limbs (SRLs) have received growing interest from the research community. Unlike robotic exoskeletons, SRLs are kinetically independent from human movement, allowing the limb to be actuated even when the human limb is stationary. SRLs vary in size, shape, material, and the control method. Since smaller SRLs, such as [29], have low inertia that requires less actuator power, they do not tend to suffer from excessive vibration that interferes with their operation. In larger SRLs, like the one we developed for this project, the vibrations, pressure on the skin, and noise often lead to a negative effect on user satisfaction and comfort [30–36]. Various measurement methods have been provided to asses the operation of SRLs [37–39]. In this project we chose to use the

Fig. 7.8 The third drumming arm (TDA) platform

Quebec User Evaluation of Satisfaction with Assistive Technology (QUEST) [40] and the Locally Experienced Discomfort (LED) evaluation tool [41], which have been commonly used as tools for measuring the subjective perception of a wearable device [30].

7.3.3 Motivation

This work focused on addressing the problem of residual vibrations, observable in an early prototype of The TDA developed at GTCMT. This problem led users to pause their playing frequently, maintain constant visual link to the arm in an effort to understand its behavior, and spend a significant part of their cognitive load to predict the arm's trajectory and final position, often leading to failure to play along with the rhythm properly. To address this problem, we have studied the nature of these vibrations, which informed an input-shaping approach designed to control them. Unlike previous related input-shaping efforts, we also used a modified and simplified version of QUEST to assess the performance of our approach.

In the next sections, we describe TDA platform, discuss the design and implementation of the input-shaper method that was used to minimize the residual vibrations of the arm, and describe the user studies that were performed to evaluate the performance of the controller regarding comfort criteria.

7.3.4 Design

The TDA is a four DOF robotic arm that can be attached to the human body (see Fig. 7.8). To provide user comfort and to accommodate different body sizes, the arm's shoulder attachment socket is made up of a layer of ABS plastic and a layer of soft foam. The top degree of freedom (DOF)—the shoulder joint—rotates the arm around the body (horizontal abduction/adduction). The second DOF connects the shoulder joint to the elbow joint and performs a function similar to a human elbow (flexion/extension). The third DOF performs rotation of the wrist (supination/pronation), while the fourth DOF is designed to actuate a drumstick to hit the drum surface. We use Dynamixel MX-64 servomotors[1] to actuate the arm, and Max/MSP[2] to send musical commands to the motors using an Arduino Mega2560[3] board.

7.3.5 Dynamics Modeling

Since the movements of the second and third DOFs do not result in considerable residual vibrations, we focused on modeling the movements of the shoulder joint. To simplify modeling, we assume that the second motor is positioned in such a way that link 1 and link 2 are co-linear, as in Fig. 7.8, marking a worst-case scenario in which maximum residual vibration occurs. Due to the musical demand for accurate time, the arm has to react to the positioning commands as fast as possible. Therefore we set a maximum actuator effort in order to rotate a distance of θ in minimum time t. After the arm stops moving at the end of its travel distance, it continues to vibrate due to the elasticity present in the attachment to the body.

As a first step, we decided to derive the system model using experimental approaches rather than analytic or numerical methods. This allowed us to observe and learn unknown parameters such as the elastic constants and damping ratios of the human tissue, the shoulder mount material, and the robot material. In an effort to derive a physical model of the arm and its vibrations, we used a 9 DOF inertial measurement unit (IMU) mounted at the end of link 2 to record and model movement parameters based on the actual response. The angular position of the arm was recorded after its initial displacement.

The black solid lines in Fig. 7.9 show that the vibration characteristics of the system due to an initial displacement closely match the response of a simple harmonic oscillator with elastic constant of K_T and damping coefficient of ζ.

The equation of motion of this system is described as:

$$J\ddot{\theta} + B_T\dot{\theta} + K_T\theta = \tau(t) \tag{7.1}$$

[1] http://www.robotis.com/xe/dynamixel_en.

[2] https://cycling74.com/.

[3] https://www.arduino.cc/en/Main/ArduinoBoardMega2560.

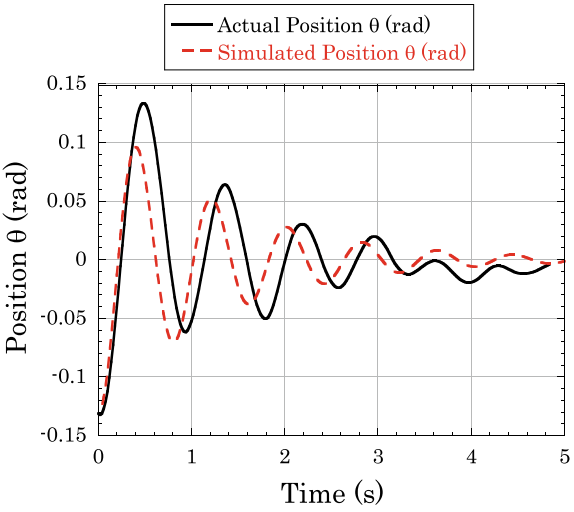


Fig. 7.9 Actual (recorded) and simulated response of the manipulator to an initial displacement

where:

- J is the rotational inertia (kgm^2)
- K_T is the torsional spring stiffness (Nm/rad)
- B_T is the torsional damping constant (Nms/rad)
- $\tau(t)$ is the input torque (Nm)
- θ is the robot arm's rotational displacement (rad).

Equation 7.1 can be normalized into the following equation:

$$\ddot{\theta} + 2\zeta\omega_n\dot{\theta} + \omega_n^2\theta = \omega_n^2 u(t) \tag{7.2}$$

where:

- ζ is the damping ratio
- ω_n is the natural frequency rad/s
- $u(t)$ is the input signal rad.

Similar behavior can be observed from the response of the system to a ramp position input. This is shown as the black solid line in Fig. 7.10.

The damped frequency of oscillations rad/s is calculated using:

$$\omega_d = \frac{2\pi}{T} \tag{7.3}$$

where T is the time needed to complete one period of oscillation. Because the damping ratio is relatively small, it can be calculated using logarithmic decrement:

$$\zeta = \frac{\ln \frac{x_0}{x_1}}{2\pi} \tag{7.4}$$

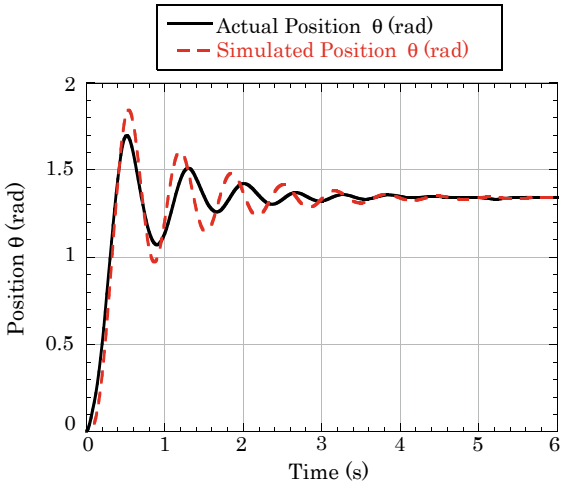


Fig. 7.10 Actual (recorded) and simulated response of the manipulator to a ramp position input

Table 7.2 Damping ratio and frequency of the system

Response type	Calculated parameters		
	ω_d (rad/s)	ζ	ω_n (rad/s)
Free vibration response	7.78	0.098	7.82
Forced vibration response	9.58	0.094	9.62

where x_0 and x_1 are two successive peaks extracted from the graphs. The natural frequency of oscillation can be calculated by:

$$\omega_n = \frac{\omega_d}{\sqrt{(1-\zeta^2)}} \tag{7.5}$$

Damping ratio and natural frequency are calculated using logarithmic decrements. Both the free and forced responses were recorded five times to improve the approximation of the actual values. The parameters were calculated from all graphs and the average values were found. Results are summarized in Table 7.2.

In both cases, the small amount of damping in the system lead to a small difference between damped and natural frequencies. The oscillation frequencies in the forced vibrations case were greater due to the inner PID feedback controller, which controlled the position of the motor, comprising only a proportional parameter. The effectiveness of the model is demonstrated by considering the relative similarity of the simulated responses to the experimental responses that are shown in Figs. 7.9 and 7.10 with red dashed lines.

7.3.6 *Input-Shaper*

An input shaper is a sequence of convolved impulses that can limit the residual vibrations in a system. We chose to use a zero vibration derivative (ZVD) input shaper for an input shaper since it is robust to disturbances and modeling errors, and easy to implement. We calculated the ZVD shaper based on system parameters: $\omega_n = 9.62$ rad/s and $\zeta = 0.1$.

Informed by [42], a ZVD shaper consisted of three impulses, using the following amplitudes and application times:

$$\begin{bmatrix} A_i \\ t_i \end{bmatrix} = \begin{bmatrix} \frac{1}{(1+k)^2} & \frac{2k}{(1+k)^2} & \frac{k^2}{(1+k)^2} \\ 0 & \frac{\pi}{n} & \frac{2\pi}{n} \end{bmatrix} \tag{7.6}$$

where $k = e^{\frac{-\zeta\pi}{\sqrt{(1-\zeta^2)}}}$. Thus, substituting numerical values in (7.6), the shaper is expressed by:

$$\begin{bmatrix} A_i \\ t_i(s) \end{bmatrix} = \begin{bmatrix} 0.3344 & 0.4877 & 0.1778 \\ 0 & 0.3266 & 0.6531 \end{bmatrix} \tag{7.7}$$

The result of convolving this shaper with the original input command consisting of a ramp input of 1.45 rad (solid black line in Fig. 7.11) is shown in Fig. 7.11 with red dashed line.

Figure 7.12 shows the simulated system response to the unshaped and shaped commands with solid black line and red dashed line respectively. It can be seen that the ZVD input shaper is capable of canceling the residual vibrations completely, but

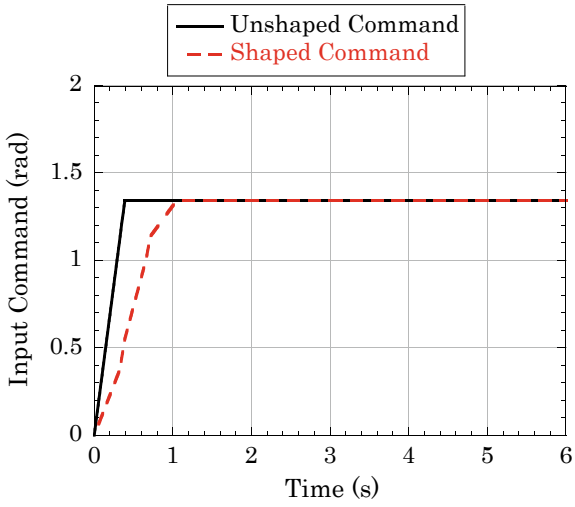


Fig. 7.11 Unshaped and shaped commands used as input to the system

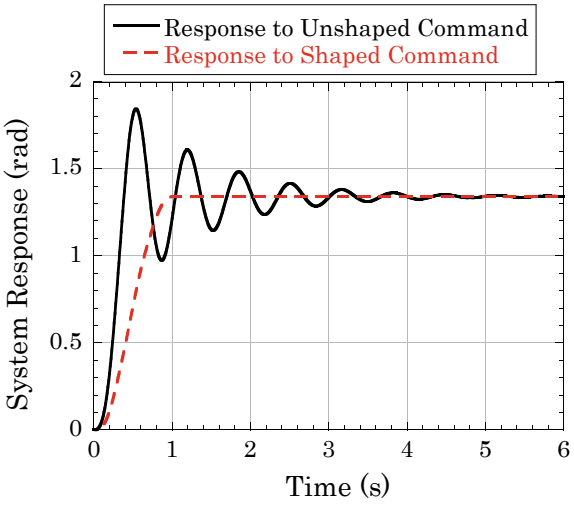


Fig. 7.12 Simulated unshaped and shaped responses of the system to a ramp input

Table 7.3 Timing and travel angle of each segment in shaped and unshaped commands

Segment	Time delay (s)	θ_i (rad)	θ_f (rad)	Velocity (rad/s)
Unshaped command				
1	0	0	1.34	3.45
Shaped command				
1	0	0	0.37	1.154
2	0.32	0.37	0.57	2.84
3	0.4	0.57	0.96	1.68
4	0.62	0.96	1.15	2.3
5	0.73	1.15	1.34	0.62

introduced a time delay of 0.65 s. In an effort to obtain the actual system response to the shaped commands, we used a micro-controller that divided the original ramp motion profile into five segments. Each of these segments had a starting and an ending position and a specified speed, as illustrated in Table 7.3. Figure 7.13 demonstrates that the ZVD input shaper was capable of reducing the maximum overshoot to 0.82%, thus significantly minimizing residual vibrations. Here too, this led to a time delay penalty of 0.6 s, which should be accounted for when designing the trajectory of the robot arm while it is playing music.

The input shaper robustness is an important criterion ensuring that the shaper works properly for a wide range of conditions. Two special cases were investigated; first the designed input shaper was tested on different subjects. Every subject had a different arm circumference and a different tissue elasticity. The response of the system for these subjects is shown in Fig. 7.14. Then, the designed input shaper was

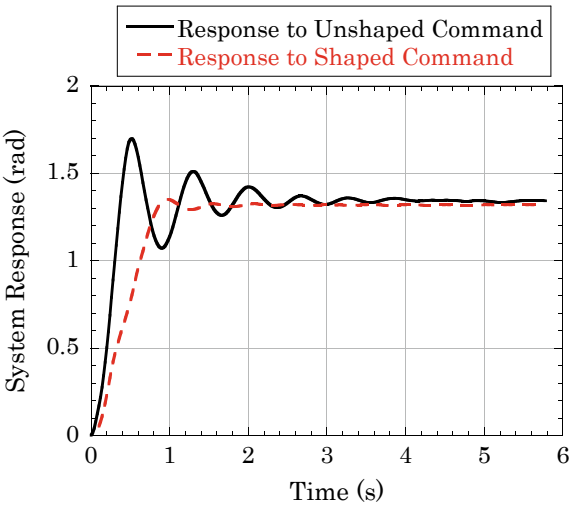


Fig. 7.13 Actual (recorded) response of the system to a ramp input

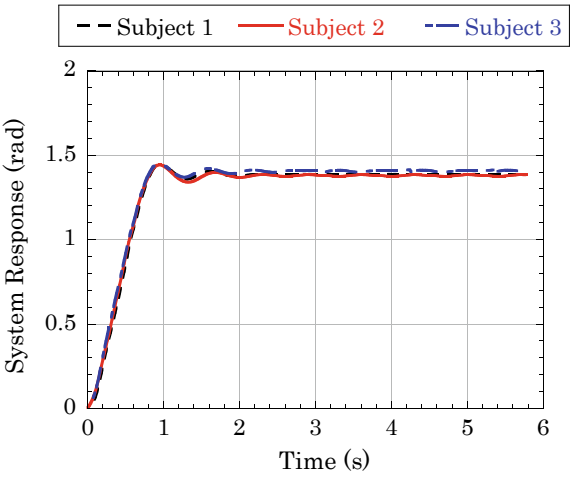


Fig. 7.14 System ramp response for 3 different users wearing the arm

used to move the robot arm to a different final desired position (an angular distance of 1 rad) for a single user. The system response is shown in Fig. 7.15. The data shown in Figs. 7.14 and 7.15 demonstrates that the designed input shaper is robust and can be reliably used under different conditions.

7.3.7 User Survey

Fourteen subjects were selected through email advertisements and snowball sampling, consisting of 57.1% male, 64.2% between 23–29 years old, 85.7% who played

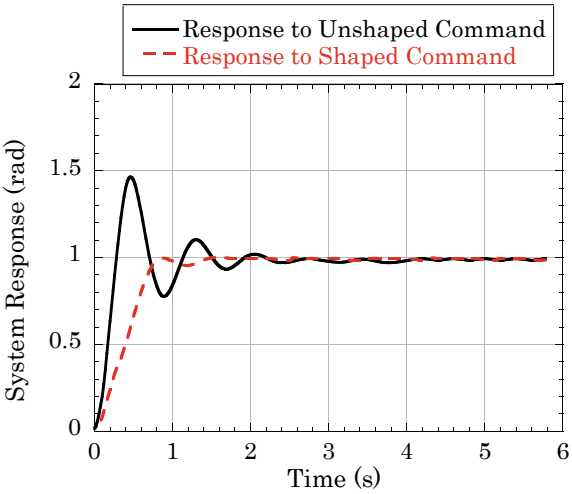


Fig. 7.15 System response to a move distance of 1 rad

at least one musical instrument and 57.1% who were music technology students. Studies were performed on each participant individually after they completed the IRB approved consent form and demographics questionnaire.

The circumference and length of the bicep of each participants were measured and noted on the questionnaire. This was used to observe if there was any correlation between the socket size and the perceived comfort of the TDA movements. After introducing the participants to the TDA, it was placed on their shoulder. Three different scenarios were studied. In the first scenario, the arm was kept stationary and the level of comfort with the socket was studied. In the second scenario, the TDA was moved using unshaped commands. The third scenario tested movement using the input shaped commands. In all the scenarios, the users were sitting stationary on a chair in a comfortable position and observed the movement of the robotic arm.

A simplified version of the QUEST test was administered in the form of an interview to obtain the perceived comfort of using the robotic arm. We used 8 of the 12 QUEST satisfaction questions that were related to weight, dimensions and comfort. Each question was rated from 1 to 5. Four of the QUEST items that address assistive devices were not relevant to the TDA, and therefore were not used.

Figure 7.16 shows the average level of comfort of the users in three different scenarios based on their arm circumference. The users with arm circumference of 10–11 inches expressed highest comfort. This is due to the fact that the size of the user's arm matched the socket size used for the shoulder attachment. It is also observed that regardless of the arm circumference, the comfort achieved from the shaped command is always higher.

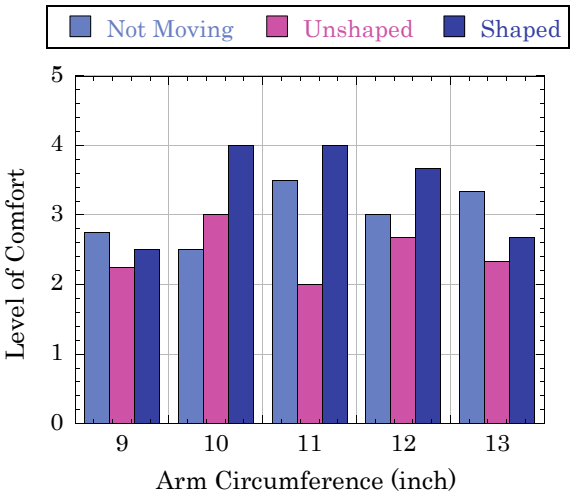


Fig. 7.16 Average user comfort level for arm being stationary, as well as moving with shaped and unshaped commands

7.3.8 Conclusion

Our Experiments with the TDA platform have shown that a wide variety of criteria should be taken into account while designing supernumerary robotic limbs compared to exoskeleton designs or other robotic manipulators. In time demanding musical scenarios, residual vibrations might be a concern in low impedance robotic limbs including the SRLs. We have shown that the conventional methods for suppressing the residual vibrations in the structures such as cranes and robotic manipulators can be applied to SRLs. However, these methods might not increase the user satisfaction and comfort, which are crucial for musical playing. Future work may include comparing other methods of vibration control such as model predictive control in addition to input shaping to find out the best control strategy based on the level of comfort of the users with these methods.

7.4 The Skywalker Piano Hand

The Skywalker Piano hand (see Fig. 7.17) was designed to provide amputees with dexterous and detailed operation of a prosthetic hand by supporting finger-by-finger control at a level that can allow playing musical instruments such as the piano. The project was designed to improve upon the EMG sensors that were used in the Robotic Drumming Prosthetic Arm, which could only respond to binary muscle contraction and retraction. To achieve more dexterous and detailed control we developed a new sensing and modeling technology, using a wearable ultrasonic sensor and machine learning to model discrete and continuous finger gestures based on training data from forearm muscle movements.

Fig. 7.17 The Skywalker Piano Hand

7.4.1 Related Work

State of the art classification algorithms for robotic prosthetic can reach accuracy of above 95% when detecting 5–10 gestures such as pointing, closing, and opening hands [43, 44] and around 70% when detecting up to 50 gestures [45]. However, these classification methods are limited to gross discrete and sequential operation and cannot capture fine simultaneous and continuous operation of multiple fingers that are needed for subtle and detailed digit operation, required for expressive musical playing [46]. While some success using machine learning to simulate control of multiple fingers has been made with implantable EMG electrodes [47], using similar methods with surface EMG electrodes has been limited to a single finger and failed to achieve consistent and robust results [48, 49]. We have identified only one recent study that has been successful in providing a limited level of low latency continuous and simultaneous detection of finger movements using a shallow neural network and Gaussian regression [50]. However, no previous EMG work has been successful in modeling accurate continuous and simultaneous muscle movement in a manner that would allow dexterous and subtle robotic prosthetic operation for fine control tasks. We hypothesize that the failure of current EMG systems to accurately model continuous and simultaneous digit movement patterns is attributed to the noisy nature (e.g. cross talk) and low signal-to-noise ratio of EMG signals that does not allow for accurate prediction of muscle movement in relationship to digit gestures.

To address these problems, some researchers have explored ultrasound as an alternative sensing technique that may provide more accurate and less noisy signals for pattern detection. The mechanical information embedded in an ultrasound image has been demonstrated to be useful in human-machine interfaces, as morphological

changes in forearm muscles can be used to predict behavior of the fingers and wrist. In [51] the analysis of hand-selected portions of an ultrasound B-mode image led to successful prediction of wrist flexion. Castellini and Passig showed that there is a linear relationship between features gleaned from images taken of the forearm and hand kinematics [52]. The features were derived from visual inspection of the ultrasound images and were designed to describe localized image deformations. Subsequently, a ridge regression was successfully used to learn a model that uses these features to predict finger positions [53], although the researchers achieved unsatisfactory performance for predicting finger force [54].

Unlike these studies, which use ultrasound imaging as input for the models, some researchers have been experimenting with single element transducers that are not used to create images, but rather, utilize the echoes provided by the ultrasound beam directly. In [55] data extracted from an A-mode ultrasound signal was used to control a one degree of freedom (wrist extension) powered prosthesis in real-time. There, a linear regression was used to create a model mapping muscle deformation to DoF position and demonstrated an improvement over EMG [56]. Recently, a band of four single element transducers was developed to wrap the forearm [57]. In [58], a background subtraction method was used (in which a resting state served as the background) in coalescence with k-nearest neighbor (kNN) to predict individual finger flexion. Similarly, kNN was used to classify four different finger movements in [59]. We build on these efforts by modeling continuous and simultaneous finger movements and creating a low latency system that is responsive and invariant to forearm rotations.

7.4.2 Goals

Building on the related work and our interest to improve the control and expression for wearable robotic musicians, we defined two main goals for this project:

1. Conduct physiological experiments to identify the optimal location for a wearable ultrasonic sensor on the arm to improve control and expression.
2. Design machine learning algorithms that would allow for accurate low latency prediction of continuous finger gestures based on training data from forearm muscle movements that could support expressive musical instrument playing.

7.4.3 Ultrasound Configuration Experiments

7.4.3.1 Method

In order to understand the capabilities and limitations of ultrasound technology for the Skywalker Piano Hand project we first aimed to identify the optimal placement

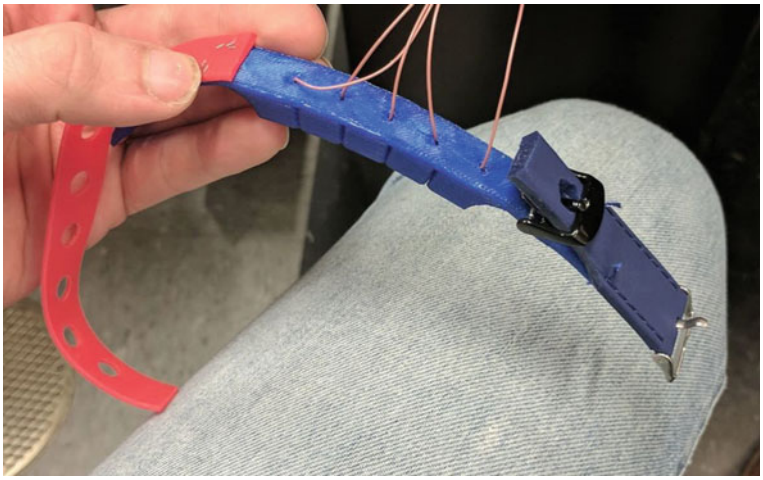


Fig. 7.18 SMG arm band

of the sensor on the arm. To address this goal we built a sensor array, consisting of five small and inexpensive individual ultrasound elements mounted on a bendable plastic band with a belt buck that allowed the band to be tightened onto any part of the arm (see Fig. 7.18).

The sensor band was connected to the medical-grade ultrasound machine *Ultrasonix SonixTOUCH*, which could pulse and receive signals from both standard medical imaging probes and non-standard single element ultrasound transducers. Each of the single piezo tracducers in the band were connected to a single channel on the *Ultrasonix* and pulsed sequentially at a sampling rate of 40 MHz. The single element transducers were rated with a bandwidth between 2 and 8 MHz, centered on 5 MHz.

Data for our experiments were collected from arm movements of three Masters student participants. Participant 1 collected data twice, on two separate days, to evaluate whether the results would remain constant over time. The collection mechanism, pictured in 7.19 consisted of five force-sensing resistors calibrated to detect force within a reasonable range for the fingertips designed to provide ground-truth. In order to test which locations provided the best results, we collected data from 22 locations on the extensor side of the subjects' arms as well as 22 locations on the flexor side. Figure 7.19 depicts the collection process for the extensor side.

In order to control for variation between different users, each participant was asked to follow an identical gesture sequence with varying complexity at each collection location, see 7.21. The vertical space between each collection line was determined based on an ideal average forearm length [60]. Data was collected twice at each location (adding up to one minute of data at each location) so that time-seperated training and testing sets could be generated (Fig. 7.20).

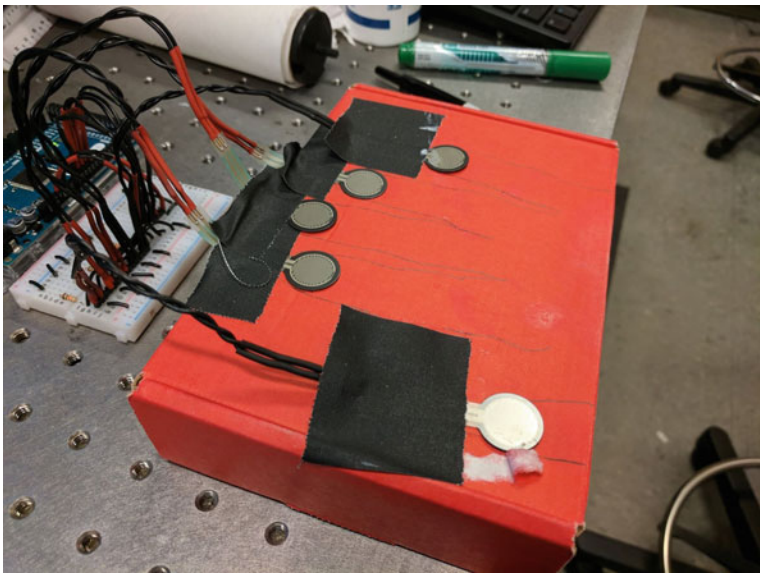


Fig. 7.19 Force sensing resistor collection system

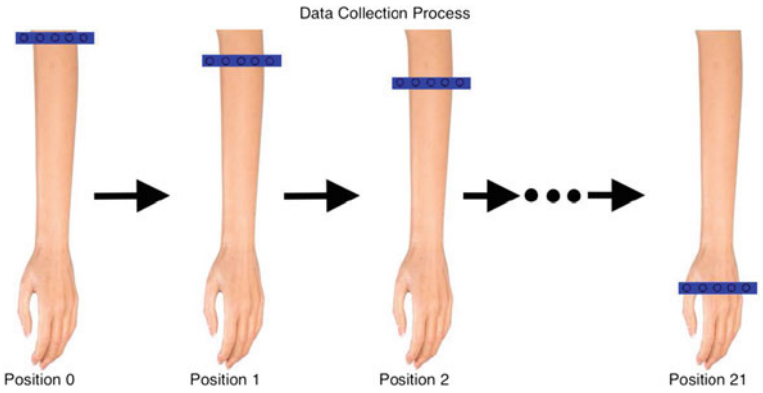


Fig. 7.20 Data collection process for extensor side

7.4.3.2 Configuration Experiment Results

The data was grouped into horizontal and vertical arrays to test which locations produced the most useful result. Figure 7.22 depicts the array design placements. In order to test each candidate array, we extracted a menagerie of different features from each array and used a Support Vector Machine to classify each hand gesture. Six distinct features were tested in an effort to evaluate the optimal locations:

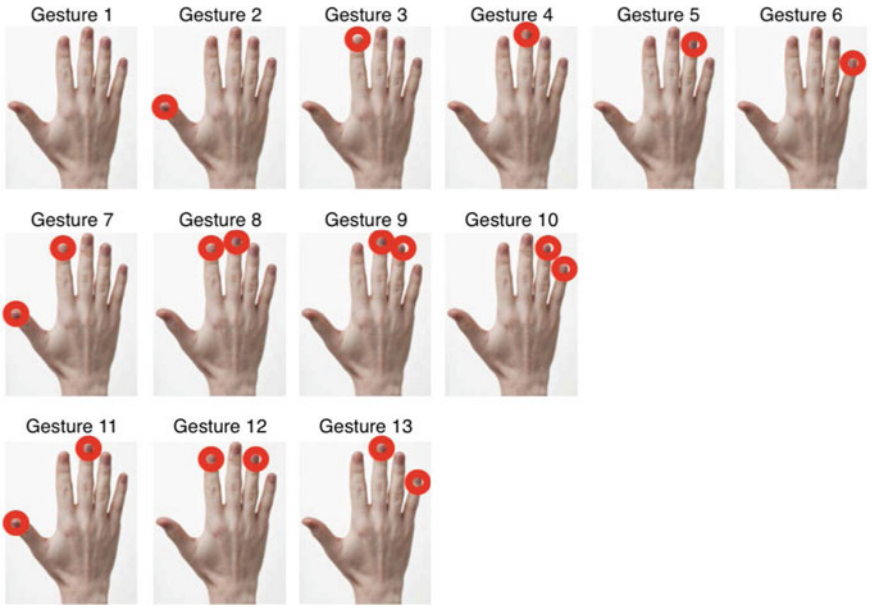


Fig. 7.21 Gestures tested. The red circles indicate that the finger is pressing down

1. Median filtered signal of various window sizes
2. Mean filtered signal of various window sizes
3. Locations of salient peaks
4. Peak heights
5. Average inter-peak distance
6. Feature extracted from a convolutional neural network trained on a large body of existing ultrasound data.

The Support Vector Machine classifier was chosen from Python's scikit-learn module. The gamma value was varied between 0.1 and 0.000001. Figure 7.23 shows an example of classification accuracy measure at one candidate array location for each of the three feature/model pipelines.

In order to test if there was correlation between optimal configurations across users, trials, and feature/model configurations, we calculated a Pearson correlation coefficient between each user and feature/model configuration pairwise. The results from performing this test are presented in Fig. 7.24. We also performed the same test between two separate trials as depicted in Fig. 7.25. Figure 7.26 shows an example of the confusion matrix produced by optimal configuration for one of the participant.

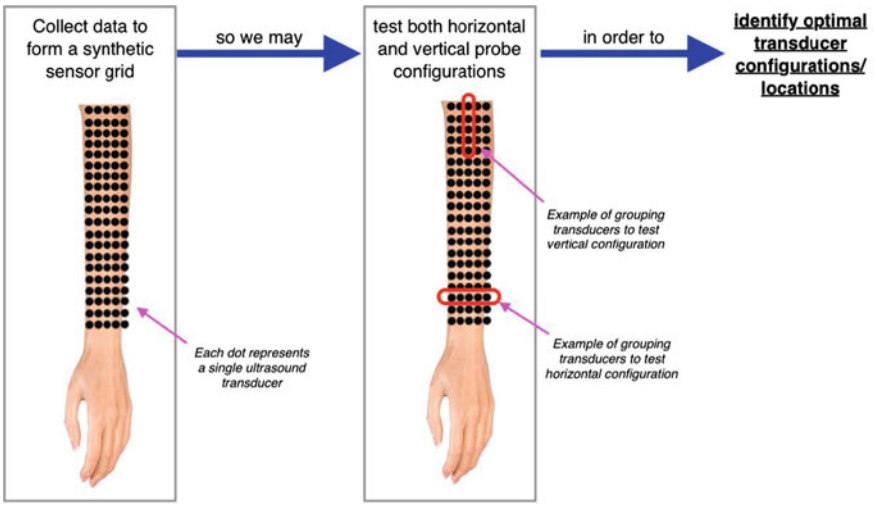


Fig. 7.22 Experimental procedure and objectives

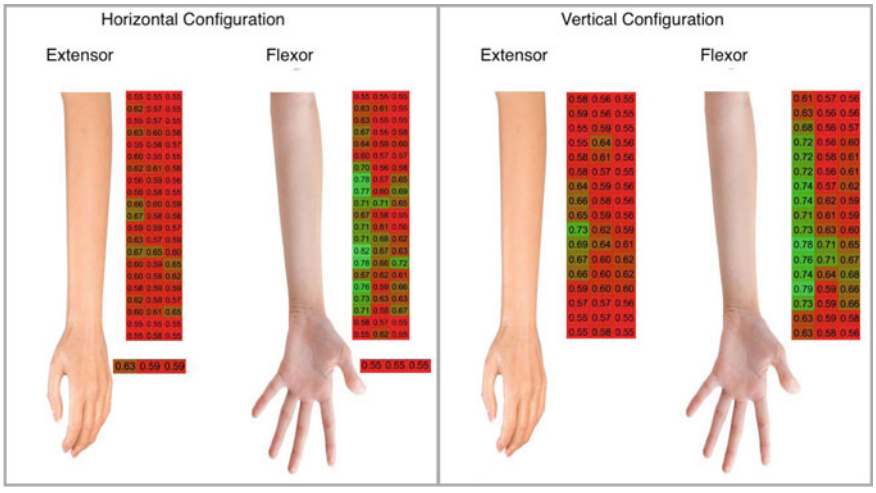


Fig. 7.23 Classification accuracy for participant 1, trial 1

7.4.3.3 Discussion

A few observations are suggested based on our results:

1. Better performance was achieved on the flexor side rather than the extensor side.
2. Better results were recorded in a location slightly past halfway down the arm towards the palm.
3. There is clear correlation within users between different feature/model pipelines.

	person1_features1	person1_features2	person1_features3	person2_features1	person2_features2	person2_features3	person3_features1	person3_features2	person3_features3
person1_features1	1.00	0.48	0.75	0.46	0.46	0.38	0.40	0.13	0.09
person1_features2	0.48	1.00	0.54	0.43	0.39	0.40	0.44	0.20	0.03
person1_features3	0.75	0.54	1.00	0.29	0.30	0.18	0.30	0.04	0.12
person2_features1	0.46	0.43	0.29	1.00	0.71	0.78	0.47	0.19	0.15
person2_features2	0.46	0.39	0.30	0.71	1.00	0.86	0.42	0.07	-0.06
person2_features3	0.38	0.40	0.18	0.78	0.86	1.00	0.39	0.12	-0.01
person3_features1	0.40	0.44	0.30	0.47	0.42	0.39	1.00	0.56	0.45
person3_features2	0.13	0.20	0.04	0.19	0.07	0.12	0.56	1.00	0.39
person3_features3	0.09	0.03	0.12	0.15	-0.06	-0.01	0.45	0.39	1.00

Fig. 7.24 Correlation of between classification results of different participants

	trial1_features1	trial1_features2	trial1_features3	trial2_features1	trial2_features2	trial2_features3
trial1_features1	1.00	0.48	0.75	0.60	0.44	0.37
trial1_features2	0.48	1.00	0.54	0.35	0.30	0.27
trial1_features3	0.75	0.54	1.00	0.61	0.55	0.42
trial2_features1	0.60	0.35	0.61	1.00	0.51	0.57
trial2_features2	0.44	0.30	0.55	0.51	1.00	0.59
trial2_features3	0.37	0.27	0.42	0.57	0.59	1.00

Fig. 7.25 Correlation of between classification results of different trials for participant 1

4. The correlation between participants 1 and 2 was stronger, and less so with participant 3.
5. There is some correlation between trials for subject one, but this correlation is less than the one observed between participants 1 and 2.

Finding 1 was expected—since we were attempting to predict which fingers are contracted, it was reasonable to assume that we would find more valuable information when observing the flexor side rather than the extensor side. Finding 4 may be explained by the fact that participant 3’s data produced less accurate results than that of participants 1 or 2. Finding 5 was unanticipated, and might be explained by errors in the data collection process. The results show that the best-case classifiers performed well for participants 1 and 2. However, even the best-case classifier performed quite poorly for participant 3. This finding indicates that data collected from more participants would be needed going forward to produce more generalized knowledge.

Gesture number	1	2	3	4	5	6	7	8	9	10	11	12	13
1	"512"	0	0	0	0	0	0	0	0	0	0	0	0
2	3	32	0	0	0	0	0	0	0	0	0	0	0
3	0	0	35	0	0	0	0	0	0	0	0	0	0
4	0	0	0	35	0	0	0	0	0	0	0	0	0
5	35	0	0	0	0	0	0	0	0	0	0	0	0
6	35	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	11	0	21	0	0	3	0
8	24	0	0	0	0	0	0	7	0	3	0	0	0
9	0	0	0	0	0	0	0	0	24	0	0	11	0
10	0	0	0	0	0	0	0	0	0	35	0	0	0
11	0	0	0	0	0	0	0	0	0	0	35	0	0
12	19	0	0	0	0	0	0	0	0	0	0	16	0
13	0	0	0	0	16	0	0	0	0	0	0	0	19

Fig. 7.26 Confusion matrix for optimal configuration for one participant

7.4.4 Machine Learning Design

7.4.4.1 Motivation

After identifying the optimal sensor location for detecting discrete finger movements using a classifier, we aimed to extend our research to regress and track continuous finger movements, which could allow for more expressive and intuitive control when playing musical instruments. Our goal was to use machine learning to extract useful features from muscle movement ultrasound images for describing continuous and simultaneous finger movements in a manner that would minimize training time for individual users. To achieve this goal we decided to use a commercial ultrasound probe that images the muscle using 32 single elements rather than the 5 single elements band described in the previous section (see Fig. 7.27).

7.4.4.2 Method

We used pre-trained Convolutional Neural Networks (CNNs), as they have shown success in imaging based diagnostic tasks such as in [55, 61]. Informed by techniques used in these works, we implemented various preprocessing steps including denoising, binarization, and edge detection to optimize the feature map for learning (see

Fig. 7.27 Machine learning training using the Sonostar 32 Channel ultrasound probe

Fig. 7.28). In order to generalize the system and to avoid collecting many hours of labeled data for individual subjects, we utilize the data and models already acquired in the pre-training phase. This allowed us to obtain useful features capable of not only discriminating between a finite set of finger configurations, but also estimating the semantic relationship among gestures such that continuous and dexterous control would be possible.

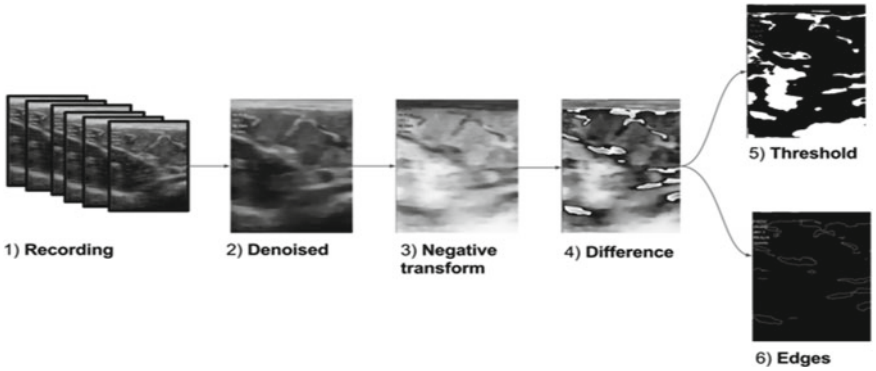


Fig. 7.28 Ultrasound image preprocessing

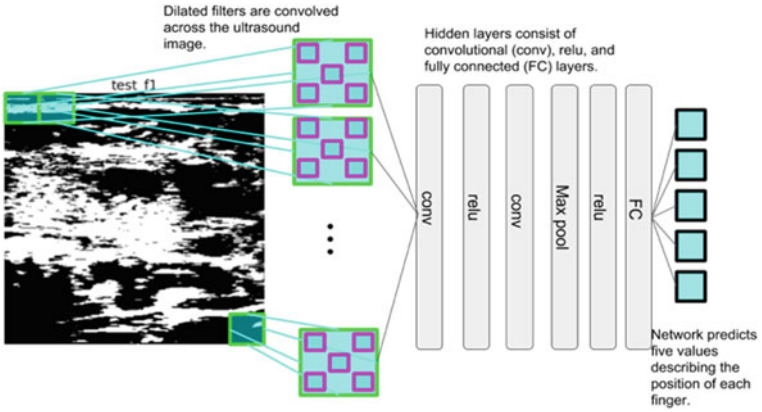


Fig. 7.29 Ultrasound B-mode images will be fed into a convolutional neural network. Each image will be labeled with values representing finger joint angles. The network will learn the relationship between muscle image features and finger positions

We extended our one-layer network to a deeper CNN in an effort to extract a better hierarchical representation of the ultrasonic image. Such networks have proven to be effective in computer vision tasks and for learning kernels that filter the images to extract meaningful features [62, 62]. Because CNNs can learn useful feature maps provided there is sufficient data, we trained our system on a large data set of images with input vectors consisting of raw pixel values. We used this data to pre-train the network and then fine-tune the parameters for individual users, a method that has been successful in related work [63–65]. In order to obtain an accurate semantic segmentation of the image we used dilated convolutions, which have shown to be useful in visual semantic segmentation tasks [66]. Such dilated convolutions allowed for the filter to increase the receptive field without sacrificing resolution (see 7.29).

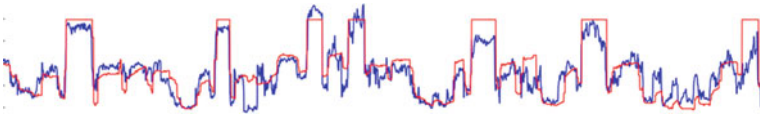


Fig. 7.30 Results

7.4.4.3 Regression Results

In a pilot study we compared EMG and ultrasound sensing approaches by training two neural network models, one with EMG input and one with ultrasound input, to predict the position of each of the five fingers. To measure position we used a VMG 10 data glove, equipped with a bend sensor and force-sensing resistor (FSR) for each finger. In an experiment, a subject was asked to move his fingers while wearing either (1) two Myoband EMG sensors, for a total of 16 electrodes, placed on the forearm to collect EMG readings or (2) an ultrasound Sonostar UProbe attached to the forearm. The subject moved his fingers while data from the glove and sensor were synchronized and recorded (see Fig. 7.27). Two neural networks, using only one hidden layer to avoid overfitting, were trained to predict the finger positions using either EMG or ultrasound. Pearson correlation between the predicted finger position values and ground truth was computed for each network. The images were pre-processed to reduce dimensionality including processes such as downsampling, histogram equalization, Gaussian blur, and background subtraction. For background subtraction, we used a background image that was acquired when the user's fingers were in a neutral and relaxed pose. The preprocessing steps helped isolate the characteristics within the image that were useful for discriminating finger positions (see Fig. 7.28). The network that used ultrasound as input outperformed significantly the network using EMG (pearson's $r = 0.82$ for ultrasound vs. $r = 0.14$ for EMG), providing promising evidence for the proposed work (see Fig. 7.30).

7.5 Conclusion

In this chapter we described our wearable robotic systems. One of the pervading themes and challenges in wearables is understanding user intent and a large portion of this chapter was dedicated to EMG and ultrasound sensing. While understanding and appropriately responding to the user is undoubtedly important, controlling wearable devices can significantly increase the cognitive load on the user, thus, some level of autonomy is typically required of the device. Here, we explore the concept of autonomy beyond just low level control, but extend it to higher level decision-making that is likely to influence the end musical result and user experience. Though we have only scratched the surface of developing and exploring interesting paradigms through the second stick of the drumming prosthetic and the third arm, we believe there is vast opportunity for enabling unique user-experiences through wearable musical robotics.

References

1. Hugh, Herr, Graham Paul Whiteley, and Dudley Childress. 2003. *Cyborg technology-biomimetic orthotic and prosthetic technology*. SPIE Press, Bellingham, Washington.
2. Llorens-Bonilla, Baldin, Federico Parietti, and H. Harry Asada. 2012. Demonstration-based control of supernumerary robotic limbs. In *2012 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 3936–3942. IEEE.
3. Brian, Dellon, and Matsuoka Yoky. 2007. Prosthetics, exoskeletons, and rehabilitation [grand challenges of robotics]. *Robotics & Automation Magazine, IEEE* 14 (1): 30–34.
4. Schirner, Gunar, Deniz Erdogmus, Kaushik Chowdhury, and Taskin Padir. 2013. The future of human-in-the-loop cyber-physical systems.
5. Li, Qinan, Weidong Chen, and Jingchuan Wang. 2011. Dynamic shared control for human-wheelchair cooperation. In *2011 IEEE international conference on robotics and automation (ICRA)*, 4278–4283. IEEE.
6. Nudehi, Shahin S., Ranjan Mukherjee, and Moji Ghodoussi. 2005. A shared-control approach to haptic interface design for minimally invasive telesurgical training. *IEEE Transactions on Control Systems Technology* 13 (4): 588–592.
7. Gil, Weinberg, and Driscoll Scott. 2006. Toward robotic musicianship. *Computer Music Journal* 30 (4): 28–45.
8. Abbink, David A., and M. Mulder. 2010. Neuromuscular analysis as a guideline in designing shared control. *Advances in Haptics* 109: 499–516.
9. Gentili, Rodolphe J., Hyuk Oh, Isabelle M. Shuggi, Ronald N. Goodman, Jeremy C. Rietschel, Bradley D. Hatfield, and James A. Reggia. 2013. Human-robotic collaborative intelligent control for reaching performance. In *Foundations of augmented cognition*, 666–675. Springer.
10. Kapur, Ajay, Michael Darling, Dimitri Diakopoulos, Jim W. Murphy, Jordan Hochenbaum, Owen Vallis, and Curtis Bahn. 2011. The machine orchestra: An ensemble of human laptop performers and robotic musical instruments. *Computer Music Journal* 35 (4): 49–63.
11. Laura, Maes, Raes Godfried-Willem, and Rogers Troy. 2011. The man and machine robot orchestra at logos. *Computer Music Journal* 35 (4): 28–48.
12. Cakmak, Maya, Crystal Chao, and Andrea L. Thomaz. 2010. Designing interactions for robot active learners. *IEEE Transactions on Autonomous Mental Development* 2 (2): 108–118.
13. Hoffman, Guy, and Gil Weinberg. 2010. Shimon: An interactive improvisational robotic marimba player. In *CHI'10 extended abstracts on human factors in computing systems*, 3097–3102. ACM.
14. Cipriani, Christian, Franco Zaccone, Silvestro Micera, and Maria Chiara Carrozza. 2008. On the shared control of an emg-controlled prosthetic hand: analysis of user-prosthesis interaction. *IEEE Transactions on Robotics* 24 (1): 170–184.
15. Chris, Lake, and Dodson Robert. 2006. Progressive upper limb prosthetics. *Physical Medicine and Rehabilitation Clinics of North America* 17 (1): 49–72.
16. Cipriani, Christian, Marco Controzzi, and M. Chiara Carrozza. 2009. Progress towards the development of the smarthand transradial prosthesis. In *IEEE international conference on rehabilitation robotics, 2009. ICORR 2009*, 682–687. IEEE.
17. Kim, Hyun K., J. Biggs, David W. Schloerb, Jose M. Carmena, Mikhail A. Lebedev, Miguel A.L. Nicolelis, and Mandayam A. Srinivasan. 2006. Continuous shared control for stabilizing reaching and grasping with brain-machine interfaces. *IEEE Transactions on Biomedical Engineering* 53 (6): 1164–1173.
18. Hochberg, Leigh R., Mijail D. Serruya, Gerhard M. Friehs, Jon A. Mukand, Maryam Saleh, Abraham H. Caplan, Almut Branner, David Chen, Richard D. Penn, and John P. Donoghue. 2006. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature* 442 (7099): 164–171.
19. Wu, Faye Y., and Harry Asada. 2014. Bio-artificial synergies for grasp posture control of supernumerary robotic fingers.
20. Davenport, Clark Clark Michael. 2013. Supernumerary robotic limbs: Biomechanical analysis and human-robot coordination training. PhD thesis, Massachusetts Institute of Technology.

21. Singer, Eric, Jeff Feddersen, Chad Redmon, and Bil Bowen. 2004. Lemur's musical robots. In *Proceedings of the 2004 conference on new interfaces for musical expression*, 181–184. National University of Singapore.
22. Weinberg, Gil, and Scott Driscoll. 2006. Robot-human interaction with an anthropomorphic percussionist. In *Proceedings of the SIGCHI conference on human factors in computing systems*, 1229–1232. ACM.
23. Puckette, Miller S., Miller S. Puckette Ucsd, Theodore Apel, et al. 1998. Real-time audio analysis tools for Pd and MSP.
24. Carlson, Tom, and José del R Millán. 2013. Brain-controlled wheelchairs: A robotic architecture. *IEEE Robotics and Automation Magazine* 20 (EPFL-ARTICLE-181698): 65–73.
25. Sun, Sisi, Trishul Mallikarjuna, and Gil Weinberg. Effect of visual cues in synchronization of rhythmic patterns.
26. Guy, Hoffman, and Weinberg Gil. 2011. Interactive improvisation with a robotic marimba player. *Autonomous Robots* 31 (2–3): 133–153.
27. Kapur, Ajay. 2005. A history of robotic musical instruments. In *Proceedings of the international computer music conference*, 21–28. Citeseer.
28. Logan-Greene, Richard. The music of Richard Johnson Logan-Greene. http://zownts.com. Accessed 5 Jan 2014.
29. Ort, Teddy, Faye Wu, Nicholas C. Hensel, and H. Harry Asada. 2015. Supernumerary robotic fingers as a therapeutic device for hemiparetic patients. In *ASME 2015 dynamic systems and control conference*, V002T27A010–V002T27A010. American Society of Mechanical Engineers.
30. Dheeraj, Nimawat, and Jailiya Pawan Raj Singh. 2015. Requirement of wearable robots in current scenario. *European Journal of Advances in Engineering and Technology* 2 (2): 19–23.
31. Samer, Mohammed, Amirat Yacine, and Rifai Hala. 2012. Lower-limb movement assistance through wearable robots: State of the art and challenges. *Advanced Robotics* 26 (1–2): 1–22.
32. Gopura, R.A.R.C., D.S.V. Bandara, Kazuo Kiguchi, and George K.I. Mann. 2016. Developments in hardware systems of active upper-limb exoskeleton robots: A review. *Robotics and Autonomous Systems* 75: 203–220.
33. Gálvez-Zúñiga, Miguel A., and Alejandro Aceves-López. 2016. A review on compliant joint mechanisms for lower limb exoskeletons. *Journal of Robotics*.
34. Gopura, R.A.R.C., Kazuo Kiguchi, and D.S.V. Bandara. 2011. A brief review on upper extremity robotic exoskeleton systems. In *2011 6th international conference on industrial and information systems*, 346–351. IEEE.
35. Rocon, E., A.F. Ruiz, R. Raya, A. Schiele, J.L. Pons, J.M. Belda-Lois, R. Poveda, M.J. Vivas, and J.C. Moreno. 2008. Human-robot physical interaction. In *Wearable robots: Biomechatronic exoskeletons*, 127–163.
36. Allan Joshua Veale and Shane Quan Xie. 2016. Towards compliant and wearable robotic orthoses: A review of current and emerging actuator technologies. *Medical Engineering & Physics* 38 (4): 317–325.
37. Lenzi, Tommaso, Nicola Vitiello, Stefano Marco Maria De Rossi, Alessandro Persichetti, Francesco Giovacchini, Stefano Roccella, Fabrizio Vecchi, and Maria Chiara Carrozza. 2011. Measuring human-robot interaction on wearable robots: A distributed approach. *Mechatronics* 21 (6): 1123–1131.
38. Knight, James F., Chris Baber, Anthony Schwirtz, and Huw William Bristow. 2002. The comfort assessment of wearable computers. *ISWC* 2: 65–74.
39. Bodine, Kerry, and Francine Gemperle. 2003. Effects of functionality on perceived comfort of wearables. In *Proceedings of seventh IEEE international symposium on wearable computers, 2003*, 57–60. IEEE.
40. Demers, Louise, Rhoda Weiss-Lambrou, and Bernadette Ska. 2002. The quebec user evaluation of satisfaction with assistive technology (quest 2.0): An overview and recent progress. *Technology and Disability* 14 (3): 101–105.
41. Nigel Corlett, E., and R.P. Bishop. 1976. A technique for assessing postural discomfort. *Ergonomics* 19 (2): 175–182.

42. Singer, Neil C., and Warren P. Seering. 1990. Preshaping command inputs to reduce system vibration. *Journal of Dynamic Systems, Measurement, and Control* 112 (1): 76–82.
43. Zhai, Xiaolong, Beth Jelfs, Rosa H.M. Chan, and Chung Tin. 2016. Short latency hand movement classification based on surface emg spectrogram with PCA. In *2016 38th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*, 327–330. IEEE.
44. Jun-Uk, Chu, Moon Inhyuk, Lee Yun-Jung, Kim Shin-Ki, and Mun Mu-Seong. 2007. A supervised feature-projection-based real-time emg pattern recognition for multifunction myoelectric hand control. *IEEE/ASME Transactions on Mechatronics* 12 (3): 282–290.
45. Huang, Yonghong, Kevin B. Englehart, Bernard Hudgins, and Adrian D.C. Chan. 2005. A gaussian mixture model based classification scheme for myoelectric control of powered upper limb prostheses. *IEEE Transactions on Biomedical Engineering* 52 (11): 1801–1811.
46. Silvia, Muceli, and Farina Dario. 2011. Simultaneous and proportional estimation of hand kinematics from emg during mirrored movements at multiple degrees-of-freedom. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 20 (3): 371–378.
47. Afshar, Pedram, and Yoky Matsuoka. 2004. Neural-based control of a robotic hand: Evidence for distinct muscle strategies. In *Proceedings of IEEE international conference on robotics and automation, 2004. ICRA'04*, vol. 5, 4633–4638. IEEE.
48. Hioki, Masaaki, and Haruhisa Kawasaki. 2012. Estimation of finger joint angles from sEMG using a neural network including time delay factor and recurrent structure. *ISRN Rehabilitation* 2012.
49. Shrirao, Nikhil A., Narender P. Reddy, and Durga R. Kosuri. 2009. Neural network committees for finger joint angle estimation from surface emg signals. *Biomedical Engineering Online* 8 (1): 2.
50. Ngeo, Jimson G., Tomoya Tamei, and Tomohiro Shibata. 2014. Continuous and simultaneous estimation of finger kinematics using inputs from an emg-to-muscle activation model. *Journal of Neuroengineering and Rehabilitation* 11 (1): 122.
51. Zheng, Yong-Ping, M.M.F. Chan, Jun Shi, Xin Chen, and Qing-Hua Huang. 2006. Sonomyography: Monitoring morphological changes of forearm muscles in actions with the feasibility for the control of powered prosthesis. *Medical Engineering & Physics* 28 (5): 405–415.
52. Castellini, Claudio, and Georg Passig. 2011. Ultrasound image features of the wrist are linearly related to finger positions. In *2011 IEEE/RSJ international conference on intelligent robots and systems*, 2108–2114. IEEE.
53. Castellini, Claudio and David Sierra González. 2013. Ultrasound imaging as a human-machine interface in a realistic scenario. In *2013 IEEE/RSJ international conference on intelligent robots and systems*, 1486–1492. IEEE.
54. Vikram, Ravindra, and Castellini Claudio. 2014. A comparative analysis of three non-invasive human-machine interfaces for the disabled. *Frontiers in Neurorobotics* 8: 24.
55. Xin, Chen, Zheng Yong-Ping, Guo Jing-Yi, and Shi Jun. 2010. Sonomyography (smg) control for powered prosthetic hand: a study with normal subjects. *Ultrasound in Medicine & Biology* 36 (7): 1076–1088.
56. Guo, Jing-Yi, Yong-Ping Zheng, Laurence P.J. Kenney, Audrey Bowen, David Howard, and Jiri J. Canderle. 2011. A comparative evaluation of sonomyography, electromyography, force, and wrist angle in a discrete tracking task. *Ultrasound in Medicine & Biology* 37 (6): 884–891.
57. Li, Yuefeng, Keshi He, Xueli Sun, and Honghai Liu. 2016. Human-machine interface based on multi-channel single-element ultrasound transducers: A preliminary study. In *2016 IEEE 18th international conference on e-health networking, applications and services (Healthcom)*, 1–6. IEEE.
58. Sikdar, Siddhartha, Huzefa Rangwala, Emily B. Eastlake, Ira A. Hunt, Andrew J. Nelson, Jayanth Devanathan, Andrew Shin, and Joseph J. Pancrazio. 2013. Novel method for predicting dexterous individual finger movements by imaging muscle activity using a wearable ultrasonic system. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 22 (1): 69–76.

59. Hariharan, Harishwaran, Nima Aklaghi, Clayton A. Baker, Huzefa Rangwala, Jana Kosecka, and Siddhartha Sikdar. 2016. Classification of motor intent in transradial amputees using sonomyography and spatio-temporal image analysis. In *Medical imaging 2016: Ultrasonic imaging and tomography*, vol. 9790, 97901Q. International Society for Optics and Photonics.
60. Gordon, Claire C., Thomas Churchill, Charles E. Clauser, Bruce Bradtmiller, John T. McConville, Ilse Tebbetts, and Robert A. Walker. 1989. *Anthropometric survey of us army personnel: Summary statistics, interim report for 1988*. Technical report, Anthropology Research Project Inc Yellow Springs OH.
61. Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI conference on artificial intelligence*.
62. Khazendar, S., A. Sayasneh, H. Al-Assam, Du Helen, L. Jeroen Kaijser, Dirk Timmerman Ferrara, S. Jassim, and Tom Bourne. 2015. Automated characterisation of ultrasound images of ovarian tumours: The diagnostic accuracy of a support vector machine and image processing with a local binary pattern operator. *Facts, Views & Vision in ObGyn* 7 (1): 7.
63. Oquab, Maxime, Leon Bottou, Ivan Laptev, and Josef Sivic. 2014. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1717–1724.
64. Cireşan, Dan C., Ueli Meier, and Jürgen Schmidhuber. 2012. Transfer learning for latin and chinese characters with deep neural networks. In *The 2012 international joint conference on neural networks (IJCNN)*, 1–6. IEEE.
65. Yosinski, Jason, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, 3320–3328.
66. Yu, Fisher, and Vladlen Koltun. 2015. Multi-scale context aggregation by dilated convolutions. arXiv:1511.07122.

Index

G. Weinberg et al., *Robotic Musicianship*, Automation, Collaboration, & E-Services 8, https://doi.org/10.1007/978-3-030-38930-7