6-7-2019

MindMusic: Brain-Controlled Musical Improvisation

Rachel Goldstein

Andy Vainauskas

SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

Date: June 11, 2019

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Rachel Goldstein and Andy Vainauskas

ENTITLED

MindMusic: Brain-Controlled Musical Improvisation

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

Thesis Advisor

Department Chair

MindMusic: Brain-Controlled Musical Improvisation

by

Rachel Goldstein and Andy Vainauskas

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 7, 2019

MindMusic: Brain-Controlled Musical Improvisation

Rachel Goldstein and Andy Vainauskas

Department of Computer Engineering
Santa Clara University
June 7, 2019

ABSTRACT

MindMusic explores a new form of creative expression through brain controlled musical improvisation. Using EEG technology and a musical improviser system, Impro-Visor (Keller, 2018), MindMusic engages users in musical improvisation sessions controlled with their brainwaves. Brain-controlled musical improvisation offers a unique blend of mindfulness meditation, EEG biofeedback, and real-time music generation, and stands to assist with stress reduction and widen access to musical creativity.

Acknowledgements

We would like to thank Dr. Maya Ackerman for her support, encouragement, and enthusiasm throughout the year. Her passion for computational creativity helped us discover a new area of interest for the both of us.
We would also like to thank Dr. Robert M. Keller for his support in our collaboration efforts as we integrated Mind-Music into his Impro-Visor system.

Table of Contents

List of Figures

1 Introduction

1.1 Motivation

In our society, an overwhelming number of individuals suffer from highly stressful environments. With all the stimuli that come from never-ending todo lists, email notifications, and social media, it can be easy for one to lose mental clarity. As a result, mental health has become an increasingly important topic of discussion. Now more than ever, it is critical that people experiencing stress have access to tools that can help them find peace of mind and regain control of their mental state.

Playing a musical instrument and meditation are two popular ways of improving mental health. These activities help provide mental clarity by promoting creative expression and relaxation. While playing a musical instrument does aid in improving focus, it lacks the ability to show users quantitative, tangible metrics about their current mental state in real time. The same can be said for many popular guided meditation apps which are also missing relevant feedback features. Muse, a headband that detects brain signals, tries to solve this problem. It provides users with charts and graphs informing them about their mental state during a meditation practice along with real time feedback through soundscapes that alter in volume and intensity based on a users brain activity as they meditate. Our solution, Mind-Music, combines the benefits of both music and enhanced meditation via Muse to provide the ideal stress management and creativity tool.

1.2 Background and Previous Work

Musical self-expression and meditation offer two complementary approaches to improving mental health and well-being. Musical improvisation has been shown to activate the sensorimotor and language areas of the brain that otherwise remain dormant during the performance of predetermined melodies (Lopez-Gonzlez and Limb, 2012). Furthermore, brain regions that manage executive functions such as planning, abstract reasoning, and working memory were found to be deactivated, which also occurs during meditation and dreaming.

In this work, we explore a new terrain of musical creative expression by cutting out all intermediaries and controlling musical improvisation directly through brain signals. This form of expression requires no training or musical expertise, is more widely accessible than traditional musical instruments, and stands to offer a variety of mental health benefits. In particular, we utilize the lightweight EEG headband, Muse[1], to access the user's EEG data, which is subsequently used to drive musical improvisation, incorporated within the musical improvisation system, Impro-Visor.[2]

Combining EEG signals with music generation offers a variety of benefits. Alvin Lucier was one of the first to transform brainwaves into sound by amplifying his EEG signals to generate music, which offered a way for creative expression to be conducted by one's brain activity (Lutters and Koehler, 2016). Loudspeakers were placed near percussive instruments to resonate when frequencies outside the human range of hearing were generated.

(Eaton et al., 2015) identified that brain waves provide means of passive control in a BCMI, to allow for mental states to be approximated and mapped to relative musical phrases (Eaton et al., 2015). In regards to therapeutic applications, (Keune Ne Muenssinger et al., 2010) expanded upon Brain-Computer Interfaces (BCIs) to enable creative expression for patients with ALS. Their P300-Brain Painting BCI produced meaningful experiences for both healthy and para-

[1]https://choosemuse.com
[2]https://www.cs.hmc.edu/ keller/jazz/improvisor/

lyzed individuals.

Real-time feedback allows the user to gauge their mental state through changes in the music, which informs the user of their mental state in regular intervals. *Biofeedback* uses instruments to provide information on one's physiological function to allow greater awareness of that function. It has been shown to be an effective way to control one's mental state (deCharms et al., 2005). By becoming aware of one's mental state in an explicit, yet pleasantly communicated manner, the user is placed in a better position to transition to a healthier state of mind.

Direct EEG feedback is complemented through an alternate improvisation driver by allowing the user's head tilts to affect the music. The Muse device can detect head position, identifying whether the user tilts their head left or right, or keeps it centered. Head position subsequently controls the durations of notes played in the improvisation. This more easily controlled form of feedback can provide musical expression to those who cannot play musical instruments due to paralysis or other disorders.

Mindfulness meditation has been shown to reduce stress and improve mental states. When compared to relaxation meditation, mindfulness meditation can better improve focus and quiet distracting thoughts (Jain et al., 2007). Mindfulness meditation has also been shown to reduce anxiety in both a broad range of clinical patients and patients suffering from generalized social anxiety disorder (Koszycki et al., 2007). Additionally, research suggests that mindfulness-based cognitive therapy significantly reduces relapse and recurrence in patients suffering from major depression with at least three previous episodes of depression (Teasdale et al., 2000).

Music is fundamentally relaxing, especially when the listener finds the music pleasurable to listen to (Stratton and Zalanowski, 1984). One study, which investigated the effects that drumming had on six soldiers combating PTSD, found a reduction in some of their symptoms after playing the instrument (Bensimon et al., 2008). Another study found that not only does listening to music reduce the perception of pain, but when combined with traditional pain-management techniques, music therapy can enhance the effectiveness of pain-management for patients recovering from surgery (Bernatzky et al., 2011). Improvisation in music in particular has a unique effect on the brain. A study looking at improvisation in professional jazz pianists that utilized functional MRI scans found that, when compared to well-rehearsed music, improvisation correlated to disassociated activity in the prefrontal cortex. This suggests that when engaging in improvisation, musicians are using fewer brain processes involved with self-monitoring, planning, and problem-solving and are instead using more areas of the brain associated with meditation and daydreaming, thereby supporting creativity in the context of improvisation (Limb and Braun, 2008).

1.3 Solution

MindMusic is an application that combines brain signals that Muse detects with innovative improvisational music technology. A users brain activity is used as input and inspiration for the real-time generation of music. This music is tailored to your current brain state and produces a unique musical expression of your mood. Instead of simply projecting ones brain activity like Muse does, MindMusic is a new way to engage in musical improvisation. By controlling the music with ones mental state, the user can experience the benefits of both biofeedback and playing a musical instrument, thereby gaining a mindful self-awareness through this revolutionary way to relieve stress.

2 Requirements

The various requirements for our system are outlined below. Functional requirements describe what the system will do, non-functional requirements detail how the functional requirements will be achieved, and the design constraints outline the various limitations necessary for our system.

2.1 Functional

The system will:

- take user's brain activity as input into the Impro-Visor system and generate responsive music as output.
- allow users to engage in unique musical improvisation, thereby providing the user with benefits of biofeedback and playing a musical instrument.
- output music in real-time.

2.2 Non-Functional

The system will be:

- accurate in its measurement of brainwave activity.
- intuitive and easily-accessible for a diverse set of users.

2.3 Design Constraints

- Our system must run on Mac or PC with Impro-Visor preinstalled.
- Users must have access to a Muse headband.

3 Use Cases

The user is the only actor that will be involved in using the MindMusic system. They will be able to perform five main actions, which can be seen in Figure 3.1.

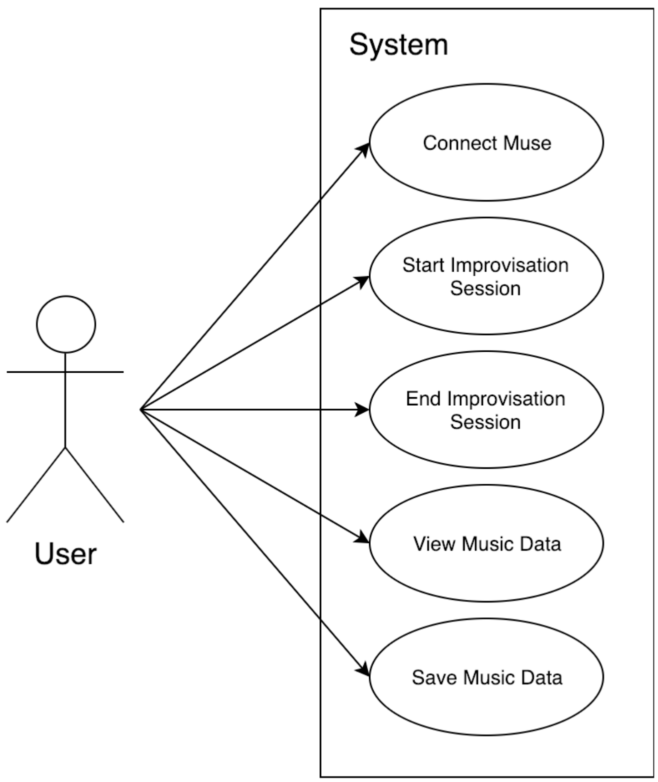

Figure 3.1: Use Case Diagram

3.1 Connect Muse

- **Goal:** Establish a Bluetooth connection between the Muse and the computer to interface with Impro-Visor.
- **Actor:** User
- **Pre-Conditions:** Both the laptop and Muse are charged and powered on.
- **Steps:** Put the Muse in Bluetooth pairing mode and connect using the laptop's Bluetooth settings.
- **Post-Conditions:** The laptop indicates that the Muse is connected.
- **Exceptions:** None

3.2 Start Improvisation Session

- **Goal:** Begin capturing data from the Muse and generate improvised music based on measured brain activity via Impro-Visor.
- **Actor:** User
- **Pre-Conditions:** The Muse and the laptop are connected via Bluetooth and Impro-Visor is open.
- **Steps:** Click the green ”Improv” button after selecting the improvisation mode.
- **Post-Conditions:** Music is continually generated based on the Muse data.
- **Exceptions:** None

3.3 End Improvisation Session

- **Goal:** Stop generating music output and end Muse data capture.
- **Actor:** User
- **Pre-Conditions:** A session is in progress.
- **Steps:** Click the red ”Stop” button.
- **Post-Conditions:** A session has ended and is ready for viewing and saving.
- **Exceptions:** None

3.4 View Music Data

- **Goal:** View the lead sheet of the generated music in Impro-Visor.
- **Actor:** User
- **Pre-Conditions:** A session has ended and music has been generated.
- **Steps:** Complete a session in order to view music data.
- **Post-Conditions:** Improvised music is arranged on a lead sheet.
- **Exceptions:** None

3.5 Save Music Data

- **Goal:** Save the lead sheet for later viewing.
- **Actor:** User
- **Pre-Conditions:** A session has ended and music has been generated.
- **Steps:** Click File, select Save, then choose the file destination.
- **Post-Conditions:** A lead sheet file is saved to the computer.
- **Exceptions:** None

4 Activity Diagram

The activities the user can engage in when using MindMusic are illustrated in Figure 4.1.

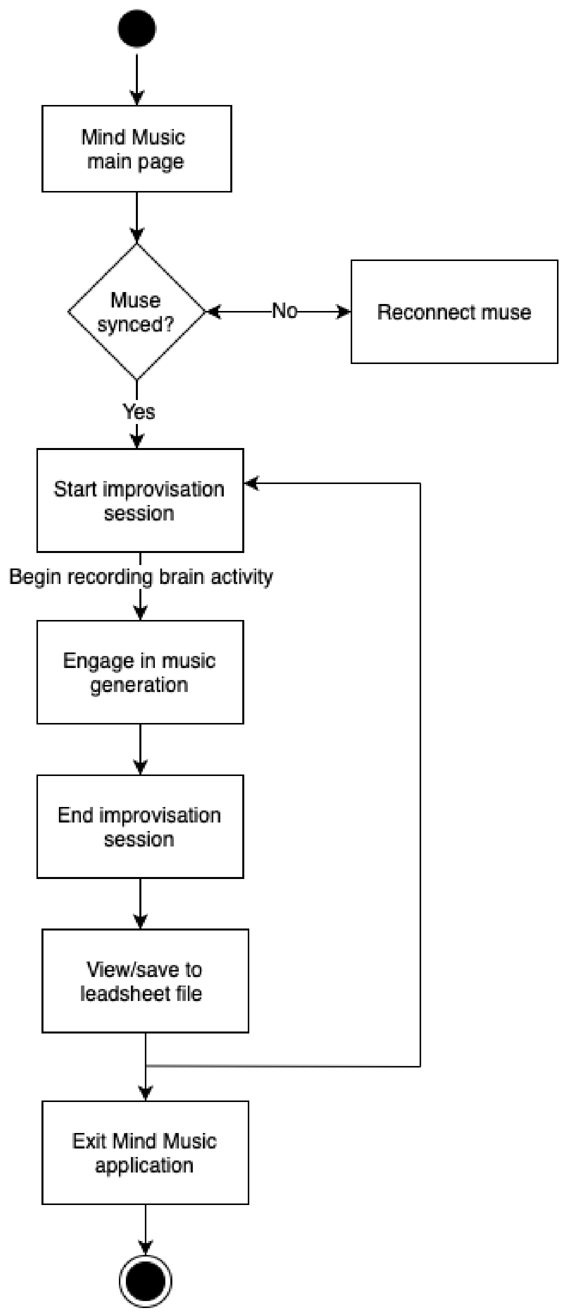

Figure 4.1: Activity Diagram

5 User Interface

The MindMusic user interface has already been created through Impro-Visor. Nearly all aspects of the visual layout and functionality remain the same, the only change that has been made is the creation of two additional modes: muse-head-tilt and muse-brainwave.

5.1 Impro-Visor

As seen in Figure 5.1, the main view of Impro-Visor features two main sections: various buttons along the top, and a lead sheet that shows the music staff, notes, and chords. The buttons control various functions of Impro-Visor, including starting, ending, and saving a session, as well as changing the improvisation settings. When Impro-Visor is initially launched, the lead sheet is empty. Notes will appear only if a saved session is loaded or after a session has begun.

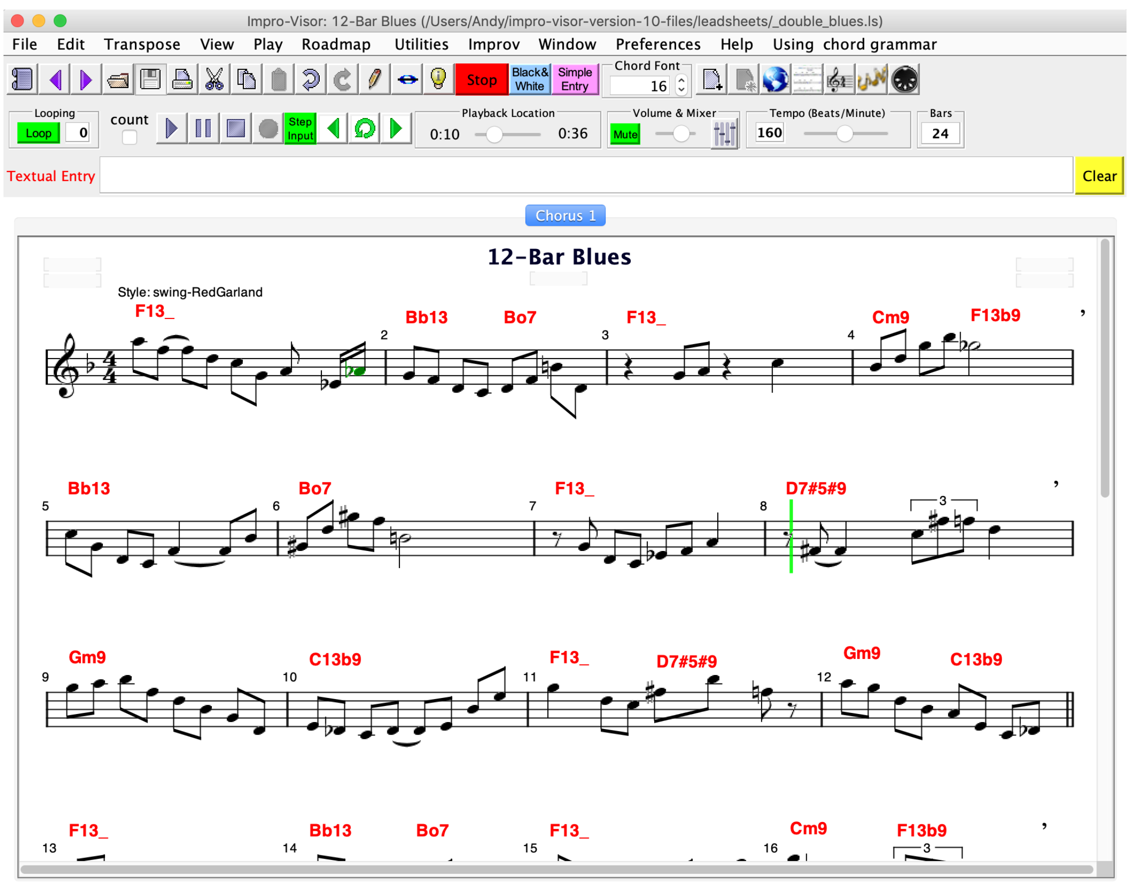

Figure 5.1: Impro-Visor Main Interface

5.2 Trading Mode

Figure 5.2 shows the settings panel for the various improvisational trading modes. Trading allows for the user to play a certain number of bars, then the computer will respond with improvised music similar to what the user played but modified in a way that makes it unique. This call and response is done several times to take turns or ”trade” playing with Impro-Visor.

- **Repeat:** Echoes the melody that the user played with only minor variations in rhythm.
- **Repeat and Rectify:** Similar to Repeat, but corrections are made on incorrect notes. Impro-Visor replaces these incorrect notes with preferred tones that are in the chord specified for that measure.
- **Modify and Rectify:** Takes the melody that the user played and performs ”composer-like transformations”, such as reversing the entire line. This is more of academic interest rather than practical musical use.
- **Abstract Melody:** Captures the shape and contour of the melodic line without using the same notes. This is done by transposing up or down intervals.
- **Transformation:** Uses one of several possible transformational grammars selected by the user, which makes the melody either more or less ornamented. These grammars were learned by Impro-Visor using transcriptions of famous jazz soloists like Charlie Parker, John Coltrane, and Miles Davis.
- **Grammar:** Uses passive trading, which generates music on its own using a ”generative grammar” rather than responding to the user. These grammars were also learned from soloists or constructed by hand.
- **Chop and Memorize:** Memorizes some of the things the user plays, but plays them later in the piece, not immediately after.

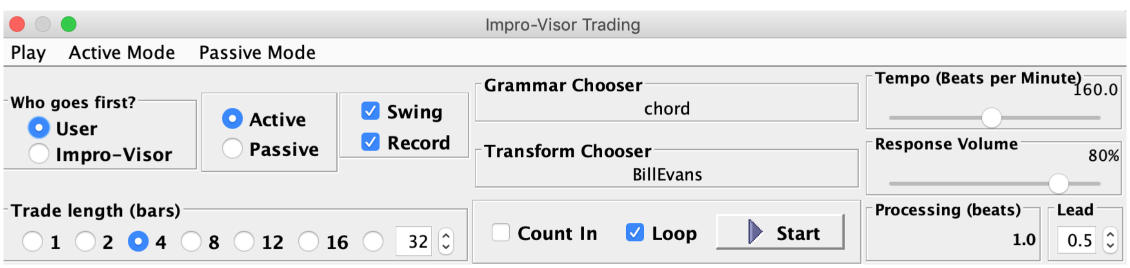

Figure 5.2: Trading Mode Settings

5.3 Grammar Selection

Impro-Visor uses a collection of probabilistic context-free grammars to establish a set of rules that are used for music generation. The grammar selection pane allows the user to change the mode or musical style to be used for music generation, and a portion of the entire list can be seen in Figure 5.3. Two additional grammars were created to support MindMusic's functionality: head tilt and brainwave. These two grammars allow the user to engage in both intentional and reflective modes of musical creation, respectively.

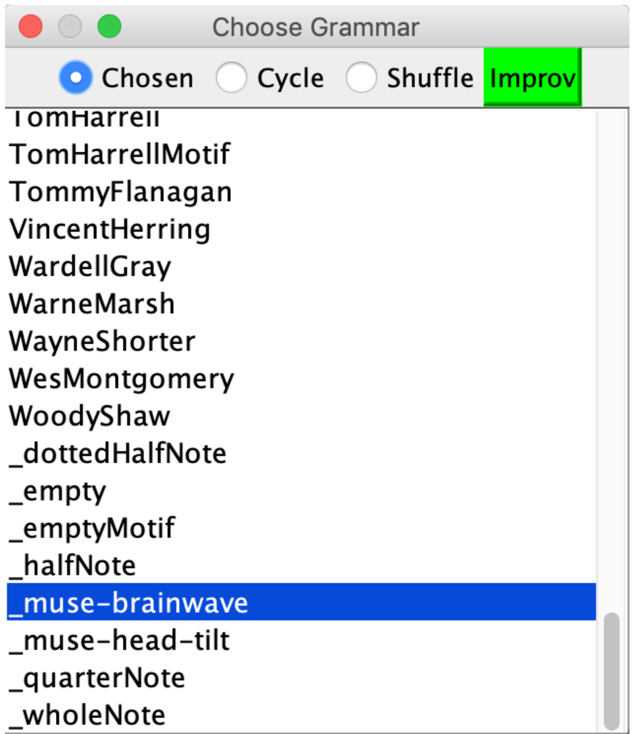

Figure 5.3: Grammar Selection Pane

6 Technologies Used

The various software and hardware technologies used in our system are listed below.

6.1 Software

- Muse Monitor iOS Application
- Impro-Visor Desktop Application
- Java
- OSC

6.2 Hardware

- Muse Headband
- MacBook

7 Architectural Diagram

MindMusic uses a Data Flow architecture, which was selected due to its simple division of independent subsystems and for its ability to be easily expanded. The Muse uses Bluetooth to connect to a device running the Impro-Visor software. Once Impro-Visor is initialized, an OSC server begins to listen for the data being sent from the Muse. During an improvisation session, Impro-Visor processes the Muse data, categorizes the current brain activity level, and generates music that is reflective of the user's current mental state.

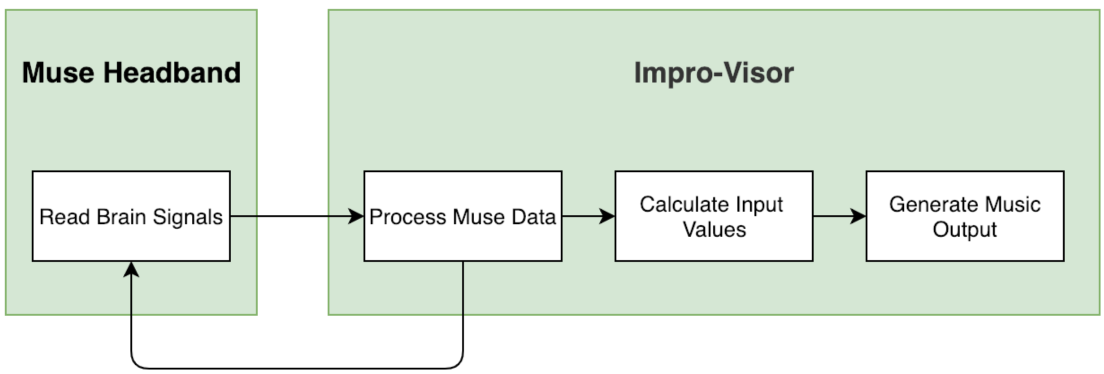

Figure 7.1: Data Flow Architecture

8 Design Rationale

- **Muse Headband:** The Muse is a headband with seven EEG sensors, two on the forehead, two behind the ears, and three reference sensors.[1] It samples brainwave data at a rate of 10 Hz, and can connect to a device over Bluetooth. The Muse device is the first to offer a lightweight, affordable EEG solution for the broad consumer market.

- **Alpha Brainwaves:** EEG brainwaves can be split into five frequency-based categories – delta, theta, alpha, beta, and gamma. At each moment, our brains have activity in each of these frequency ranges. Our research focuses on the most prominent relaxation-related band, alpha brainwaves, which act as the bridge between the conscious and the subconscious. There is more activity in the alpha frequency range when a person is alert but not actively concentrating or processing information. Alpha brainwaves are linked to feelings of calmness, relaxation, and tranquility while being conscious and non-drowsy.[2] These waves tend to be more active during meditation than during normal activity, for example. Looking at the power spectral density function (PSD), higher readings within the alpha range indicate that a user is in an "alpha state." [3]

- **Impro-Visor:** Impro-Visor is a unique tool that makes improvisational music generation possible. Impro-Visor provides a streamlined way to convert brain activity to generated music that is improvised in real-time. Rather than using MIDI notes as input, MindMusic uses an OSC server to process the data from the Muse into generated music.

- **Brain Data Activity Categorization:** To find the user's neutral state, MindMusic begins with a 22 second calibration stage. This phase allows MindMusic to assess what constitutes a relaxed or anxious state for different users. During this period, a sample is taken of the user's alpha brainwaves. From that sampling, the user's average alpha, μ, and standard deviation, σ, are saved.

  Subsequently, using the user's most recent three seconds of alpha brainwave data, the average of this collection of data is taken. From there, the z-score of this new sampled alpha is calculated, using the following equation,

  $$z = \frac{x - \mu}{\sigma}$$

  where x is the average of the most recent read values, and μ and σ are the values saved from the user's calibration stage.

  Using this z-score, these new alpha values can then be mapped to different activity states. We chose to have five different activity states: very low, low, medium, high, and very high. A z-score close to the user's average will be closer to 0 and categorized as a medium activity level, very high and low z scores are categorized as very low and very high activity levels, and those in-between are categorized as low and high. Here, it is important to note that a higher alpha value read in reflects that the brain has more brainwaves that are falling into the alpha frequency range, and so the user can be seen as more calm and less active.

- **Musical Representation:** Initially, we created the head tilt grammar as a way to test the system. Since accelerometer data is easily identifiable, we could ensure that the Muse was communicating properly by having Impro-Visor perform specific actions based on the direction of head tilt. For example, if the user's head is tilted to the left, the grammar will execute a rule that returns notes that have a longer duration, such as quarter notes. If their head is tilted to the right, notes with a shorter duration are generated, such as eighth notes. During

[1] https://choosemuse.com/how-it-works/
[2] https://nhahealth.com/brainwaves-the-language/
[3] https://musemonitor.com

this testing, we realized that using the accelerometer to intentionally influence music generation was a fun and exciting way to make music, so we added it as a separate feature of MindMusic.

The brainwave grammar uses alpha waves read by the Muse that have already been categorized into an activity level. If the level is “low” or “very low”, then notes of a longer duration are generated (half or whole notes). If the level is “medium”, this means their current activity level is similar to what was measured during the calibration stage, so quarter notes are generated. If their activity level is “high” or “very high”, then faster notes, such as eighth and sixteenth notes, are generated. To improve brain activity categorization, we tested MindMusic with several users to get their feedback on the accuracy of the musical representation of brain activity. Minor adjustments were made to the range calculations after discovering that achieving a “very low” or “very high” state occurred too often. Future work will include detailed user studies.

9 Testing

In order to validate the functional and non-functional requirements previously specified, we conducted a series of extensive tests of our system.

9.1 White Box Testing

- Unit Tests
  - Muse Data
    - Is the OSC Server receiving all the data from the Muse?
    - Is the data received in Impro-Visor correct?
    - Is the data from the Muse correctly normalized?
  - Improvisation Input Values
    - Are the generated values unique based on the Muse data?
    - Are the generated values abstract to allow for different attributes of improvisation to be influenced?
  - Music Output
    - Does the generated music reflect changes in Muse data?
    - Does the music generation allow for conscious control?
- Validation Testing
  - Overall System
    - Does the MindMusic system achieve all specified requirements?
    - Is each component correctly communicating with each other?

9.2 Black Box Testing

- Unbiased Users
  - Can users of all technical backgrounds use the system with minimal guidance?
- Achieve Goals
  - Can users achieve the specified goals and tasks required by the system?
- User Perspectives
  - Do users feel that their activity level is accurately represented in the music?
  - Study a wide range of user perspectives to improve the usability of the system.

10 Risk Analysis

There is an assumption of risk when developing a project of any size. The most probable risks that may be encountered during the development of Mind Muse are outlined in Table 10.1. The probabilities of each risk are rated on a 0-1 point scale, with a larger number noting a higher probability of encountering that risk. The severity of each risk is rated on a 0-10 point scale, with a larger number noting a higher severity. The impact is the product of the probability and severity. The table is ranked by impact.

Table 10.1: Risk Analysis

Risk	Consequences	Probability	Severity	Impact	Mitigation
Platform Compatibility Issues	Data from the Muse will need to be processed outside of the Impro-Visor software.	0.7	7.5	5.25	Begin to work early to ensure all possible solutions of getting Muse data into Impro-Visor are attempted.
Bugs In System	Delays are introduced to the development timeline, pushing back expected dates of completion.	0.99	4	3.96	Write clean code, test often, and read documentations thoroughly
Time/Scheduling	An increase in school workload will limit the ability to include all of the project features on time.	0.9	2.5	2.25	Spend significant time on a development timeline. If time constraints arise, establish communication with the group early and work towards an altered timeline and task assignment.
Overambitious Goals	The project becomes increasingly difficult to complete due to addional features being added during the development process.	0.3	4	1.2	Before beginning development, outline clear and concise goals and functionality. If these goals are achieved with time remaining, only then will features be added.

11 Development Timeline

In order to ensure our system will be completed on time and within the required specifications, we have outlined the distribution of our time through the weeks leading up to the Senior Design conference in Table 11.1

Table 11.1: Development Timeline

	Weeks of Fall 2018			Winter Break	Weeks of Winter 2019			Weeks of Spring 2019		
TASK	**1-3**	**4-7**	**8-10**	**Dec. 7 - Jan. 6**	**1-3**	**4-7**	**8-10**	**1-3**	**4-7**	**8-10**
Reports and Presentations										
Problem Statement										
Project Definition										
Design Report										
Design Review										
Final Report										
Final Presentation										
Implementation										
Muse Data in Impro-Visor										
Generate Improvisation Values										
Explore Other Features										
Testing										
White Box Testing										
Black Box Testing										

12 Social Implications

12.1 Ethical

An ethical concern that could be raised is data privacy. MindMusic analyzes brain data during sessions. However, the app that propagates this data from iPhone to computer, Muse Monitor, does not save data at any point, and neither does our system. Data privacy will continue to be a concern to keep in mind if the system expands to a commercial market.

12.2 Social

Mindfulness meditation has been shown to reduce symptoms of major depression and anxiety, and music has been shown to alleviate pain. Through the integration of mindfulness and music, MindMusic has the potential to lead to more effective treatment options for those suffering from similar conditions. MindMusic gives healthy individuals an audible way to connect with their current mental state, and this approach may help people with disabilities engage in musical expression when they cannot engage in this creative activity through traditional instruments.

12.3 Political

While this work does not have any specific political implications, as mentioned in our background section, three out of four Americans experience frequent symptoms from stress. This system could act as a tool for stress-relief.

12.4 Economic

There is a slight economic barrier to this project. The Muse, as of the writing of this paper, costs about 200 USD. So, if one cannot afford that investment, they will not be able to use MindMusic. A laptop and iPhone, as well as the app Muse Monitor (which costs 20 USD), are also needed. Impro-Visor, however, is free to download and use.

12.5 Health and Safety

The Muse headband is non-invasive, and has passed TUV, CE, and FCC certification as a safe consumer device. If we continue to research and include human subjects, we will have to ensure that we create a safe environment for them.

12.6 Manafacturability

MindMusic is very easy to build. All one has to do is download Impro-Visor, download the Muse Monitor app on their iPhones, and connect their Muse headband to their iPhones via Bluetooth and the system is ready to be used.

12.7 Sustainability

This system can be useful for a reasonable amount of time. Given that the Muse headband, iPhone, and laptop are functioning correctly, the user should be able to continue to engage with the system. However, if bugs or issues arise, we do not have an active support team to address them, which could be an issue if this project continues to grow but a support team does not.

12.8 Environmental Impact

There are not any significant environmental impacts of this project. MindMusic relies on energy consumption because of the technology that it uses, but these technologies are often already used by people everyday.

12.9 Usability

This project is fairly user friendly. The user will have to know their IP address and manually input it into Muse Monitor. Their interaction with the system will be via Impro-Visor, which is quite easy to use as it has a working graphical user interface.

12.10 Lifelong Learning

The experience of working on this project was similar to what a computer engineer might encounter in their career. We learned how to evaluate an existing codebase and find ways in which to expand upon a large project, both of which are necessary skills in the computer engineering field. Additionally, MindMusic required us to research and learn about the area of neuroscience, something that we have not had any previous background or experience in.

12.11 Compassion

Through this project, we aimed to widen the access to musical creativity. Not everyone can play an instrument, either because they have not been trained or because they lack the physical ability to do so. MindMusic provides an opportunity for them to use their mind as an instrument and perhaps still gain benefits from playing an instrument and making music in this way. Additionally, we wanted to create a tool to help people be mindful and regain control over their mental states; these are two topics of concern across the nation. By combining both mindfulness meditation and musical improvisation, perhaps the user can benefit from both of these activities as they engage with MindMusic.

13 Conclusion

Through the integration of mindfulness and music, MindMusic has the potential to serve as a stress management tool as well as provide the opportunity for creative expression. MindMusic gives individuals an audible way to connect with their current mental state, which may help people with disabilities engage in musical expression when they cannot engage in this creative activity through traditional instruments.

13.1 Lessons Learned

- Project Integration

  MindMusic was built upon an existing application, Impro-Visor, which has been developed for over ten years. As such, there are many existing components that are coordinated to produce improvised music notation and generation. This required significant time spent on learning how the system works and how our idea would be able to be incorporated into the existing structure of Impro-Visor. We had the benefit of being able to conduct meetings with Dr. Robert Keller, the creator of Impro-Visor, who was very helpful in ensuring that we could achieve the goals we had set for this project.

- Project Scope

  We had initially envisioned MindMusic being able to generate music with multiple aspects reflective of brain activity. We quickly realized the difficulty in developing a project that could achieve this, so we reevaluated our expectations and focused on providing an experience that most accurately reflected one's brain activity, rather than having a multitude of musical aspects be influenced by brain activity.

- Understanding Brain Activity

  Since MindMusic depends on a basic understanding of neuroscience to understand how to properly analyze and categorize brainwave data, we encountered a fairly steep learning curve during this phase of development. This is something that we did not account for enough initially in our development timeline, as we underestimated the lengths needed to fully understand this crucial aspect of MindMusic.

13.2 Advantages and Disadvantages

One of the main advantages of MindMusic is that it is already incorporated with the working version of Impro-Visor. This means that it is ready to download from the Impro-Visor website.[1] However, one of the drawbacks of MindMusic is its dependence on the Muse, which is a piece of hardware that many individuals don't already own. Additionally, the cost of purchasing a Muse might prevent more users from using the system.

13.3 Future Work

MindMusic has laid the groundwork in this new area of musical creativity upon which future improvements and expansion can be made. We have initiated the exploration of mind-driven improvisation by effecting the note duration based on either head tilt or brain activity. However, there is significant potential to utilize a variety of musical features to reflect a user's mental state and create an appropriate musical mood.Aspects such as pitches, volume, tempo, dynamics, note register, and note quality could also be influenced by brain data, and more advanced systems could reflect

[1] https://www.cs.hmc.edu/ keller/jazz/improvisor/

higher order concepts such as the musical tone and mood.

The other four EEG brainwaves (beta, theta, delta, gamma) can be integrated to influence the musical generation. The power spectral density function (PSD) gives an indication of how strongly an individual is generating each type of brainwave. By incorporating other waves, a more complete picture of one's mental state could be captured and better reflected in the music.

Finally, isolating data from each individual sensor, instead of collecting the average across all sensors, could provide more insight to how each hemisphere of the brain is working. Communication across hemispheres could be measured using the two sensors positioned at the middle of the forehead, and the two on the sides could indicate their individual levels of activity.

Appendices

A Source Code

A.1 OSC Server Code

```java
/**
 * This Java Class is part of the Impro-Visor Application
 *
 * Copyright (C) 2019 Robert Keller and Harvey Mudd College
 *
 * Impro-Visor is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * Impro-Visor is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * merchantability or fitness for a particular purpose. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with Impro-Visor; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
 */

package imp.osc;
import oscP5.*;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.Queue;

/**
 *
 * @author Andy and Rachel
 */
public class MuseServer {

    OscP5 oscServer;

    double currentAccValue = 0.0;
    double currentAlphaValue = 0.0;

    int alphasSize = 220;
    public int windowSize = 10;
    ArrayList<Double> alphas = new ArrayList<Double>();
    Queue<Double> window = new LinkedList<Double>();

    double averageAlpha = 0.0;
    double standev = 0.0;
    double windowSum = 0.0;

    void oscEvent(OscMessage msg) {
```

```
    // Accelerometer Data
    if (msg.checkAddrPattern("/muse/acc") == true) {
        currentAccValue = msg.get(1).floatValue();
    }

    // Brain Data
  if (msg.checkAddrPattern("/muse/elements/alpha_absolute") == true) {
        currentAlphaValue = msg.get(0).floatValue();
        updateWindow();

        // Saves a collection of alpha values to compute an average from (calibration phase)
        if (alphas.size() < alphasSize) {
            alphas.add(currentAlphaValue);
        } else {
            if (averageAlpha == 0.0) {
                System.out.println("CALIBRATION COMPLETE");
                calculateAverage();
                calculateStandardDeviation();
            }
        }
  }
}

void updateWindow() {
  Double poppedSample = 0.0;
  if (window.size() < windowSize) { //initializing queue
        window.add(currentAlphaValue);
        windowSum += currentAlphaValue;
  }
  else {
        window.add(currentAlphaValue);
        poppedSample = window.remove();
        windowSum += currentAlphaValue;
        windowSum -= poppedSample;
  }
}

void calculateAverage() {
  Double sum = 0.0;
  for (Double alpha : alphas) {
        sum += alpha;
  }
  averageAlpha = sum / alphas.size();
}

void calculateStandardDeviation() {
  Double sum = 0.0;
  for (Double alpha : alphas) {
        sum += Math.pow((alpha - averageAlpha), 2);
  }
  standev = Math.sqrt(sum/alphas.size());
}

void resetCalibration() {
    alphas.clear();
    window.clear();
    averageAlpha = 0.0;
    standev = 0.0;
```

```
        windowSum = 0.0;
    }

    double getAccValue() { return currentAccValue; }

    double getAlphaValue() { return currentAlphaValue; }

    double getAverageAlpha() { return averageAlpha; }

    double getSD() { return standev; }

    double getWindowSum() { return windowSum; }

}
```

A.2 Muse Receiver Code

```
/**
 * This Java Class is part of the Impro-Visor Application
 *
 * Copyright (C) 2019 Robert Keller and Harvey Mudd College
 *
 * Impro-Visor is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * Impro-Visor is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * merchantability or fitness for a particular purpose. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with Impro-Visor; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
 */
package imp.osc;
import oscP5.*;

/**
 *
 * @author Andy and Rachel
 */
public class MuseReceiver {

    MuseServer museServer;
    int recvPort = 5003;

    public MuseReceiver() {
        museServer = new MuseServer();
        museServer.oscServer = new OscP5(museServer, recvPort);
    }

    public Long getMuseValue(Long grammarMode) {

        // _muse-head-tilt grammar
```

```
    if (grammarMode == 0) {
        double currentAccValue = museServer.getAccValue();

        if (currentAccValue < -0.2) {
            System.out.println("LEFT");
            return 0L;
        } else if (currentAccValue > 0.2) {
            System.out.println("RIGHT");
            return 1L;
        } else {
            System.out.println("MIDDLE");
            return (long)Math.round(Math.random());
        }
    }

    // _muse-brainwave grammar
    else {
        double averageAlpha = museServer.getAverageAlpha();
        double standev = museServer.getSD();
        double sampledAlpha = museServer.getWindowSum() / museServer.windowSize;

        // Only outputs when calibration is complete
        if (averageAlpha != 0.0) {
            double zscore = (sampledAlpha - averageAlpha)/standev;
            System.out.println("\nZ Score: " + zscore);
            System.out.println("Average Alpha: " + averageAlpha);
            System.out.println("Sampled Alpha: " + sampledAlpha);
            System.out.println("Standard Dev: " + standev);

            if (zscore < -2.0) {
                System.out.println("VERY HIGH");
                return 4L;
            }
            else if (zscore >= -2.0 && zscore <= -0.75)
            {
                System.out.println("HIGH");
                return 3L;
            }
            else if (zscore > -0.75 && zscore <= 0.70) {
                System.out.println("MEDIUM");
                return 2L;
            }
            else if (zscore > 0.70 && zscore <= 2.0) {
                System.out.println("LOW");
                return 1L;
            }
            else {
                System.out.println("VERY LOW");
                return 0L;
            }
        }
        else {
           return -1L;
        }
    }
}

public void resetCalibration() {
    museServer.resetCalibration();
```

```
    }
}
```

A.3 Grammars

A.3.1 Brainwave

```
/*
 * Example of the new muse-switch wrapper available in grammars.
 * Robert Keller, 4/23/2019
 * With this addition, the old builtin muse is no longer supported
 * (although it could be revived if necessary).
 *
 * The wrapper muse-switch generally expands to a list that can be further expanded.
 * It is an element in a right-hand side list, not the right-hand side by itself.
 * It has two arguments:
 *    an expression that evaluates to an integer (Long), which will be sent
 *    to the internal method getMuseValue, and
 *
 *    an association list, which is a list of pairs giving the replacements
 *    for each value that the internal method getMuseValue can return.
 *
 * All told, we have lots of parentheses, so care must be taken to make sure
 * there is the right number and that everything matches correctly.
 *
 * The integer (Long) value that getMuseValue returns is used to find the
 * first pair in the association list, the first component of which matches.
 * The expansion is then the second component.
 *
 * If there is no match, the expansion is the one given in the default pair.
 * Such a pair should always be provided.
 *
 * For example, below the argument 1 to muse switch is evaluated to give 1.
 * 1 is passed to getMuseValue. If the latter returns 0, then the expansion
 * will be low, which is further expanded into C4.
 * But if, instead, getMuseValue returns, say, -1, then default will be
 * used and the expansion will be none, which is further expanded into R1.
 */

/* Note that the (parameter (chord-tone-weight 0.7)) has to be non-zero,
 * otherwise chord tones will not be produced, even if C1 for example
 * is specified.
 */

(startsymbol P)

(rule (P) ( (muse-switch 1 ((default none) (0 v_low) (1 low) (2 mid) (3 high) (4 v_high))) ) 1)
(rule (none) (R1) 1)
(rule (v_low) (C1) 1)
(rule (low) (C2) 1)
(rule (mid) (C4) 1)
(rule (high) (C8 C8) 1)
(rule (v_high) (C16 C16 C16 C16) 1)

(parameter (use-grammar true))
```

```
(parameter (rectify false))
(parameter (avoid-repeats false))
(parameter (use-grammar true))
(parameter (scale-type Use_First_Scale))
(parameter (scale-1x D))
(parameter (auto-fill true))
(parameter (chord-tone-decay 0.0))
(parameter (chord-tone-weight 0.7))
(parameter (color-tone-weight 0.15))
(parameter (leap-prob 0.01))
(parameter (max-duration 8))
(parameter (max-interval 6))
(parameter (min-duration 8))
(parameter (min-interval 0))
(parameter (rest-prob 0.1))
(parameter (scale-tone-prob 0.05))
(parameter (use-syncopation false))
(parameter (syncopation-type C))
(parameter (syncopation-multiplier 0.0))
(parameter (syncopation-constant 0.2))
(parameter (expectancy-multiplier 0.0))
(parameter (expectancy-constant 0.7))
```

A.3.2 Head-Tilt

```
(startsymbol P)
(rule (P) ( (muse-switch 0 ((0 left) (1 right))) ) 1)
(rule (left) (C4) 1)
(rule (right) (C8 C8) 1)

(parameter (use-grammar true))
(parameter (rectify true))
(parameter (avoid-repeats true))
(parameter (scale-type Use_First_Scale))
(parameter (min-pitch 58))
(parameter (max-pitch 82))
(parameter (min-interval 0))
(parameter (max-interval 6))
(parameter (min-duration 8))
(parameter (max-duration 8))
(parameter (rest-prob 0.1))
(parameter (use-syncopation false))
(parameter (syncopation-type C))
(parameter (syncopation-multiplier 0.0))
(parameter (syncopation-constant 0.7))
(parameter (expectancy-multiplier 0.0))
(parameter (expectancy-constant 0.7))
(parameter (leap-prob 0.01))
(parameter (chord-tone-weight 0.7))
(parameter (color-tone-weight 0.2))
(parameter (scale-tone-weight 0.1))
(parameter (chord-tone-decay 0.0))
(parameter (auto-fill true))
(parameter (scale-root C))
```

References

Moshe Bensimon, Dorit Amir, and Yuval Wolf. Drumming through trauma: Music therapy with post-traumatic soldiers. *The Arts in Psychotherapy*, 35(1):34 – 48, 2008. ISSN 0197-4556. doi: https://doi.org/10.1016/j.aip.2007.09.002. URL http://www.sciencedirect.com/science/article/pii/S0197455607000883.

Guenther Bernatzky, Michaela Presch, Mary Anderson, and Jaak Panksepp. Emotional foundations of music as a non-pharmacological pain management tool in modern medicine. *Neuroscience Biobehavioral Reviews*, 35(9):1989 – 1999, 2011. ISSN 0149-7634. doi: https://doi.org/10.1016/j.neubiorev.2011.06.005. URL http://www.sciencedirect.com/science/article/pii/S0149763411001151. Pioneering Research in Affective Neuroscience: Celebrating the Work of Dr. Jaak Panksepp.

R. Christopher deCharms, Fumiko Maeda, Gary H. Glover, David Ludlow, John M. Pauly, Deepak Soneji, John D. E. Gabrieli, and Sean C. Mackey. Control over brain activation and pain learned by using real-time functional mri. *Proceedings of the National Academy of Sciences*, 102(51):18626–18631, 2005. ISSN 0027-8424. doi: 10.1073/pnas.0505210102. URL https://www.pnas.org/content/102/51/18626.

Joel Eaton, Duncan Williams, and Eduardo Miranda. The space between us: Evaluating a multi-user affective brain-computer music interface. *Brain-Computer Interfaces*, 2(2-3):103–116, 2015. doi: 10.1080/2326263X.2015.1101922. URL https://doi.org/10.1080/2326263X.2015.1101922.

Shamini Jain, Shauna L. Shapiro, Summer Swanick, Scott C. Roesch, Paul J. Mills, Iris Bell, and Gary E. R. Schwartz. A randomized controlled trial of mindfulness meditation versus relaxation training: Effects on distress, positive states of mind, rumination, and distraction. *Annals of Behavioral Medicine*, 33(1):11–21, Feb 2007. ISSN 1532-4796. doi: 10.1207/s15324796abm3301.

Robert M. Keller. Impro-Visor: Jazz Improvisation Advisor for the Improviser. 2018. URL https://www.cs.hmc.edu/ keller/jazz/improvisor/.

Jana Keune Ne Muenssinger, Sebastian Halder, Sonja Kleih, Adrian Furdea, Valerio Raco, Adi Hsle, and Andrea Kbler. Brain painting: First evaluation of a new braincomputer interface application with als-patients and healthy volunteers. *Frontiers in neuroscience*, 4:182, 11 2010. doi: 10.3389/fnins.2010.00182.

Diana Koszycki, Melodie Benger, Jakov Shlik, and Jacques Bradwejn. Randomized trial of a meditation-based stress reduction program and cognitive behavior therapy in generalized social anxiety disorder. *Behaviour Research and Therapy*, 45(10):2518 – 2526, 2007. ISSN 0005-7967. doi: https://doi.org/10.1016/j.brat.2007.04.011. URL http://www.sciencedirect.com/science/article/pii/S0005796707001003.

Charles J. Limb and Allen R. Braun. Neural substrates of spontaneous musical performance: An fmri study of jazz improvisation. *PLOS ONE*, 3(2):1–9, 02 2008. doi: 10.1371/journal.pone.0001679. URL https://doi.org/10.1371/journal.pone.0001679.

Monica Lopez-Gonzlez and CJ. Limb. Musical creativity and the brain. *Cerebrum*, 02 2012.

Bart Lutters and Peter J. Koehler. Brainwaves in concert: the 20th century sonification of the electroencephalogram. *Brain*, 139(10):2809–2814, 08 2016. ISSN 0006-8950. doi: 10.1093/brain/aww207. URL https://doi.org/10.1093/brain/aww207.

Valerie N. Stratton and Annette H. Zalanowski. The Relationship Between Music, Degree of Liking, and Self-Reported Relaxation. *Journal of Music Therapy*, 21(4):184–192, 12 1984. ISSN 0022-2917. doi: 10.1093/jmt/21.4.184. URL https://doi.org/10.1093/jmt/21.4.184.

J. D. Teasdale, J. M. G. Williams, J. M. G. Williams, J. M. Soulsby, and M. A. Lau. Prevention of relapse/recurrence in major depression by mindfulness-based cognitive therapy. *Journal of Consulting and Clinical Psychology*, 68(4): 615–623, Feb 2000. ISSN 1532-4796. doi: 10.1207/s15324796abm3301$_2$.