Table of Contents

Abstract ........................................................................................................................................... 2
Introduction/Background/Inspiration .............................................................................................. 2
Relevant Prior Work .......................................................................................................................... 2
Unique Potential of Jam Sesh ........................................................................................................... 3
Purpose of Project ............................................................................................................................. 3
Method ............................................................................................................................................... 4
How we will proceed ......................................................................................................................... 4
Compartmentalization ......................................................................................................................... 6
Analysis — James, Thomas, Stephen .................................................................................................. 6
Synthesis — James, Thomas, Edward, Ben ......................................................................................... 6
Application(GUI/Computer Application) - Stephen, Ben, Edward .................................................... 6
Schedule ............................................................................................................................................ 7
Resources/Budget ............................................................................................................................... 7
Recent Developments ....................................................................................................................... 7
References .......................................................................................................................................... 8
Abstract

Jam Sesh is a computer-based application that will analyze music in real time and synthesize background music to support the user. The application will give the user a choice of different instruments and/or genres with a preset template of instruments to support the user. The program will be swiftly adaptable to changes made by the user, be it a different key, tempo, or rhythm.

Introduction

Jam Sesh’s central purpose is to enhance the experience of freestyling music by providing accompaniment. The current state of computer programs or applications relating to Jam Sesh’s focus are components of the goal, such as APIs devoted to the real time analysis and synthesis of music. We will be implementing a few such APIs, yet the core concept of our mission, to our knowledge, has never been done. The team’s inspiration for the project comes from our love of music, and our vision of making the next essential item in technological and music interaction.

Prior Work

Auto-generated accompaniment programs have been made before, such as “Midi Utility” created by KH Midi Music Ltd. This program differs from our project goal, however, in that it requires the user to input the music as a midi file and select a number of options. Our goal is to have a program respond in real time to live music, requiring only a few user options, including instrument selection. Unfortunately, we will not be able to use the prior work done in “Midi Utility,” as its code is not open source. Another important piece of prior work regarding real-time improvisation has been done by Al Biles of RIT; his project, GenJam, is based on a genetic algorithm that can learn to improvise jazz based upon the input of the user. This project is intriguing because it runs in real-time just as we hope to do, but our goal is to do so without a genetic algorithm and instead aim purely for improvisation.
Unique Potential

There are other computer programs with similar functions being done already. For example, two YouTube videos (https://www.youtube.com/watch?v=bB8kGVXdlg4 and https://www.youtube.com/watch?v=rFBhwQUZGxg) have shown computer programs with the function of improvising live music. However, we believe Jam Sesh is unique. In both videos, computer programs generated a real-time improvised audio track, but neither of the two programs actually generated completely improvised background accompaniment. Music generated by the program in 1st video is more similar to a counterpoint of the user input, which is almost as important as the user input. In this program, both the program and the user are improvising. In Jam Sesh, the user will do the most part of improvisation. Music generated by Jam Sesh is not meant to be an equally-important counterpoint of the user input but a background support of the user input. In the 2nd video, the program generated improvised music in a fixed structure. The program knew the overall structure of the improvised piece (the number of choruses overall, number of choruses that repeat the head, etc.) and thus requires the user to have a big idea of the structure of the music, which is hard for non-professional users. In Jam Sesh, we would like users to be able to use our program without having to worry about the overall structure. Jam Sesh would ask the user to choose the overall key, rhythm, and maybe total time of the music, but it would not need the user to enter a structure.

Overall, we believe Jam Sesh is unique because it is designed to generate background supports of music in real-time for non-professional users’ improvisation. We believe the current existed solutions are too complicated for normal users, not in completely real-time, or not real background supports.

Purpose of Project

The aim of this project is to demonstrate the power of a basic yet strong connection between music and technology and how they can benefit each other. We thought that this program will fill a missing part in this connection, as the two have become incredibly dependent as seen by autotune, iTunes, or a tuner app on your phone. Not only does this project aid this connection it is also incredibly marketable and would be useful to any and all musicians. We could see the future of this project be an app for smartphones, or the fine tuning of synthesis, or addition of iTunes.
Method

The requirements for the user to run the program is Java, microphone, and speakers that will not interfere with the microphone.

The reason for the split in the recording to music analysis is that Tarsos can do real time pitch detection, meaning that we can get instant basic data about the users music, but we need to send a specific file to send to The Echo Nest, meaning it isn’t real time but we get a better characterization of the music. This means we will use Tarsos for detecting changes such as a key change, while we use The Echo Nest for general Music synthesis.
This is a first draft of the layout and function of our Computer Applications UI.

The UI allows the user to see the specific .wav file for each component of the background music playing, including seeing his own. He will have the option of choosing the microphone and speakers. The buttons up top allow the user to start, stop, and pause the background music. The buffer time allows the user to choose the amount of time the program waits before starting to record allowing the user to set up after hitting record. Choosing background genre in the top right dictates what instruments will be playing, although you will also have the option of making your custom set of instruments. We have not decided on the other choices in the toolbar but we know we want to add features to import a .wav file to play in the background and the ability to export all the .wav files compounded together.
**Procedure**

We will start by understanding the feedback from the two APIs, Tarsos and The Echo Nest, along with beginning to generate basic Synthesised sounds to a basic rhythm and key. We will also start writing the computer application but mainly we need to test the interplay between analysis and synthesis.

**Compartmentalization**

**Analysis**

*James, Thomas, Stephen*

Plan A: Input is a .wav file of the user's music— the application will analyze for pitch, tempo, rhythm, key, and chord progression and pass on a data table of said information to the synthesis group.

Fallbacks: Our main contingency plan is to not analyze the music in real-time and instead review the performed piece.

**Synthesis**

*James, Thomas, Edward, Ben*

Plan A: Given the user's desired selection of instruments (e.g. Drums, Guitar, and Bass), apply the information from analysis to synthesize supporting tracks for the soloist.

Fallbacks: Our contingency plan is to limit the control the user has of choice of instruments for synthesis. Further, using less complex synthesis (i.e. varied rhythms and technical chord structures) will be traded off for more systematic chord progression.

**Application**

*Stephen, Ben, Edward*

Plan A: Mediate between the user, analysis and synthesiser, via GUI.

Fallbacks: Limit the interaction in the GUI by limiting the options of synthesis and saving/importing/exporting synthesis.

*Note:* depending on fallback needs of the analysis and synthesis groups, the application may need to accept more inputs for the program to run smoothly.
**Schedule**

Progress Report/Demonstration/Presentation 11/19-11/22  
Enhancing Analysis 12/2  
Synthesizing different tone qualities 12/2  
Functional Application UI 12/4  
Final Project Submission 12/6  
Final Web Page 12/20  

**Resources/Budget**  

The **Echo Nest** is an open-source API that can take a .wav file and analyze it for basic musical information such as key, tempo, and rhythm, as well as more abstract data such as energy, liveness, and danceability. It is non real-time analysis.  

The Echo Nest-$0  

The **TarsosDSP** is an open-source API that analyzes music in real-time and makes use of multiple live pitch-detection algorithms. It also builds on top of Java’s existing sound API and allows for sound synthesis. This API will provide the tools to make our project real time, enabling it to adapt with the user.  

TarsosDSP API-$0  

**Recent Developments**  

In light of the extreme difficulties we have encountered facing accurate and simultaneously live feedback, the overall goals of our project have necessitated a fair deal of trimming. We have realized that the scope of our project was relatively steep and acknowledge the low probability of our initial concepts to be achieved. That said, we have scaled back our aims accordingly and plan to produce a meaningful final product as the final deadline approaches.
References


