S.T.R.U.M.

Devin R. Mix, Jacob R. McLaren, Robert A. Quito

Advisor: Cheryl Resch

University of Florida

Submitted 1/29/2023

Abstract

Learning how to play the guitar can be an intimidating task for beginners. Trying to learn off of professional studio recordings of songs can be an impossible task for beginners due to pitch correction, while other video games such as Guitar Hero and Rocksmith don’t provide precise feedback aside from how well they played with a given score. Our product aims to help users learn how to play the guitar by providing feedback on which strings and frets to strum as they play through a given song. The user is also given feedback for their pitch, tempo, and dynamic accuracy based on their performance. Our project uses Fast Fourier Transformations to analyze pitches and determine if the user is playing the correct notes at the right time. It will be built using Python and will require a PC soundcard that an electric guitar would connect to as an input for the software.

Table of Contents

Introduction

Problem Formation

The proliferation of readily accessible information via the internet has democratized access to instrument learning tools for an increasingly large number of people. However, the key issue with such tools is their reliance on an individual learner's willingness to evaluate their own skills, a technique which paradoxically relies on familiarity with what a strong performance should sound like. Oftentimes, students learning to play a new instrument refer to professional studio recordings of songs to guide their practice. Unfortunately, these recordings often rely heavily on pitch correction or other effects that elevate the production beyond what early music students could hope to achieve without years of training. This can lead many to focus on their distance from what is an objectively lofty goal for beginners while failing to recognize their personal progress. This can prove discouraging to many and, in turn, lead to disenchantment with the idea of playing an instrument at all.

The electric guitar is an instrument particularly affected by this reality. Though considered highly accessible by many, the process of learning it can prove confusing to beginners in the digital age. While beginners past would rely on studio-adjusted performances as their target goal in the same way as today, beginners of the modern era are bombarded with dozens of sources which preach "correct" ways of playing the instrument which may or may not be best suited for the individual learner. Furthermore, the guitar's existence as a chorded instrument makes identifying issues with individual pitches difficult for those without a musical background, and its primarily analog nature prevents it from interfacing directly with most assistive technologies in the way that, say, a MIDI-interfacing keyboard could.

Combined, these factors illustrate a gap in the current software market, as there exists no software for electric guitar learners specifically that is capable of assisting the learner's performance, evaluating their technique, and providing suggestions for improvement the way that a professional instructor would.

Project Rationale

Our team believes that the best approach to solving the problem outlined above lies in the development of free, open-source software. The reasons for this are as follows:

- Open-source software is fundamentally customizable and will thus enable learners to modify the tools that they use to best fit their needs.
- Most computing is considered ubiquitous in the modern era, and its usage is considered an enabler of access to otherwise inaccessible learning tools for many.
- A free, software-driven approach will provide a previously unexplored method of teaching guitar, in which specific performance feedback can be provided to users without the involvement of any third party.

While previous software-based attempts at lowering the barrier to entry of guitar learning such as Guitar Hero and Rocksmith have found commercial success, the primarily game-driven nature of these tools significantly hinders their ability to effectively serve as educational assistants. Of these, Rocksmith arguably comes closest to providing an educational experience, but its focus on teaching users through progression (i.e., evaluating user performance as a song is played) places less emphasis on the user's goals and more on entertainment. This fact is only amplified by its failure to consider aspects such as dynamics that would be critical to professional performance.

Proposed Solution

To best provide an enhanced, software-driven experience for beginning guitar learners, our team proposes the development of S.T.R.U.M., a novel software product capable of recording and evaluating the accuracy of a user's guitar performance. Said evaluation will include consideration of the user's tempo, pitch, and dynamic accuracy. This will be supplemented by on-screen guides which will inform out-of-practice users of which chords they should play and in what manner. These guides will be generated from files using a novel markup language capable of being modified by a text editor. These files will be used to guide performance evaluation as well. Should a user be satisfied with their performance, a means of exporting it to a file for later use (e.g., in a digital audio workstation) will be provided as well.

Contribution toward software engineering applications

This solution will enhance the software engineering field by applying techniques of signal processing in a practical manner to the field of music education. By offering our solution to the open-source community free of charge, we invite future collaborators to apply its underlying analytical tools to other applications, particularly those which may increase the accessibility of other instruments. We also encourage the usage of said techniques in other areas of audio engineering, such as pitch correction or sound effect design.

Objectives

Our project aims to provide a tool for new learners of the electric guitar. This tool will provide playback of a song alongside on-screen prompts stating which strings and frets a user should strum at a given time. Following a song's completion, users will be provided feedback of their performance, specifically its pitch, tempo, and dynamic accuracy. These feedback tips will focus on the accuracy of the user's playing rather than the sound produced by studio effects, thereby emphasizing the process of learning and gaining confidence in one's abilities rather than the aesthetic appeal of the recording produced. In total, the main objective of the software is thus to enhance the experience of independent guitar learning and encourage practice above all else. This will also have the effect of increasing the accessibility of guitar to those who lack access to a professional teacher so long as they are able to acquire a guitar.

Literature Survey

Our project will operate on the assumption that standard A4 = 440 Hz. tuning and equal temperament will be used. Under this assumption, the MIDI standard outlines a means of converting the frequency produced by an instrument to a MIDI number and vice versa. Under this system, each increase by one in the MIDI number correlates with moving up one equal tempered semitone in pitch (Wolfe). MIDI numbers thus allow for a concise encapsulation of the pitches which a user is intended to play on an instrument.

In analyzing the actual pitches played by the user, our project will make use of the Fast Fourier Transformation (FFT). The FFT is an encapsulation of the Discrete Fourier Transformation, whose principles allow for the Fourier analysis of a set of samples rather than a continuous function (Butz, 89). Critically, the FFT is limited by what is known as the Nyquist frequency, which sets a mathematical upper bound on the frequencies whose amplitude can be analyzed. This upper bound is dependent on the distance between discrete samples taken in the time domain (Butz, 98). In the practical context of our project, the Nyquist frequency will be equal to one half of the sampling rate of the guitar’s input device. In rare cases, this could limit the measurable frequencies to less than those capable of being produced by a guitar, which merits consideration in development.

The effect of sympathetic resonance is an important consideration when handling frequency data produced by the FFT. Sympathetic resonance is the effect in which periodic strikes of an object (often through sound waves) produce recurrently amplifying oscillations in the object itself (Helmholtz, 36). In effect, this means that the vibration of one string on a guitar may produce harmonic oscillations in another. However, “When the pitch of the original sounding body is not exactly that of the sympathizing body, or that which is meant to vibrate in sympathy with it, the latter will nevertheless often make sensible sympathetic vibrations, which will diminish in amplitude as the difference of pitch increases" (Helmholtz, 39). That is to say that any strings currently tuned to a pitch other than that of the sounding string will resonate with a lower volume than those tuned to the same pitch, and that that amplitude will decrease the further that the resonating string's pitch deviates from that of the sounding body. Notably, the harmonic under tones of the resonating string are most likely to create sympathetic vibrations, though their amplitude decreases as the difference in frequency between the sounding and resonating strings increases (Helmholtz, 44). In practice, these principles suggest that the amplitude of the sounding string's frequency will be greater than that of its resonating counterparts. Still, consideration must be given in development to the potential for two complimentary frequencies to affect and amplify the sympathetic resonance of a single string beyond that of another.

Proposed Work

Plan of work

Our team will make use of a standard open-source pipeline. Code will be hosted on Github, with CircleCI providing testing integration to ensure functionality. Testing will drive significant portions of development, with the exception of interactive segments which will require manual review due to their need for a physical guitar and human user. To supplement our testing practices, SonarCloud will be employed as a static testing tool. This tool will help ensure that no significant vulnerabilities exist in the software product itself, thereby bolstering user confidence in the final deliverable.

Technologies to be used

The primary language for our project will be Python 3. Interfacing of the project with an audio input device will be accomplished using the PyAudio library, which provides means of collecting samples from audio input devices and writing samples to audio output devices. The input device used will be a simple PC soundcard with a multipurpose jack. This jack will be connected to an electric guitar via a 3.5mm male-to-male audio cable and an 0.25” male to 3.5mm male adapter. In testing, the guitar may be substituted for an audio output device capable of reproducing certain sounds repeatedly and consistently. While specific soundcard and guitar combinations will be used, no specification will be provided for either to encourage compatibility with the greatest possible number of devices. Analysis of the incoming samples will be accomplished using the library SciPy, which provides functions for performing a Reverse Fast Fourier Transformation on an incoming amplitude signal. SciPy’s output is in the form of objects specified by the NumPy library, thereby necessitating its use. Conveniently, NumPy’s performant functions will also serve to benefit the program’s efficiency. A currently undetermined library capable of displaying an interactive GUI to the user will be employed as well.

Analytical techniques

In analyzing user-provided audio signal data, our project will primarily refer to the pitches played by a user at a particular time. These will be determined using the Fast Fourier transform. The results of the transformation will be compared to those generated by a signal generation tool in order to determine whether the relative volume of the intended pitches is correct. In effect, this will amount to determining whether the user played the correct notes at the correct times with the correct intensity.

Given the particularly central nature of this comparison algorithm, testing will centrally focus on it as well. Computer- and human-generated samples will be provided to the algorithm during testing to determine whether the results it produces fall within expected tolerances. Without proper clearance of the expected tests, the algorithm will not be deemed complete.

Project Plan

The following table outlines the intended completion targets for the project. This includes both those specified by the senior project course and those intended to motivate progress internally. Note that additional advisor meetings may be added, and internal deadlines may be adjusted to best suit the needs of the team.

Date	**Objective/Deadline**
2/9/2023	Second advisor meeting
2/12/2023	Targeted deadline for completion of basic Fourier transformation functionality. At this point in development, basic work should be underway for the core backend features of the software, namely user performance analysis and accuracy determination.
2/19/2023	Presentation #1; Completion deadline for architecture diagrams, UI design schematics
2/23/2023	Third advisor meeting
3/9/2023	Fourth advisor meeting
3/23/2023	Fifth advisor meeting
3/19/2023	Targeted deadline for completion of basic GUI functionality in accordance with the UI design schematics. Incorporation of the previously outlined Fourier transformation functionality should be underway.
3/26/2023	Presentation #2; Deadline for having project walkthrough, test plan both finalized. This deadline will coincide with our first examination SonarCloud's static testing results, which will be presented to Prof. Resch in accordance with her request.
4/6/2023	Sixth advisor meeting
4/9/2023	Targeted deadline for complete functionality of most aspects of the software product. Testing should be completed, with all or an acceptable number of tests passing. End-to-end, user-driven testing should occur with little to no need for intervention on the part of the development team. The product should be in a release-ready state.
4/16/2023	Final presentation; Deadline for having project walkthrough finalized

	This deadline will coincide with our second examination SonarCloud’s static testing results, which will be presented to Prof. Resch in accordance with her request.
4/20/2023	Seventh (final) advisor meeting
4/21/2023	Poster session; Deadline for completion of all functional aspects
4/26/2023	Project submission; Deadline for completion of submission document, project documentation

Conclusion

The goal of this project is to provide a tool for new learners of the electric guitar. It will provide a playback of a song and the strings the user must follow. After the song is completed, several tips and feedback will be given to the user on how to improve their playing skills. This project will be developed using a standard open-source pipeline, with the code hosted on Github. To make sure the tool works as wanted, it will be tested repeatedly. Since it will require the use of a guitar to practice in person, SonarCloud will be used as a static testing tool. To make sure all parts of the project are completed in time, certain parts of it will be spread out between members and set to be developed by a certain date. This will allow all parts of the project to be completed and tested separately and in time.

References

Butz, T. (2006). *Fourier transformation for pedestrians*. Springer.

Helmholtz, H. (1954). On the Sensations of Tone as a Physiological Basis for the Theory of Music (2nd ed.). Dover.

Wolfe, Joe. (n.d.). *Note names, MIDI numbers and frequencies*. https://newt.phys.unsw.edu.au/jw/notes.html