Fundamental Frequency Estimation for Non-Interactive Audio-Visual Simulations

Rahul AGNIHOTRI, Romain MICHON and Timothy S. O'BRIEN
CCRMA (Center for Computer Research in Music and Acoustics),
Stanford University,
Stanford, CA, 94305
{ragni, rmichon, tsob}@ccrma.stanford.edu

Abstract

This paper aims to demonstrate the use of fundamental frequency estimation as a gateway to create a non-interactive system where computers communicate with each other using musical notes. Frequency estimation is carried out using the YIN algorithm, implemented using the ofxAubio add-on in openFrameworks. Since we have used only open-source technologies for the implementation of this project, it can be executed on any platform: Linux, Macintosh OS or Windows OS. As a simulation, this system is used to depict a conversation between people at a round-table event and presented as an audiovisual art installation.

Keywords

Pitch Estimation, MIR, YIN algorithm, Aubio, Audio Simulation.

1 Introduction

Pitch detection algorithms have been used in various contexts in the past:

- audio editing programs (pitch correction and time scaling) such as Melodyne[1],
- analysis of complicated melodies of world music cultures (Indian Classical music),
- music notation programs like Sibelius[2],
- MIDI interfaces such as the Roland GI-20 to get data from guitar MIDI pickups.

Since it lies firmly within the domain of music information retrieval (MIR), pitch estimation has many applications in recommender systems, sound source separation, genre categorization and even music generation. With the popularity of machine learning, neural-networks, and data mining, audio signal processing tools are utilized more and more to create user-specific systems. When considering applications for pitch detection and source separation, we often consider the example of identifying individual speakers at a round-table event. Computers, unlike humans, have a difficult time identifying the words of a particular person if multiple people are communicating with each other simultaneously. Building on that concept, we simulate a conversation between multiple computers using musical notes. The YIN pitch detection algorithm is employed to detect trigger notes, to which other computers in the network respond.

Since we desire a continuous conversation, we must use an efficient detection algorithm with the following features:

- real time response,
- minimal latency,
- accurate identification in the presence of noise.

We need to be careful about both the latency of the attack and of the pitch detection algorithm since if a note is played, the human ear needs at least seven periods of a waveform to identify its pitch. Hence, note onsets and note pitches are not directly related [3]. Additionally, we must ensure that the pitch recognition algorithm is reasonably robust to the sort of noise which is inevitable in a performance scenario.

After comparing different methods for pitch estimation, as in [3], we chose the YIN algorithm for its real-time tracking ability. YIN is a time-domain algorithm based on the autocorrelation method for estimation. [2]. Using the common autocorrelation method, its error rates are analyzed and corrected for every new iteration to ensure the best possible accuracy. Using YIN is beneficial since it can accurately analyze higher frequencies which we might use as trigger notes in our system. To make sure that we have the lowest possible pitch identification latency and have a very small frame size for incoming

[1] `http://www.celemony.com/en/melodyne/what-is-melodyne`. URLs in this paper were verified on Feb. 16, 2017.

[2] `http://www.avid.com/sibelius`

audio, we use the YIN algorithm implemented in the aubio framework [1], extended further as an addon in openFrameworks[3] for our computer network.

2 Methods and Implementation

Since fundamental frequency identification works well on monophonic sounds (e.g., a guitar solo or any wind instrument), it was decided to use this approach to reduce the problem of incorrect triggers for the other computers in the network. Individual notes are generated one after the other at randomized tempos (to mimic the prosody of human speech) in a particular musical scale. Every other computer has its own "voice" which does not overlap with that of any other computer in the network. At a given point in time, multiple "people" can "speak" simultaneously.

The YIN algorithm is accurate enough to sufficiently identify the trigger notes at any given time within the chaotic yet pleasing tone of the conversation, and the computers react accordingly. Visual feedback is provided to portray whether a computer is voicing itself or is remaining silent in response to the trigger note. For the test system, three computers, each with their own voice, constituted the network.

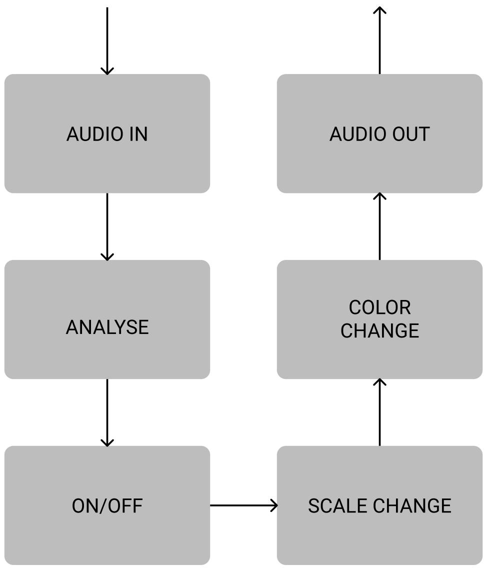

Figure 1: Overview of the system

2.1 Audio

Each voice for the "speaker" in the system is a musical scale. For our demonstration, each computer was set to the G, C and D Major scale respectively. The choice for these particular scales was arbitrary and baseless. The `ofxStk`[4] add-on was used to generate the sounds using the `Moog` synthesis class. The generated sound has a reverb effect applied to it in order to create a wider stereo image when all computers are in place.

In order to mimic the characteristics of human speech, there is no fixed tempo for the system. When a particular trigger note is heard by a computer, it goes silent and shifts its scale by an octave higher or lower and also changes its corresponding trigger note in order to keep the whole system ambiguous. The ambiguity lies in the fact that when all computers are communicating simultaneously, it is difficult to identify what the actual trigger note is and maintains the illusion of an improvised conversation. The exact flow of the network is represented in Figure 1. This flow remains constant for any new computer added to the system.

2.2 Graphics

Graphical feedback is used to convey whether a particular computer is active or silenced. We used the `ofxParticles` add-on to implement particle physics in order to visually represent the current state of our system.

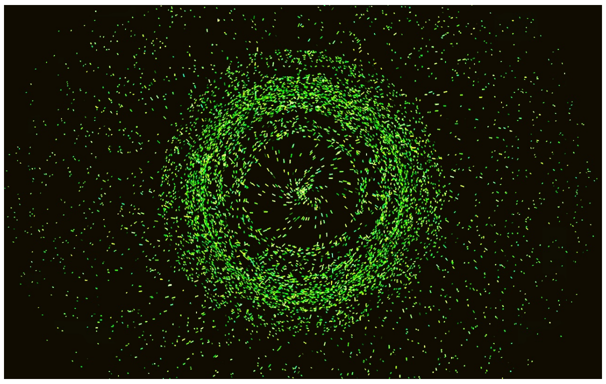

Figure 2: Screenshot of the particle physics graphics

White particles at any given instant represent an "in-active state" of the computer, and colored particles are used when the computer is active (as shown in Figure 2). The color of the particles for each computer changes when it becomes active after being inactive for a random amount of time, as if joining the conversation

[3]See `https://github.com/aubio/ofxAubio` and `http://openframeworks.cc/`

[4]`https://github.com/Ahbee/ofxStk`, which encapsulates the original Synthesis ToolKit, `https://ccrma.stanford.edu/software/stk/`

again to state its opinion. Particles are emitted from the exact center, spiral outward, eventually fade out. There is a pseudo-gravitation effect that makes the particles orbit around the center of the screen after their spiral trajectory. We designed this visual tool to create a psychedelic effect for the audience.

2.3 Evaluation

When presenting this system to a group of observers it was noticed that despite having an underlying pattern to the changes in scale and trigger conditions, they could not detect this and the audience expressed that they believed the computers were having a conversation, albeit through musical notes. The audience also responded that having a visual feedback gave each computer a unique personality.

3 Future Work

The future scope of this project involves the integration of machine learning in order to implement musical improvisations as a response to the trigger conditions. This would make the whole system more expressive (i.e., trivial actions such as having the computer go silent or be active, etc.). Polyphonic sound identification is also a viable addition to have a more immersed experience of musicians improvising with one another.

4 Conclusion

In this paper we presented pitch estimation as a tool for a musical performance system. Since the future scope does involve the use of machine learning and data mining techniques, as is the custom with music information retrieval, this was a relevant stepping stone. Presently, this system is being modified to include multiple triggers for the whole system. We are trying to move away from the initial condition of having only one computer respond to a single trigger note but have multiple computers react to more than one trigger added into the network.

5 Acknowledgements

Many thanks to the CCRMA community for its support.

References

[1] P Brossier. Automatic Annotation of Musical Audio for Interactive Applications. *Centre for Digital Music Queen Mary University of London*, Diploma of(August):215, 2006.

[2] Alain de Cheveigné and Hideki Kawahara. YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

[3] P De La Cuadra. Efficient pitch detection techniques for interactive music. *International Computer Music Conference*, pages 403–406, 2001.