A Framework for Developing Music-Generated Games

Erik Azzarano
Advised by Sreepathi Pai (CS)

Introduction

Music and sound has been an integral component of video games, but mainly in two simple forms:

- *Adaptive:* games with sounds or music that play at calculated points
- *Interactive:* rhythm games like "Guitar Hero" [4] and "Rock Band" [7]

However, this work focuses on a less commonly produced type of music game, namely, reactive, and aims to create a framework for developing games of this nature:

- ***Reactive: a* game where its mechanics react to the features of music**

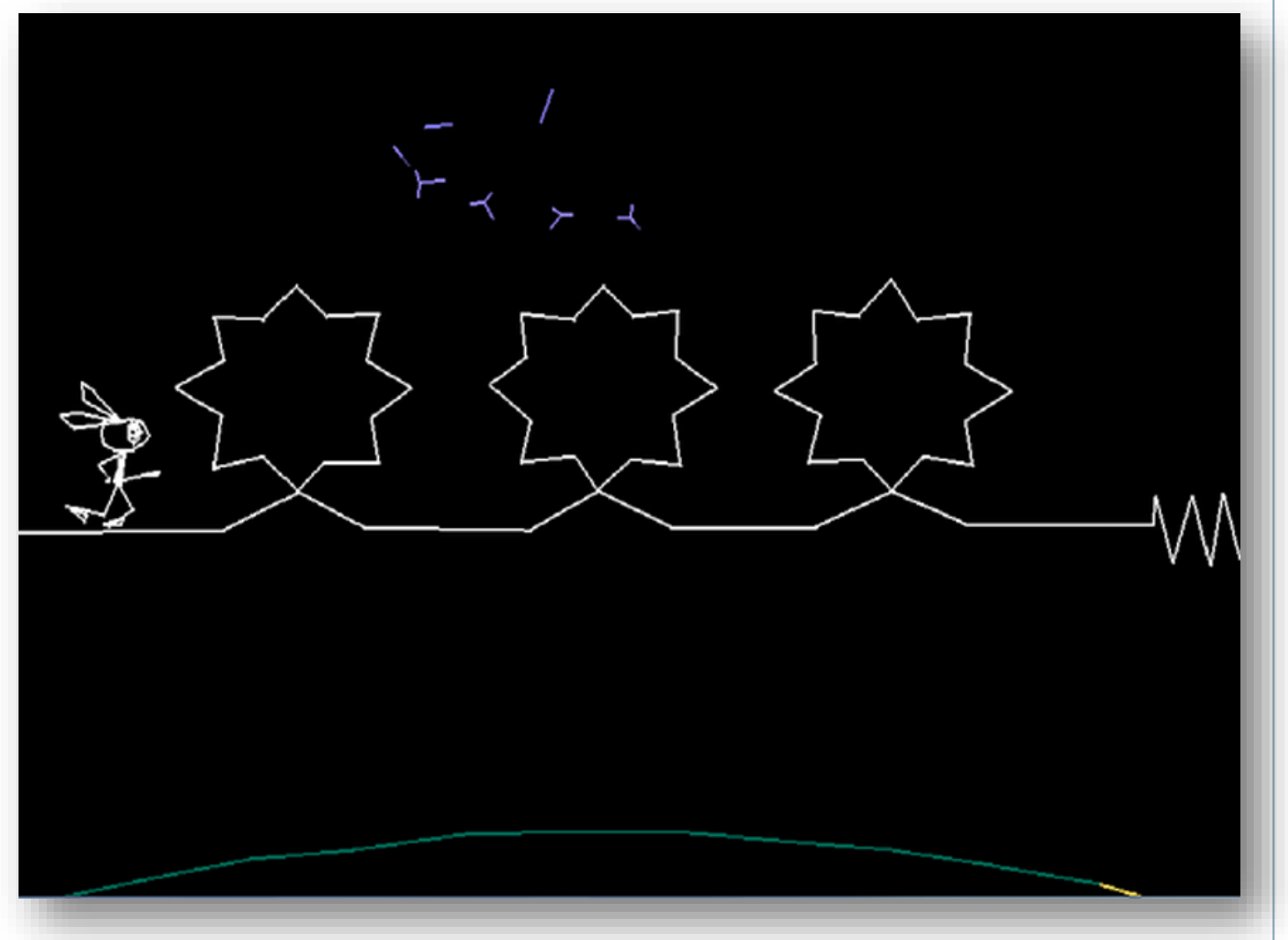

Figure 1: Gameplay from "Vib-Ribbon" [9], a 1999 music-reactive game where the level is created by a processed music file

Background

Figure 2: Gameplay of "Audiosurf" [1] where the player collects colored blocks that are created in time with the music

Previous literature has looked into making music-generated games, where audio data, rather than predetermined values, drives the game:

- "Audiosurf" is one of the first rhythm games, where the game level is created based on a selected audio file [1] (See figure 2)
- "Briquolo" [2], similar to the game "Breakout," has mechanics like speed and paddle size change based on tempo and amplitude [4]
- "Cello Fortress" analyzes live input from a cello at different frequencies and amplitudes to control a tower-defense game [3]

Framework

The key to developing a music-generated game is to establish an intuitive mapping between audio features and game mechanics [5] (See tables 1 & 2):

- Similar aspects of audio and games should be paired, such as tempo (or amplitude) and game speed
- Musical events such as beats should create objects or start actions, as they occur periodically and temporarily
- More consistent states of music such as mood should affect the look and feel of the game space
- How little or how much an audio feature affects a game parameter must be carefully tested
- It is important to test the game with a single song of a single genre to make sure it is working before expanding to new songs and new genres
- Overall, what happens in the music should have an obvious relationship with what happens in the game world

Table 1: List of potential audio features

Audio Features [5] [6]
Tempo (speed of the music)
Beat/Note Onsets (sets tempo or rhythm)
Amplitude (intensity of the music or frequency)
Mood (emotion associated with the music)
Event (temporary; periodically occurring)
State (constant; consistently occurring)
Transition (movement between states)
Others (lyrics, vocals, timbre, harmony, refrain, chorus, instrument solo)

Table 2: List of main game mechanics

Game Mechanics [8]
Space (places that exist in the game)
Time (how the game moves forward)
Objects, Attributes, and States (people, places, and things in the game)
Actions (What can be done in the game)
Rules (Defines the other mechanics)
Skill (abilities needed to play the game)
Chance (provides surprise and uncertainty to the other mechanics)

Prototype: "Audio Attack"

An example of the framework was created in the style of a 2D arcade-like Shoot 'em Up (See figure 3):

- It is built with the Processing programming language and the Minim sound library
- It uses the computer's audio output as its input, so streaming music on or offline will create the game
- A Fast Fourier Transform (FFT) analyzes the audio and the extracted data alters specific game parameter (See table 3)
- The threshold for beat detection and amplitude can be manipulated for more control of how the game plays
- The player's goal is to dodge or shoot incoming enemies for the entirety of a song

Table 3: Mappings between audio features and game mechanics for "Audio Attack"

Audio Features Used		Game Mechanics Used
Beat/Note Onset	→	Enemy spawn, background object color and size change
Amplitude	→	Enemy speed, player reload speed
Frequency Bands	→	Location/type of enemy when spawning

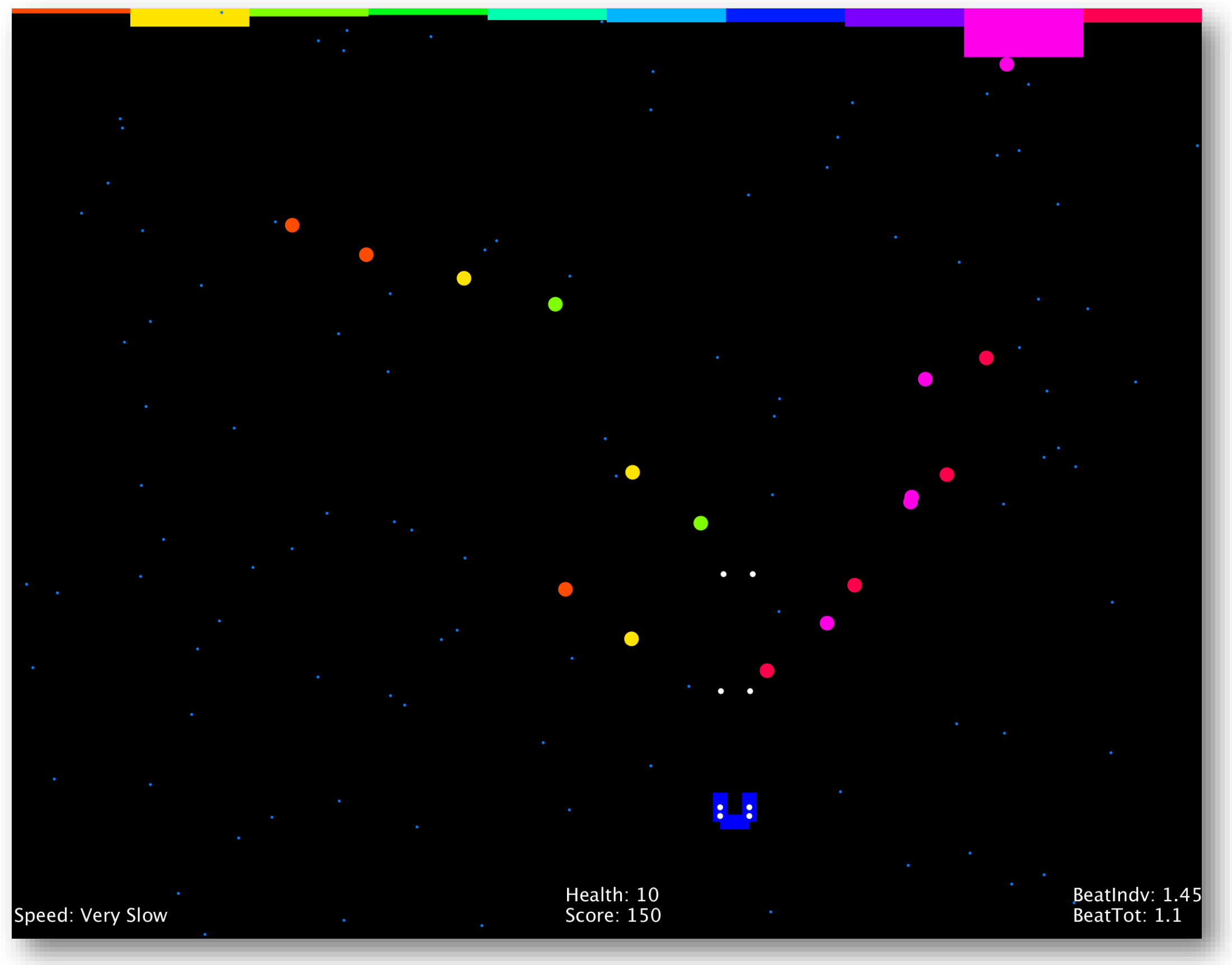

Figure 3: Gameplay from "Audio Attack"

Future Direction

The proposed framework should enable researchers to perform experiments testing the viability of music-generated, or music-reactive games for the game industry in the following ways:

- Have players test the game and provide feedback about how it compares to a non-music-reactive game
- Make new games or modifying existing games based on the framework
- Expand upon the mappings offered by this framework and find which ones work best
- Use more sophisticated audio analysis software to collect more information about the audio
- Explore other platforms such as web or mobile, and connect it directly to a music-streaming website

References

[1] Audiosurf, Invisible Handlebar, 2008
[2] Briquolo. http://briquolo.free.fr, accessed on July 17, 2018.
[3] Cello Fortress Web Site. http://www.cellofortress.com, accessed on July 17, 2018
[4] Guitar Hero Web Site. http://www.guitarhero.com, accessed on July 17, 2018.
[5] Juha Arrasvuori, Jukka Holm, Background music reactive games. Proceedings of the 14th international academic MindTrek conference: Envisioning future media environments, p.135-142, October 06-08, 2010, Tampere, Finland
[6] Jukka Holm, Kai Havukainen, Juha Arrasvuori, Personalizing game content using audio-visual media, Proceedings of the 2005 ACM SIGCHI international conference on advances in computer entertainment technology, p. 298-301, June 15 - 17, 2005, Valencia, Spain
[7] Rock Band Web Site. http://www.rockband.com, accessed on July 17, 2018.
[8] Schell, J., The Art of Game Design. Natick, MA, CRC Press, 2014
[9] Vib-Ribbon, Sony Computer Entertainment, 1999

Acknowledgements

A special thanks to:

- Prof. Al Biles and Prof. Ian Schreiber at Rochester Institute of Technology
- Prof. Ehsan Hoque and ROC HCI
- The Kearns Center staff
- My faculty mentor Prof. Sreepathi Pai

This research was funded by the National Science Foundation.