Daniel Holmes

2023-2024

Abstract

The player movement conductor system is a system that is designed to translate the motion of a player character into audio response from the games’ soundtrack. The more fluid the motion of the player character, the greater the fluidity, audibility and complexity of the sound produced. This system is designed to work best with physics-driven character movement, with each limb controlling one instrument/instrument group that makes up the background music.

TECHNICAL DESIGN DOCUMENT

Player Movement Conductor system

Table of Contents

[Glossary 1](#_Toc160112957)

[Problem 2](#_Toc160112958)

[Introduction 2](#_Toc160112959)

[List of Features 3](#_Toc160112960)

[Engine Requirements 3](#_Toc160112961)

[Audio and Visual Requirements 3](#_Toc160112962)

[User Setup Guide 3](#_Toc160112963)

[UI of the System/Tool 3](#_Toc160112964)

[Prototypes 3](#_Toc160112965)

[Polished System/Tool 4](#_Toc160112966)

[Add more headers if/as appropriate. Make sure it is clear how to use your system or tool. 4](#_Toc160112967)

[Bibliography 5](#_Toc160112968)

[Appendices 6](#_Toc160112969)

[Appendix 1: Title 6](#_Toc160112970)

[Appendix 2: Title 6](#_Toc160112971)

# Glossary

UE – Unreal Engine

# Problem

Musical scores in games are often set in stone, perhaps with set triggers within the game that cause certain tracks from the score to play at certain moments, but no fluidity of the score beyond that. This creates a very static soundscape concerning when tone is changed and what kind of tone is portrayed to the player at specific points, with little of the player’s actions affecting what kind of music is played and, more importantly, how that music is played and how it may contradict with the actions taken by the player. Particularly in cases where the music is attempting to create a dramatic or tense tone, but the player character is just stood still with no response to this change in tone, can ruin immersion.

This is the problem I will be trying to solve, through the creation of a music system that reacts directly to the movement of the player character. This system would hopefully be able to produce results akin to how musical scores were used in Charlie Chaplin’s silent films, where the score would seemingly respond to the movement of the actors, therefore leading to their movements establishing the tone of the film. The system I would create would ideally do this dynamically, making the player and their character a ‘conductor’ to the games’ soundtrack.

# Introduction

The system I will create directly translates player movement into audio feedback, specifically musical feedback, making the motion of the player character control the frequency, pitch, volume etc. of the musical score within the game. The more fluid the motion of the player character, the greater level of musical fidelity they will be able to produce, with each limb of the player character controlling a musical instrument/instrument group within the games’ ‘orchestra’, effectively making the player the ’conductor’ of said orchestra. The more fluid the motion of the player character and its individual body parts, the more harmonic each of the different instruments/instrument groups will become. This system will of course work best with a game that gives players directly control over the individual limbs of their character, so a game akin to QWOP, Human: Fall Flat or Octodad: Dadliest Catch would likely work best for utilising this system. The potential for this system to support a game that plays into slapstick comedy is also evident by both the games listed previous, all of which feature some form of slapstick comedy element, as well as the systems inspiration from Charlie Chaplin films, which are mainly slapstick comedy.

This kind of adaptive music system in response to player input is likely best paralleled to the music system found within the game “Ape Out”. This system is discussed in “Algorithms, apes and improv: the new world of reactive game music” by Will Betts (2019), in which the role of the music in the game is described as ‘fundamentally different from the role of music in [play along] rhythm games. Rhythm games ask you to pay attention to the music, whereas with Ape Out, the music is paying attention to you, the player.’ This is exactly what I want my system to do, albeit looking at different actions from the player to produce audio response than the actions Ape Out is looking for, the principle of the system is the same.

It was also worth looking into responsive music systems outside of games, as David Rokeby’s ‘Very Nervous System’ (1986-1990) is somewhat akin to what I imagine a fully polished version of the system I plan to create would do. Rokeby’s system is essentially an instrument you play with your body, with specific motion from your arm’s, hands, legs etc. producing different audio responses. My system will likely do more on its end to produce a pleasant sound in response to fluid player character motion, as the control a player has over their character is nowhere near as precise as someone has over the motion of their own body, but the principle of sound response to motion remains the same.

# List of Features

The ‘Player Movement Conductor’ system must be able to take in motion data from as many motion sources on the player as possible, with the system taking in multiple different data points indicating current motion of a particular body part of the player character. Any value from the player character should be able to control any aspect of the audio playback, be that volume, frequency, pitch, distortion, etc., but the system itself will recommend to developers what values should be used to control what variable of the audio. To give an example, the inertia of a specific limb may be best suited for distorting the waveform that is being played as a result of that limbs motion, so the system will by default recommend that value be used to control level of distortion.

This system should also not be a black box, with developers able to look into and fully understand the inner workings of it so that they can make better use of it to achieve the reactive audio they want. Developer control over the audio produced is paramount, and whilst there will be a basic pre-made source within the system that developers can use if they don’t want to design their own, the system will also accommodate developers who do want to customise their own responsive music.

# Engine Requirements

This system will be built in MetaSounds, so the Engine needed will be Unreal Engine. The minimum version of said engine will be UE5.2.1.

# Audio and Visual Requirements

Both headphones and speakers will be usable with this system, and some form of screen/computer will be needed to run the Unreal Engine project the system is within. Since the output of the audio is driven by controlling a character, direct input from the user will be needed for any sound to be produced.

# User Setup Guide

How to setup the tool/system, including how to setup the game engine to accommodate the tool/system process (if appropriate). All sections are a user setup guide and how to use them.

**DO THIS AT THE END**

# UI of the System/Tool

Since this tool is built using UE 5’s MetaSounds system, the UI of the tool is essentially the UI of MetaSounds itself, and utilises the gizmos offered for audio control found within MetaSounds itself.

# Prototypes

Prototype 1 – posted February 11th, 2024.

Link to Post: <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/63079-holmes-daniel-h011646k/&do=findComment&comment=853997>

Prototype 2 - posted February 25th, 2024.

Link to Post: <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/63079-holmes-daniel-h011646k/&do=findComment&comment=863667>

Prototype 3 – posted February 27th, 2024.

Link to Post: <https://digitalacademy.staffs.ac.uk/forum/index.php?/topic/63079-holmes-daniel-h011646k/&do=findComment&comment=865474>

# Polished System/Tool

Time and date of post.

Link to post.

(Submit zip file separately to DA Upload)

**On the post:**

It is ideal to have this as a single but very detailed post on the forum.

Include:

* Have the title “Polished System” or “Polished Tool”
* Show system/tool output to a polished level.
* Show variations of system/tool output to a polished level (i.e. show the range of things it can produce or do).
* Show the system requirements for your system/tool.
* State the usage of the system and how it fits into the development cycle.
* Images
* Videos
* Audio
* Reflection/evaluation ()

# Add more headers if/as appropriate. Make sure it is clear how to use your system or tool.

# Bibliography

Music Tech, Betts, W. (2019), Algorithms, apes and improv: the new world of reactive game music [Online]. Available from: <https://musictech.com/features/interviews/ape-out-matt-boch-game-soundtrack/>

Rokeby, D. (1986-1990), Interactive Installations: Very Nervous System [Online]. Available from: <http://www.davidrokeby.com/vns.html>

# Appendices

Any additional supporting documentation can be added to the appendix. This can relate to any section of the report.

It also helps if the appendix has its own content page and naming conventions.

## Appendix 1: Ape Out Gameplay Video

<https://www.youtube.com/watch?v=ed0wicQ6rFs>

[](https://www.youtube.com/embed/ed0wicQ6rFs?feature=oembed)

## Appendix 2: The Very Nervous System Demo Video

<https://vimeo.com/8120954> (May need to log into a vimeo account to view this video)