# CMP407: AUDIO PROGRAMMING

Brodie Templeton
1601797

# OVERVIEW

- This application is a small satirical horror-like game, inspired by “Death Trips” by Alberto Navarro.
- The primary focus of the game is on the use of audio to create a creepy atmosphere and unsettle the player.
- The gameplay itself is minimal, and is presented more as an interactive scene.

# AUDIO TECHNIQUES

- This application makes use of the following techniques:
  - Recorded/Edited Audio.
  - Implementation of basic audio techniques, operating on individual samples.
  - Use of at least 3 basic audio effects in an appropriate context.
  - Use of compressed audio file formats.
  - Integration with audio middleware
  - Making use of the audio facilities in a game engine (Unity).
  - Dynamic Audio
  - Synthesis of waveforms for music, sound effects and musical instruments.
  - Use of spatial audio.

# RECORDED/EDITED AUDIO

- Both the dirt and water footsteps were recorded audio that was edited in Audacity.
- The dirt footsteps were recorded at the university with the recorder by walking in gravel.
- The water footsteps were recorded in a slightly filled bathtub using a phone.
- Both recordings had each individual step sound separated and exported to allow each step to be randomly chosen each time to disguise the repetition.
- Additionally, the water footsteps were pitched down a couple of octaves to produce a deeper sound, giving the impression of a larger body of water.
- The heartbeat sounds were created using a jar lid and was recorded on a phone, with Audacity being used to separate each beat.
- The fire sound was created in the university with the recorders by rustling some paper. Audacity was used to increase the amplitude on the clip to make it louder in game.

# BASIC AUDIO EFFECTS (USES)

The gunshot sound effect, is altered to produce a simple echo. The echo makes use of a low pass filter, a delay, and attenuation.

Mixing is used to combine the three components of the synthesised harmonica.

Additionally, all synthesised audio, makes use of an envelope to have greater control of each synthesised tone.

# BASIC AUDIO EFFECTS (IMPLEMENTATION)

- The application makes use of a custom audio manager static script. This contains many utility functions for manipulating an existing audio clip on a sample by sample basis.
- The low pass filter function takes the samples list and filters out the higher frequencies, this is useful when producing the echo effect to give the impression of the source getting further away.
- The delay function takes the samples list and a delay value which is primarily used in the echo effect to add the filtered version of the sound after the given delay.
- Attenuation mainly takes place with synthesised sounds, however does also prove useful with the echo effect also, as it lowers the amplitude of sound wave.

```
//This function creates a samples list with a delay at the start
1 reference
public static List<float> AddDelay(List<float> samples, int delay)
{
    List<float> delayed = new List<float>();

    for(int i = 0; i < delay; i++)
    {
        delayed.Add(0);
    }

    delayed.AddRange(samples);
    return delayed;
}
//This function creates a simplistic echo effect utilising a delay, a low pass filter, and attenuation
1 reference
static public List<float> Reverb(List<float> samples, int delayInc, int reverbCount)
{
    List<float> reverb;
    reverb = samples;
    for (int i = 1; i < reverbCount + 1; i++)
    {
        reverb = AddSamples(reverb, LowPassFilter(AlterGain(AddDelay(samples, delayInc*i), 1 - ((1/(reverbCount + 1)) * i))));

    }

    return reverb;
}
```

# SYNTHESISED AUDIO

- The music at the climax of the game was created using a fluctuating scale of music notes and a saw tooth wave. This erratic note shift combined with the sharpness of the wave throws the player off-balance when contrasted with the relative calmness of the scene thus far.
- The win/lose sound is synthesised using a sine wave and 3 notes on a major scale for the win sound to create a happy tone, and a minor scale for the lose sound to create a sad tone.
- The harmonica is synthesised using a square wave for the base and harmonic components, and some pseudorandom noise for the noise component. The base layer uses the given note, while the harmonics and noise use the given note, one octave up. These components are then mixed to produce the finished result.

# DYNAMIC AUDIO

- The footsteps are randomised each for each step and will change depending on the surface the player is walking on.
- The dirt and water footsteps are stored in different lists, so if the player moves from dirt to water, the list being used to select the footstep is changed.
- As the fire burns out, the speed of the heartbeat sounds increase to imitate the players heartrate increasing which instils a sense of panic in the player.

# SPATIAL AUDIO

- Steam Audio is used to create the spatial audio in the game. This makes the audio sources such as the campfire and harmonica, binaural and results in their volume varying depending on the distance to the listener.
- It creates more realistic reverbs within the scene geometry and results in sounds being muffled if blocked by some geometry.

# COMPRESSED AUDIO FORMAT

- All non-synthesised audio is stored as an mp3.
- This allows for a reduced file size given the number of clips and will reduce the loading time if needed to be loaded at runtime.
- Mp3 is a lossy compression format, however there is no noticeable drop in quality and is therefore sufficient to use for this application.

# IMPORTED ASSETS

- Gunshot & Shell Casing – Unity Asset Store - https://assetstore.unity.com/packages/audio/sound-fx/weapons/weapon-soldier-sounds-pack-29662
- Creepy Game Over Screen Music – Unity Asset Store - https://assetstore.unity.com/packages/audio/ambient/ambient-horror-sound-fx-free-64527
- Ambient Cave Sounds – Online - https://www.zapsplat.com/sound-effect-category/caves/
- Fire Particle Effect – Unity Asset Store - https://assetstore.unity.com/packages/3d/props/tools/low-poly-survival-essentials-109444
- Muzzle Flash – Unity Asset Store - https://assetstore.unity.com/packages/3d/props/guns/modern-guns-handgun-129821
- The code for synthesis using wave types other than a sine wave, in addition to the envelope/oscillation code was adapted from - https://github.com/SFML/SFML/wiki/Synth

*All other art assets were created by me, and all other audio assets were either recorded or synthesised by me