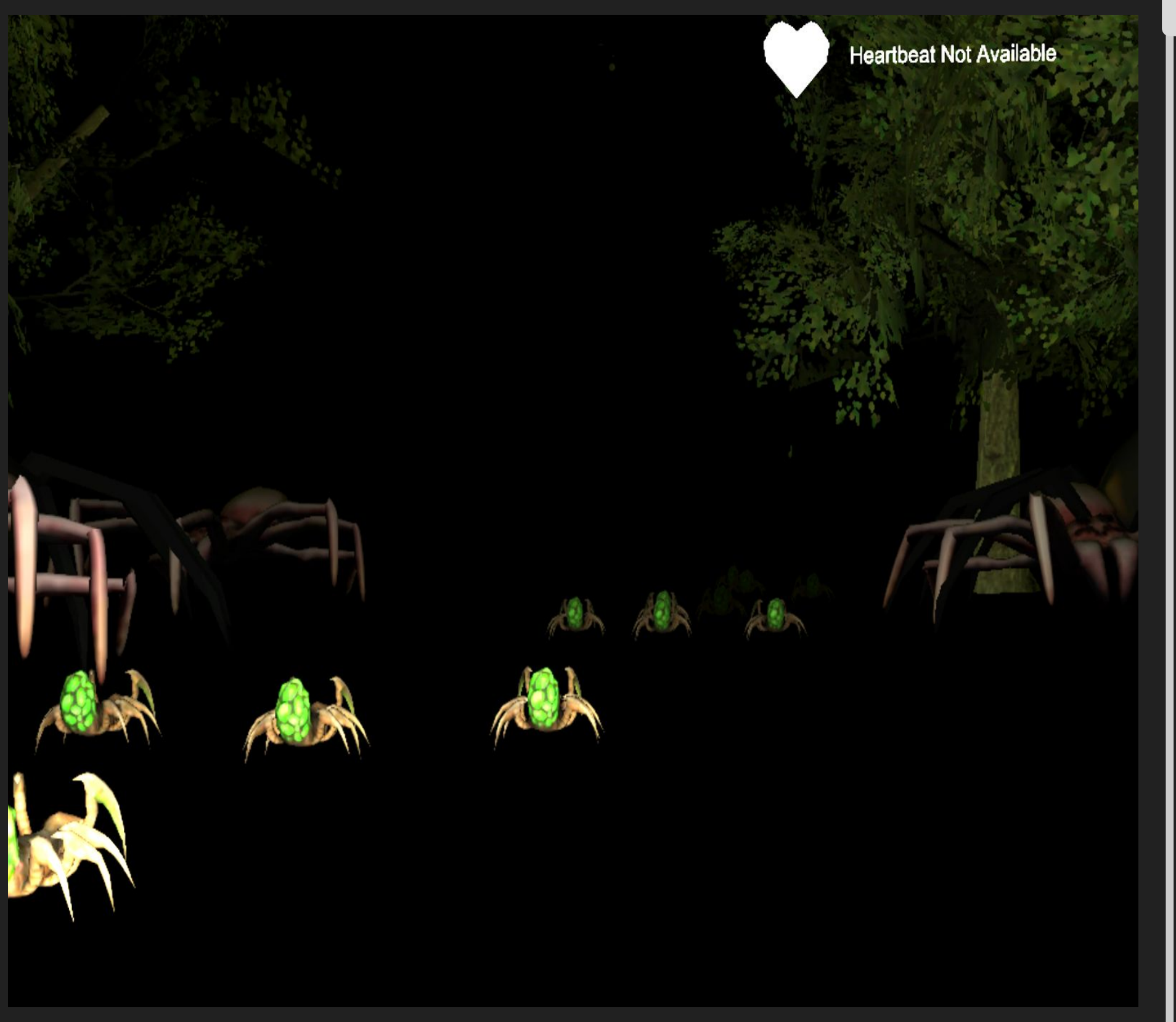
Heartbeat Horror

Team Polsky
Mitchell Neal & Jacob Brown

Scene 1: Forest



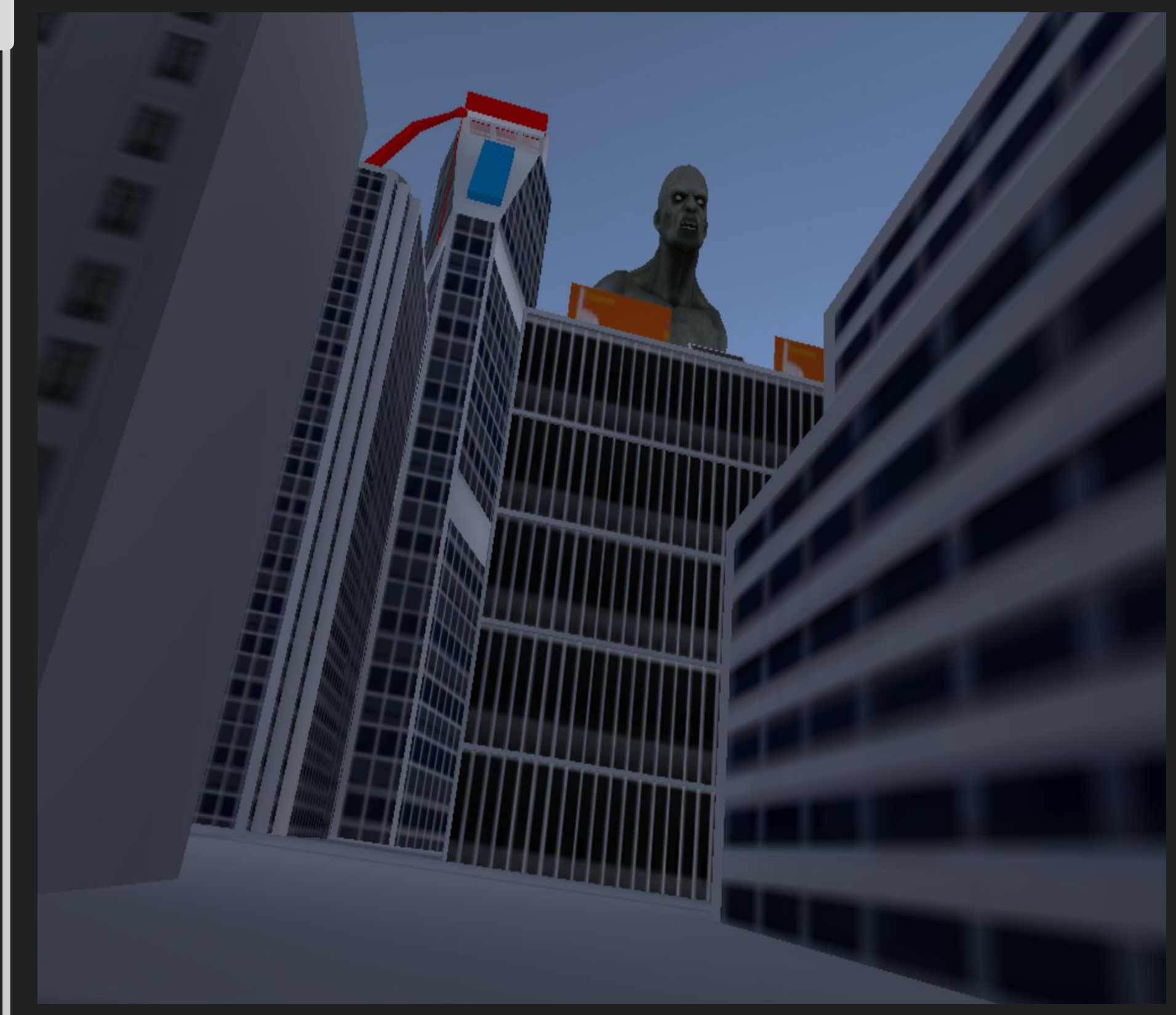
Dark forest full of procedurally generated spiders crawling around

ABSTRACT

Despite the prevalence of vast sensing capabilities in consumer devices, video games are lacking in their ability to detect the quality of a player's game experience, and to respond to it. To address this shortcoming, our goal was to create a platform that allows games to sense a player's involuntary responses during the game. Doing so could allow the game to change itself in real time based on this data to vastly improve players' experiences.

To demonstrate our work, we created a Virtual Reality horror game called Heartbeat Horror. The game senses a player's heart rate in order to determine their fear level, and changes the game accordingly. The game can detect what game objects the player's fear level responds to, and generates more of these objects dynamically during gameplay. Additionally, the player can lose the game if their heart rate gets too high, adding a unique extra element to the gameplay experience since the player must learn to control their fear response to improve their score.

Scene 2: Urban Rooftop



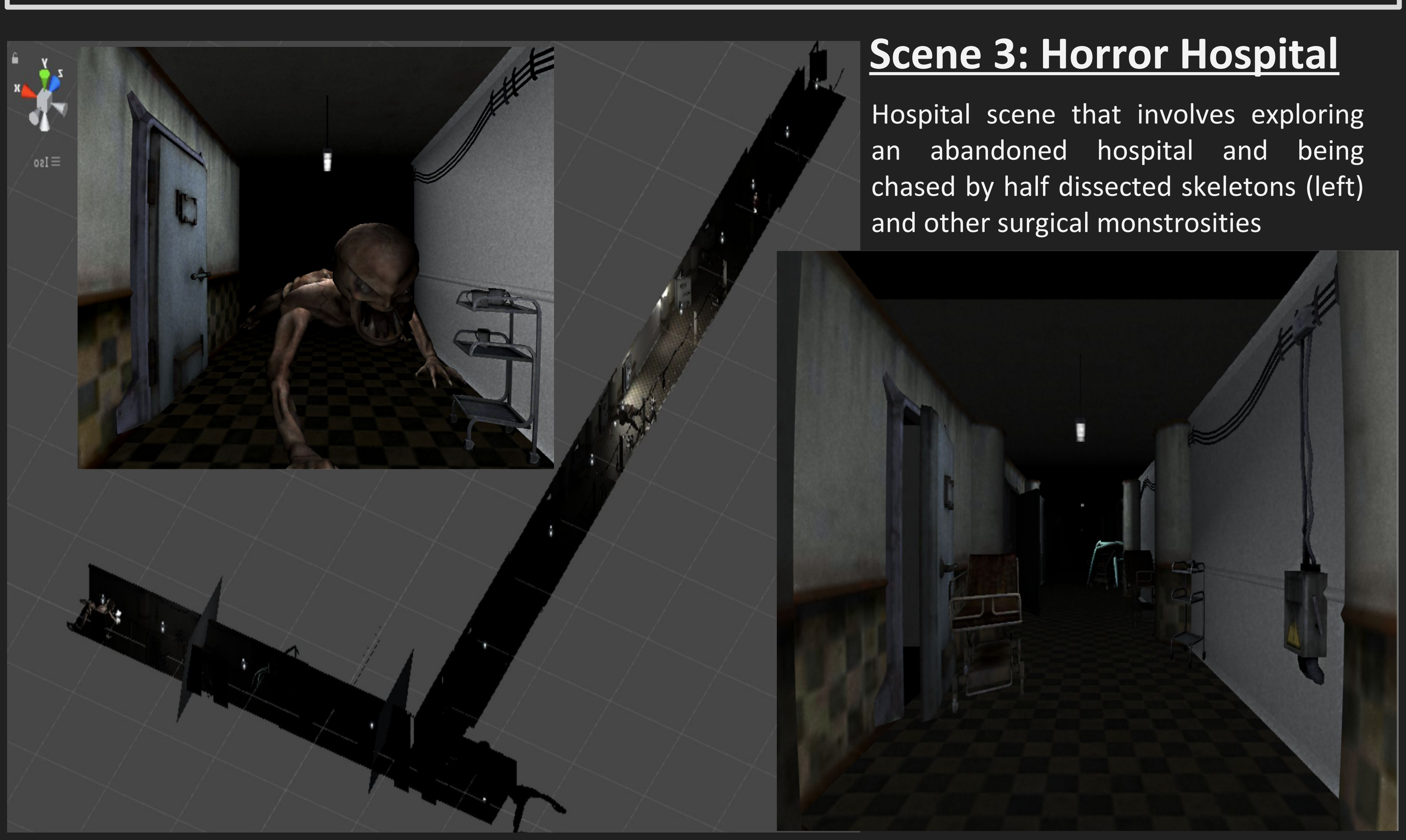
An urban rooftop scene, in which a player has to jump between buildings and ledges while being chased by a giant monster

OBJECTIVES

We developed a platform for dynamic gameplay changes based on involuntary player responses through the following objectives:

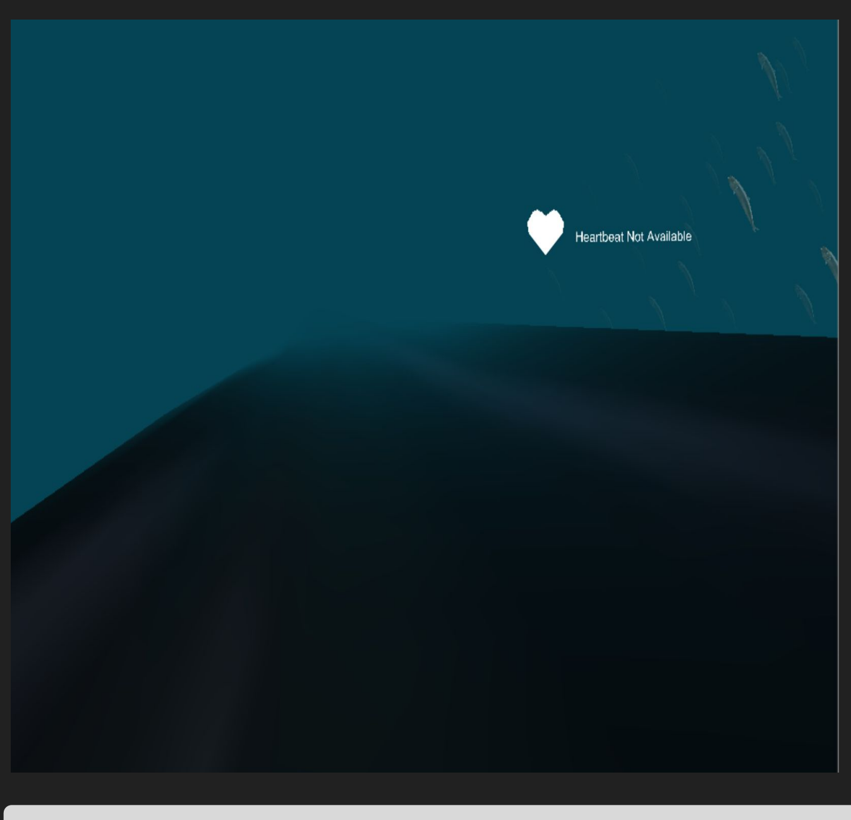
1. Create a basic horror Virtual Reality game which senses a player's heart rate through an accessory wearable device.
2. Configure our horror game so that if a player's heart rate goes a set percentage above their resting heart rate, the player will lose the game.
3. Make our horror game capable of changing in real time in response to a player's emotional responses.

Scene 3: Horror Hospital

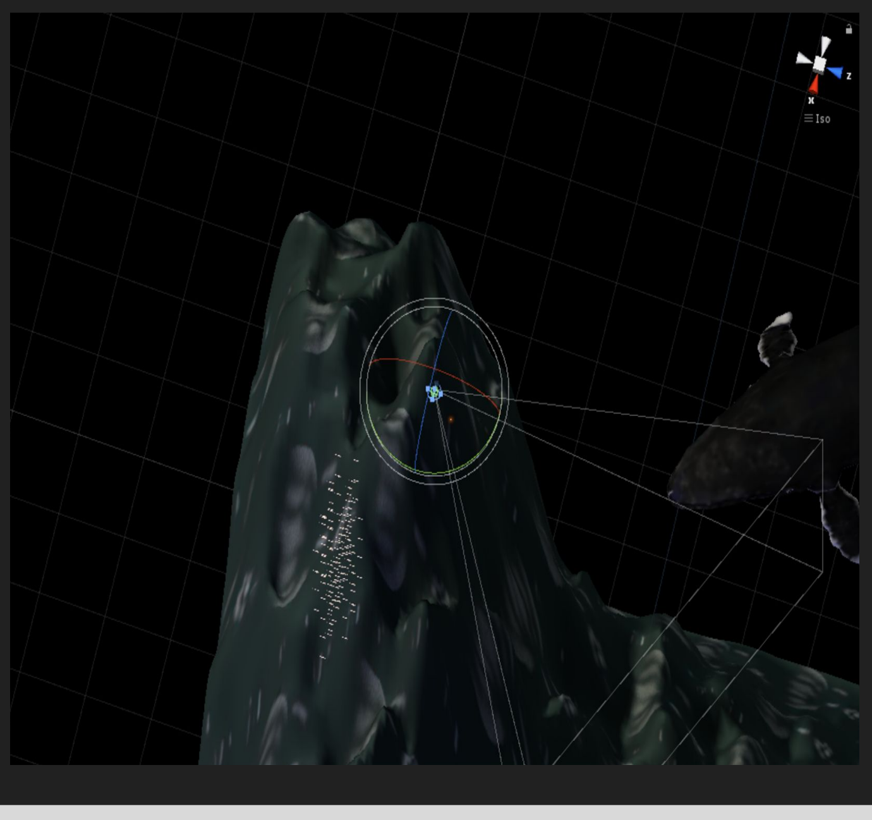


Hospital scene that involves exploring an abandoned hospital and being chased by half dissected skeletons (left) and other surgical monstrosities

Scene 4: Ice Sea



Underwater level that involves descending into the depths of the sea and encountering sea monsters



METHODS

To retrieve the heart rate data and use it in the Unity game, we created an Android wear application to sense heart rate data and send it to the paired phone, an Android service to receive the heart rate data and broadcast it to the Unity game, and finally a Java plugin to allow Unity to listen for this heart rate data in the game, as follows:

Sense Heart Rate > Send HR Data from Watch to Android Phone > Listen for Data on Phone > Send HR Data to Unity Plugin > **Use in Unity Game**

The game is also capable of tailoring the game experience based on the emotional response. A script called Fear Manager changes the frequency of objects in the scene based on the player's monitored responses. Responses considered include heartbeat, the speed of the player's head movement, and what objects the player's eyesight lingers on. Using these three responses, the script attempts to determine whether or not the player is scared of any given object, incrementing/decrementing an integer variable dedicated to that object. Once an object's variable passes a threshold of .5, the game determines that the player is scared of the object in question. After making this decision, it will then procedurally increase or decrease the frequency of this object in the game world, depending upon whether the game is set to "easy" or "hard" mode. The script attempts this decision making process with every object in game: spiders, fish, etc.

CONCLUSIONS

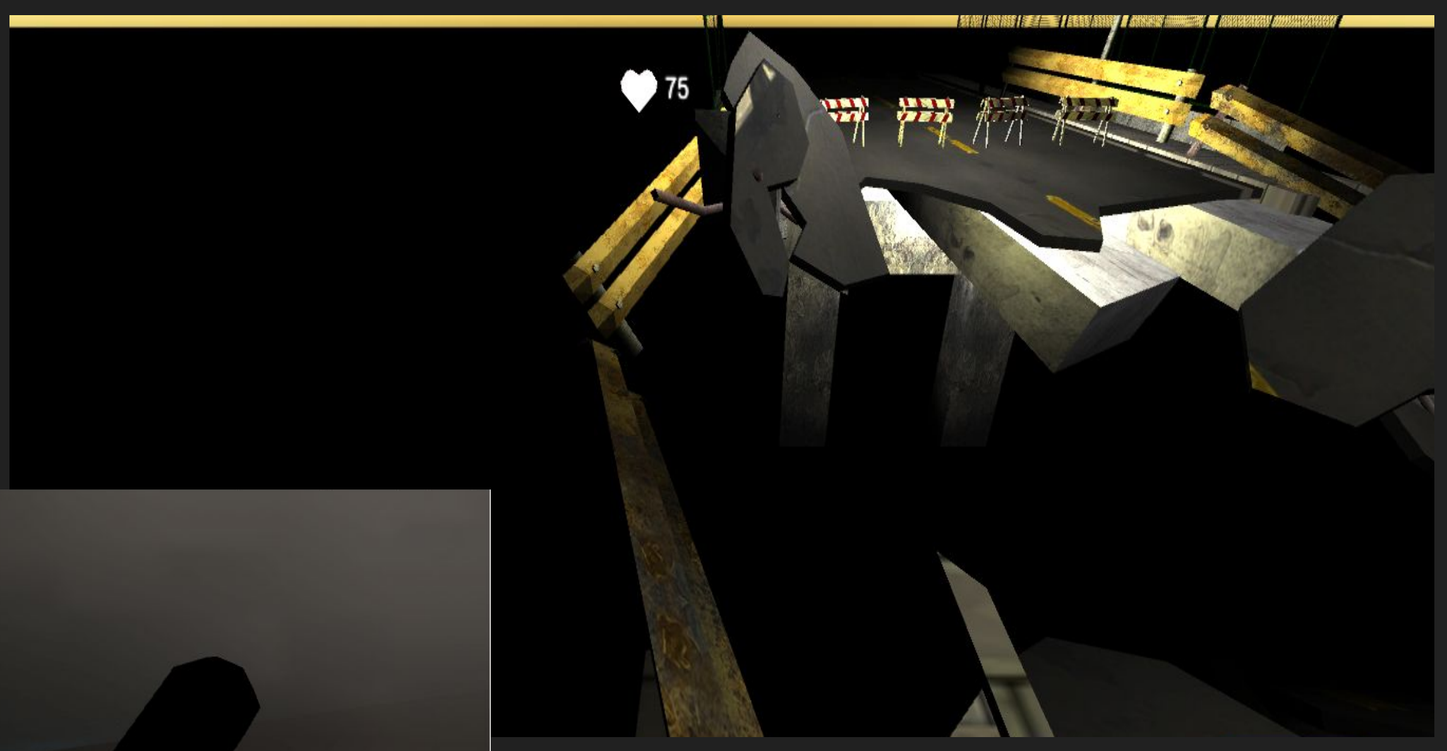
We successfully created a playable version of Heartbeat Horror with scripted player movement through four scenes, which can be seen in the pictures below. Our game included the necessary functionality and meets all three of our stated objectives. Our game demonstrates a working proof of the concept we are attempting to demonstrate.

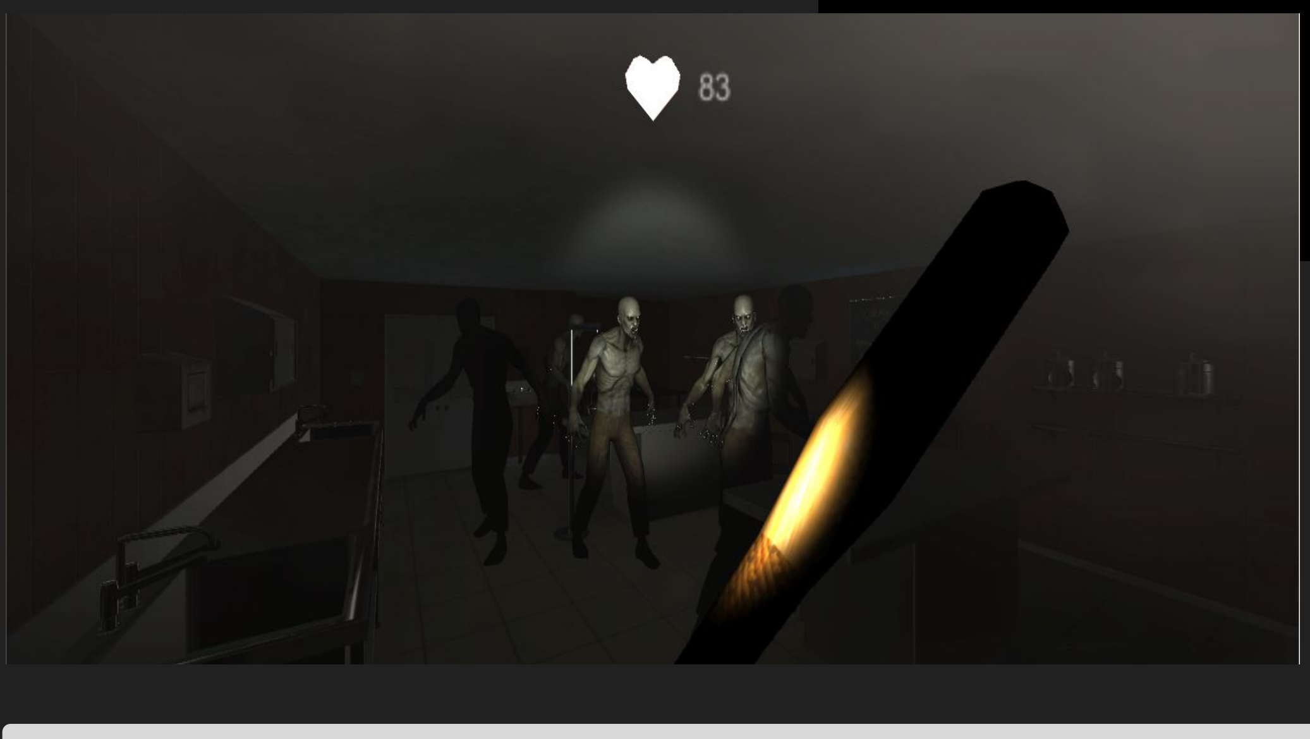
The next step in improving the game itself is to add player-controlled movement and ways for a player to interact with the game environment. Giving a player more utility within the game will allow them to become more immersed in it, making sensed data more genuine. Additional player features would also provide more sensing opportunities to more precisely gauge subconscious player responses to game aspects, and would allow for more complex feedback based on observations the game makes about a player.

We would also like to add the ability to monitor verbal player data, which would give us even more information about a player's response. The prototype scenes below are examples of the more complex scenes that we could implement with more complex player behavior, and additional features based on sensed player response that aren't possible in our current implementation.

Prototype Scenes

To the right, a player must cross a narrow ledge between two parts of a broken bridge. The higher the player's heart rate, the more the camera will shake, simulating a physical response to fear.





To the left, a player must pass through a room filled with zombies that can 'sense fear' and will only attack if the player's heart rate rises too high. The player must stay calm to avoid detection and reach safety.

ACKNOWLEDGMENTS

This project was completed as part of CS 234/334 Mobile Computing (Winter 2017), taught by Prof. Andrew A Chien with TA support by Gushu Li and Ryan Wu. We gratefully acknowledge the generous support of Samsung in providing GearVR equipment.

REFERENCES

America's Top Fears 2016 - Chapman University Survey of American Fears." *Wilkinson College of Arts, Humanities, and Social Sciences*. N.p., n.d. Web. 15 Mar. 2017

Android Native Plugin: From Android Studio to Unity - Page 5 of 13." *Add Component*. N.p., 19 Feb. 2017. Web. 15 Mar. 2017.

Mehers, Damian. "Using Android Wear to Control Google Cardboard Unity VR." Damian Mehers' Blog. N.p., 23 Aug. 2015. Web. 15 Mar. 2017.

Lugstein, Johannes. "Using Android Watch Sensors to Control a Unity Game." Haknode. N.p., 07 Feb. 2017. Web. 15 Mar. 2017.

Jackson, Patrick. "MessageApi: Simple Conversations with Android Wear." Sitewide ATOM. N.p., 18 Aug. 2014. Web. 15 Mar. 2017.

AddComponent. "Android Native Plugin: From Android Studio to Unity - Page 2 of 13." Add Component. N.p., 19 Feb. 2017. Web. 15 Mar. 2017.

Meyblum, Jean. "Communication between an Android App and Unity." Jean Meyblum - Game Programer. N.p., 17 Feb. 2014. Web. 15 Mar. 2017.