****

**University of Arkansas – CSCE Department**

**Capstone II – Project Design Document – Spring 2021**

# **Immersive Video Game to Encourage Proper Breathing and Alleviate Anxiety in Young Patients during Preoperative Anesthesia Delivery**

**William Baker, Gavin Glenn, Jackson Gregory, Jared Harris, John Ostermueller, John Shelnutt**

**Introduction**

The purpose of this document is to give a high-level overview of both the hardware and software components of the game. This document goes into detail about each of the components necessary to make this game function for its intended purpose of reducing preoperative anxiety in young patients. This document goes over each piece of hardware that is necessary to fulfill this purpose as well as how the game is coded in C#.

**Hardware Components**

### **GE Avance CS2**

This device is what monitors the patient’s breathing patterns while they are receiving anesthetic before an operation. This device is not connected to the game itself and no data from this machine is read by the game. This device is necessary for whoever is controlling the movement of the diver, as they will need to monitor this machine to determine when the diver should move based on the patient’s breathing patterns. This machine is not necessary to make the game run, but it is necessary to fulfill its intended purpose.

### **Amazon Fire Tablets**

This device is what the game runs on after loading it onto the device and was the primary way in which deployability was tested. The game allows anybody to tap the screen to create bubbles within the environment while the game is running. Due to the use of the Unity Engine, the game is able to be deployed onto any android device, meaning it is not limited to just this device.

### **DinoFire Type C/USB Presentation Clicker**

This device controls what movement mode the game is currently in and when the diver moves when the game is in manual mode. The clicker has two arrow keys that the game registers as the page up and page down keys. The page down key toggles between the manual and automatic movement modes that the game has. While the game is in automatic mode, the diver will automatically move forward after a set amount of time. While the game is in manual mode, the page up key will trigger the diver to move through the environment. The diver will not move in manual mode unless this key is pressed, and this key will do nothing while the game is in automatic mode. This device must be connected to the device that is running the game so that the diver’s movement within the environment can be controlled.

**Software Components**

### **Unity Engine**

This game was made utilizing the Unity Engine, which allowed us to make several scripts for different aspects of the game in C#. These scripts control the majority of the functionality within the game. The codebase for our game is made up of the scripts described below.

### **BackgroundScrolling.cs**

This script handles the placement of environmental elements as the camera moves with the diver. The script is connected to the main camera and starts by setting the camera boundaries. After setting the boundaries, the script creates additional copies of each specified environmental element until it has created enough duplicates for each element to move past the boundaries set by the camera. Once the camera begins moving, the script checks every frame if any of the specified elements are outside of the boundaries of the camera while it is moving to the right and if any are, they are then moved to be just outside of the boundaries for the right side of the camera so that appear in the camera’s view as it is moving.

### **BubbleMovement.cs**

This script handles the movement of all bubbles that can be created within the game. This script is an inherent property of all bubbles that are spawned and moves them upward with a set speed while they wiggle back and forth. Once a bubble is out of view of the camera, it is then removed from the game so that the game does have to keep track of bubbles that cannot be seen. This script performs this movement and the check to remove the bubble every frame.

### **CameraMovement.cs**

This script handles the movement of the diver and camera based on the current mode of movement. This script is connected to the main camera and works by running a coroutine whenever movement needs to happen. This coroutine moves both the camera and diver forward in the environment whenever it is called. In manual mode, this coroutine is called whenever the page up button is pressed and it is not already moving. In automatic mode, this coroutine is called once every 2.5 seconds. While movement is occurring, this script also calls upon the MotionSteps.cs script to smooth out the movement of the diver so that it is done as one continuous movement. This script determines what mode to be in from the InputController.cs script.

### **CoinBehavior.cs**

This script handles how coins act when they are floating in the environment and when the diver collides with them. This script is an inherent property of all coins and spins them in place while they are within the environment and removes coins from the game that move past the left boundary of the camera. This spin and check are done every frame. Whenever a diver collides with a coin, a coin is added to the inventory stored within the game, a sound plays to denote that a coin has been picked up, and it is then removed from the game.

### **InputController.cs**

This script handles the toggling between the automatic and manual modes of movement within the game as well as the spawning of bubbles. The script is associated with an invisible game object that spawns bubbles whenever the screen is tapped and tells the CameraMovement.cs script which movement mode to be in. This script also spawns in new coins after the diver has collected one. These checks for if the appropriate input for bubble spawning and movement mode toggling as well as when to spawn coins is done every frame.

### **Inventory.cs**

This script keeps track of the total amount of coins that have been collected by the diver, and displays that total as a counter in the upper right hand corner of the screen.

### **MotionSteps.cs**

This script smooths out movement of the diver. This script is not a property of any entity within the game and is not attached to anything. It is called by other scripts that determine when movement needs to occur.

### **Game Assets**

**Diver**

The diver asset is comprised of about 15 layers that make up the arms, legs, torso, etc. This asset was designed in Gimp and then skinned using Unity’s sprite editor. Bones were drawn on the sprite in order rig animations, and then animations were made for idle and swimming movements. Additional animations were made for transitioning between idle and swimming animations.

**Fish**

Similar to the diver, the fish models are comprised of several layers designed in Gimp and then skinned in Unity’s sprite editor. The fish have two animations, one for individual fin movement and another for group movement (several fish are placed next to each other in the underwater environment).

**Coin**

A coin sprite was used to add another layer of engagement to the game. Coins are spawned on the right side of the screen; the diver collects these coins as it moves past them and increments a counter in the top right of the screen.

**Bubble**

A bubble sprite was added that is spawned whenever the player taps the screen. This was done to add another level of engagement, namely touch-capability.

**Background**

The background was composed of a gradient of shades of blue starting with the lightest at the top going down to the darkest at the bottom and then a gaussian blur was applied to remove hard lines between the layers. This would emulate a sense of depth within the water without any apparent light rays passing through. Afterwards a layer of sand was added to the bottom of the water to emulate the ocean floor, and logically add flora to the bottom of the ocean.

**Flora/Rocks/Coral**

The ideas for other assets to be added to the ocean floor were mainly coral, seaweed like plants, rocks, and some things like anchors and minor flora. These would be the most identifiable to a child and were made using Gimp. An outer layer was sketched to represent the outer shape of the object to be created, and then generally bucket filled with the most appropriate color and shaded according to a light source being directly above the object. The shading would allow these objects to seem much less dull and add variety to the environment without seeming too realistic for the scene.