Boston University
Electrical & Computer Engineering
EC464 Capstone Senior Design Project

Second Prototype Testing Plan

Augmented Reality Climbing Wall

by

Team 14
Augmented Reality Climbing Wall

Team Members

Tom Panenko tompan@bu.edu
Taylor Hartman hartmant@bu.edu
Michael Igwe migwe@bu.edu
Ryan Smith rsmith66@bu.edu

Required Materials

- Circuitry
    - Full Bridge Rectifier (x2)
        - Diode (x4)
    - Comparator
        - MCP6241 Op Amp
        - Resistors (x4)
    - Multiplexor
    - Coils (x8)
    - Push Buttons (x4)
    - Function Generator
    - Power Supply
    - Jumper Wires
- Hardware
    - Enclosure (x2)
    - Button (x2)
    - Push Button Holder (x2)
    - Raspberry Pi 4 Model B
- Software
    - Laptop
    - Unity game engine
    - Game .cs scripts

Set Up

One of the features we plan on testing for this prototype testing is the game that will be played on the climbing wall. The game is played by the user holding onto the wall and dodging missiles that slowly fall down overhead. Each time a user is hit, he or she loses a life, and the game ends once a user has survived for 45 seconds or has lost all three lives. This game was coded in C# on the Unity Game Engine. The game is calibrated to work with the climbing wall by first taking in a calibration.txt file that contains a matrix used to set the location for each of the holds and takes in a data.txt file that notifies the game which holds are currently being pressed by the user using a matrix filled with 0s and 1s. Lines are drawn between the activated holds to simulate the body of the user, and the data.txt file is repeatedly checked to see if the activated holds have changed, and if so, the old holds are deactivated and the new ones are activated.

Pre-Testing Setup Procedure

1. Assemble the hold hardware by bolting the enclosure onto the test board with necessary buttons and electronics connected in between
2. Test with function generator and multimeter that wireless transfer with the hold is working correctly
3. Connect output of backside wireless power coils to full bridge rectifier and measure DC voltage
4. Connect output of full bridge rectifier to multiplexer, ground all non used inputs, and test functionality via multimeter
5. Connect output of multiplexer to comparator and test via Raspberry Pi

6. On the laptop, start Unity Engine and open the project for the climbing wall game

Testing Procedure

Multiplexing

1. Connect both holds to two different pins of a single multiplexer, which is connected to a GPIO pin on the Raspberry Pi.
2. Perform all four combinations of inputs possible: neither hold pressed, 1st hold pressed, 2nd hold pressed, both holds pressed. Observe the results on the output console and ensure that the correct bits are set.
3. Connect both holds to two different multiplexers (and two separate GPIO pins) but to the same index on the multiplexer.
4. Perform all four combinations of inputs possible: neither hold pressed, 1st hold pressed, 2nd hold pressed, both holds pressed. Observe the results on the output console and ensure that the correct bits are set.

Game

1. In the Unity game engine, start the game
2. Let the game play and watch missiles fall and collide with activated holds/lines
3. Show the test calibration file and how that matches up to the location of the holds
4. Show the test data file and how that is highlighting the activated holds
5. Change the location of activated holds in the data file and watch holds and lines change
6. Let game play till the end and observe Game Over screen with score
7. Click the play again button and watch the game start over

Measurable Criteria

The criteria for successfully testing with the correct result is as follows:

I. During the testing procedure, all of the hold data bits are unset while all holds are unpressed.
II. When a hold is depressed, only its corresponding hold data bit is set within 100ms, and remains set while the hold is held.
III. When a hold is released, only its corresponding hold data bit is unset within 100ms, and remains unset until the hold is pressed again.

Score Sheet

Action	Correct? (1/0)
Hold data console displays all 0s when no holds pressed	
Hold data console correctly displays 1 in the row/column that the depressed hold is assigned to.	
Hold data console correctly displays 1s for simultaneously depressed holds attached to	

the same multiplexer	
Hold data console correctly displays 1s for simultaneously depressed holds attached to the same multiplexer index but different GPIO ports.	
Game plays without error and within expectations for the gameplay	
Game successfully calibrates hold position and activates the depressed holds	
Game reads change in depressed holds and reacts successfully	
Total	/7

Appendix