Interface Specification: VR Tightrope Simulator

Summary: Our Tightrope Simulator consists of two separate hardware devices that require constant interaction to determine a positive user experience. As such, the Tightrope Simulator can effectively be divided into two equal components: the Arduino and the Oculus Rift. The specifications for how these devices interact are documented below.

Arduino:

GetLeftPressure(), GetRightPressure(): These function intend to expose the pressure sensors of the left and right foot. A call to one of them would return an array of pressure data from each of the individual sensors on that foot. This data scales from 0 - 1000, where 0 is the lowest pressure the sensor can handle and 1000 is the highest pressure it could handle. A JSONified version of this data might look like the following:

```
[
        {
                “sensorSet”:    ”left”,
                “sensor1”:      “1000”,
                “sensor2”:      “400”,
                “sensor3”:      “264”,
                “sensor4”:      “800”
        }
]
```

The field sensorSet would either be left or right. This indicates which foot it is returned from. Sensors 1 - 4 are the different sensors placed through the sole of the shoe.

GetArduinoStatus(): This function would run a check on the Arduino to ensure it is properly communicating data.

GetSensorStatus(): This function would run a check on each of the individual sensors on both feet to ensure that they can return properly calibrated data and are functioning as expected.

CalibrateArduino(): This function runs a procedure on the Arduino for calibrating each individual sensor. This data would be communicated through the Oculus headset to the user as a mini-tutorial where they would put pressure on each individual sensor.

Oculus, Unity:

GetArduinoData(): The simulator would be able to constantly receive real time data from the pressure sensors of the Arduino. Unity will able to retrieve the data and communicate with the

Arduino using the SerialPort class. This way those pressure values can be processed into advice for the user.

ProcessArduinoData(): This function would take in the data retrieved to Unity from the Arduino pressure sensors and respond to the simulation's user interface with appropriate feedback messages. This function will be the main backbone of the feedback loop of the simulator.

PressureChangeCheck(): This function checks to see if there are any big discrepancies between data sets in order to respond quickly. This way the simulator will understand that the user is veering off course and respond appropriately. This would allow for less repetitive and common feedback responses when considering only slight changes in pressure data sets.

SetUIData(): This function takes a set of PNG files and coordinates, and renders a graphical user interface using this data to the Oculus headset.

Basic Setup Diagram: