Harrison Hua

Student Focus: Backend, and destination configuration

Project: Researcher’s backend support

# Overview

The server will assist the researchers by helping reduce the time taken to set up each test. Currently, there is a delay between tests due to the extensive work the research team must do to correctly configure the different devices together. A simple change to a parameter or device may delay the researchers quite a bit. To address this issue, our group has proposed the use of the use of a socket-based communication server. Various parts of this project must be built to provide meaningful improvement for the researchers in the testing phase. These enhancements range from managing and displaying real-time data to handling the connections between different clients, ease of use, and reliability.

One of the most critical aspects of this server is the ability to handle multiple clients and denote the destination for each client. With the vibrating belt rig, the research team must connect many unique sensors requiring different input and output devices. To provide a meaningful enhancement to the research team’s effort, this system will be able to configure the connections without much effort and adjustment by the research team. This design will provide a basic code sample for the research team to add to the code of existing and new devices. This code sample will handle the connection between client and server, receive and handle data, and send data. This means that the server will primarily handle the individual destinations for the various clients. This design will aid the research team as less effort will be placed on establishing a start-end connection, as they will only have to configure the code to communicate with the server properly.

# Implementation

## Technologies

Python

A sample server and client were created with Python in the preliminary implementation. The choice to use Python for the server was due to the research team and design group’s familiarity with Python. Python's additional benefit is the built-in libraries for methods on sockets and threads. This is convenient for the research team for two reasons: Firstly, no external downloads are required to set up the server. Moreover, there will not be any missing dependency issues in the chance that the research team switches devices. The goal is to maintain this principle in the final design. Lastly, due to the common nature of these built-in libraries, there is more documentation and help throughout the internet. This may make it easier for the researchers to pick up, as more resources will help guide them through the code.

C#

A sample client was created with C# in the preliminary implementation. Besides Python, there were two other programming languages that the professor indicated were needed for inter-process communication: C++ and C#. The research team has indicated that they have seen C++ in the software of some of their devices and the Unity game engine is written in C#. Though Python could and has been made for both the receiver and the listener, it was important to not only understand how a different language handles socket communication, but test the server implementation with two different clients communicating on the same socket.

LINQPad

It is not common to run a singular C# file without a project or solution. In order to test a simple C# script without compiling an entire solution, LINQPad was used to compile and run a the C# script.

Sockets

Threads

## Preliminary Implementation

## Future Work

C++

Errors Handling

Setting up a test between the server and client

Setting up a test in order to test multiple connections

# Potential Effects References