6DoF Tracking Device: Glove I/O

Berkeley | EECS
Electrical Engineering and Computer Sciences
EECS 149/249A - 2018

Ahmed Malk, Luis Sanchez, Chase Muramoto, Sachin DeYoung, with support from Bala Kumaravel

https://github.com/azer0m/EECS149_Project

Project Overview

The goal of this project was to design a lightweight, hand-worn 6 Degrees of Freedom (DoF) I/O device that could be tracked in 3D space, and enable interaction with a virtual environment, the impetus for this being a glove for Augmented Reality purposes.

We achieve this by attaching an orientation (IMU) sensor to measure the rotation of the hand, and employ flex sensors to enable usage of 5 fingers. The hand is tracked in 3D space via depth sensing using a Microsoft Kinect v2.

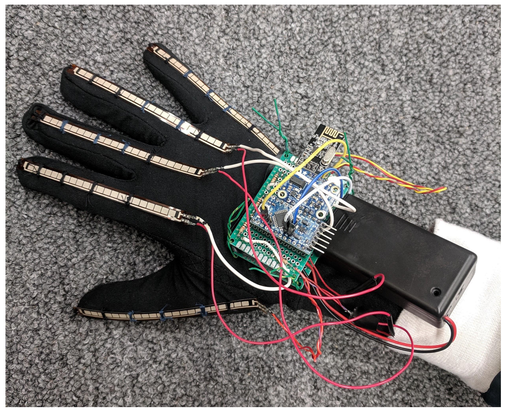
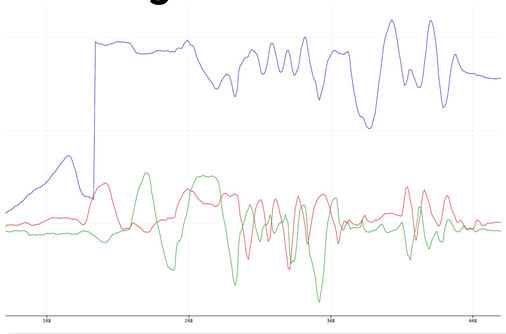
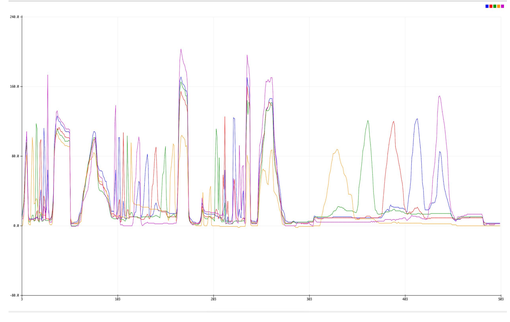


Image of glove tracking device. The protoboard holds the Arduino Pro Mini, IMU, 2.4GHz radio, and a 9V battery pack is velcroed onto the wrist. The flex sensors and protoboard are sewed onto the glove.

Testing

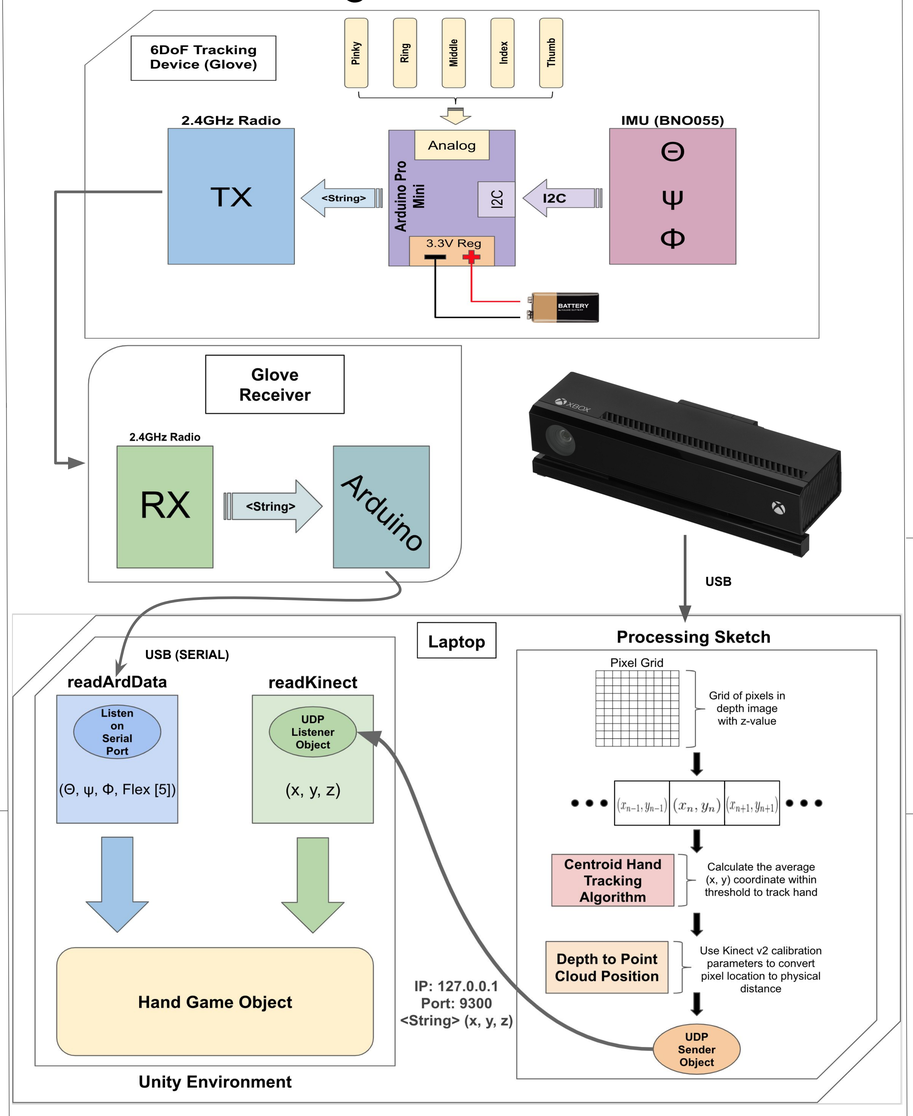


Screen capture of orientation data taken from IMU on glove while in use (blue: yaw, red: pitch, green: roll).



Screen capture of flex sensors' usage while glove in use. The 5 peaks at the end are individual fingers bent and extended.

Architecture Diagram



UDP Implementation

```
Ahmed — Listener.dll — dotnet
Waiting for broadcast
Received broadcast from 127.0.0.1:61974 :
 0
Waiting for broadcast
Received broadcast from 127.0.0.1:61974 :
 1
Waiting for broadcast
Received broadcast from 127.0.0.1:61974 :
 2
```

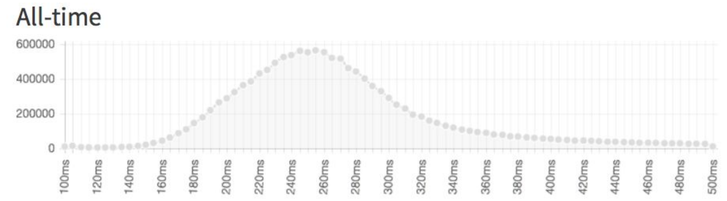
Originally, we thought to write our position data to a CSV and have Unity constantly read it. However, this proved too difficult to do without introducing latency, and releasing locks on a file during read/write processes. Instead, we opted to send our positional three-tuple of (x, y, z) as a string over UDP, on port 9300 over localhost 127.0.0.1.

```
-- UDP session started at Sun Dec 09 21:10:56 PST 2018 --
-- bound socket to host:null, port: 61974 --
[18-12-09 21:10:56.837 -0800] send packet -> address:/127.0.0.1, port:9300, length: 1
Sent number: 0
[18-12-09 21:10:59.357 -0800] send packet -> address:/127.0.0.1, port:9300, length: 1
Sent number: 1
[18-12-09 21:11:01.864 -0800] send packet -> address:/127.0.0.1, port:9300, length: 1
Sent number: 2
```

Here, our sender broadcasts ints & our client listens for messages.

Latency Matters

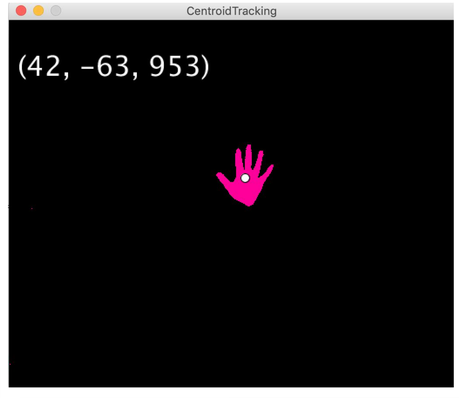
As our glove serves as a controller for a virtual environment, it's important that actions in the physical world are represented similarly in the virtual world, with similar speed. As such, there only exists minimal delay in our transmission of inputs to outputs in Unity. Any inherent delays are far quicker than the average human's reaction time (Source: IBMathsResources).



Unity Updates Too Quickly

Unity's update() function runs too quickly, causing incoming tracking data to be missed. To solve this issue, we effectively "slow down" Unity's clock by using a counter to only allow the function to perform actions every 10th iteration.

Centroid Hand Tracking Algorithm



Tracks the hand by taking the average of all visible (non-black) pixel coordinates within a min and max threshold, set to the extension range of user's arm.

Course Topics

- **Sensors & Actuators** (flex, IMU, Kinect IR depth)
- **Input & Output** (Input data controls Unity object; timer implementation for Unity)
- **Networking** (2.4GHz RF; UDP Sender & Listener)