Touch TrackIR

Team A6: Matthew Shen, Matthew Kuczynski, Darwin Torres Romero
18-500 Capstone Design, Spring 2022
Electrical and Computer Engineering Department
Carnegie Mellon University

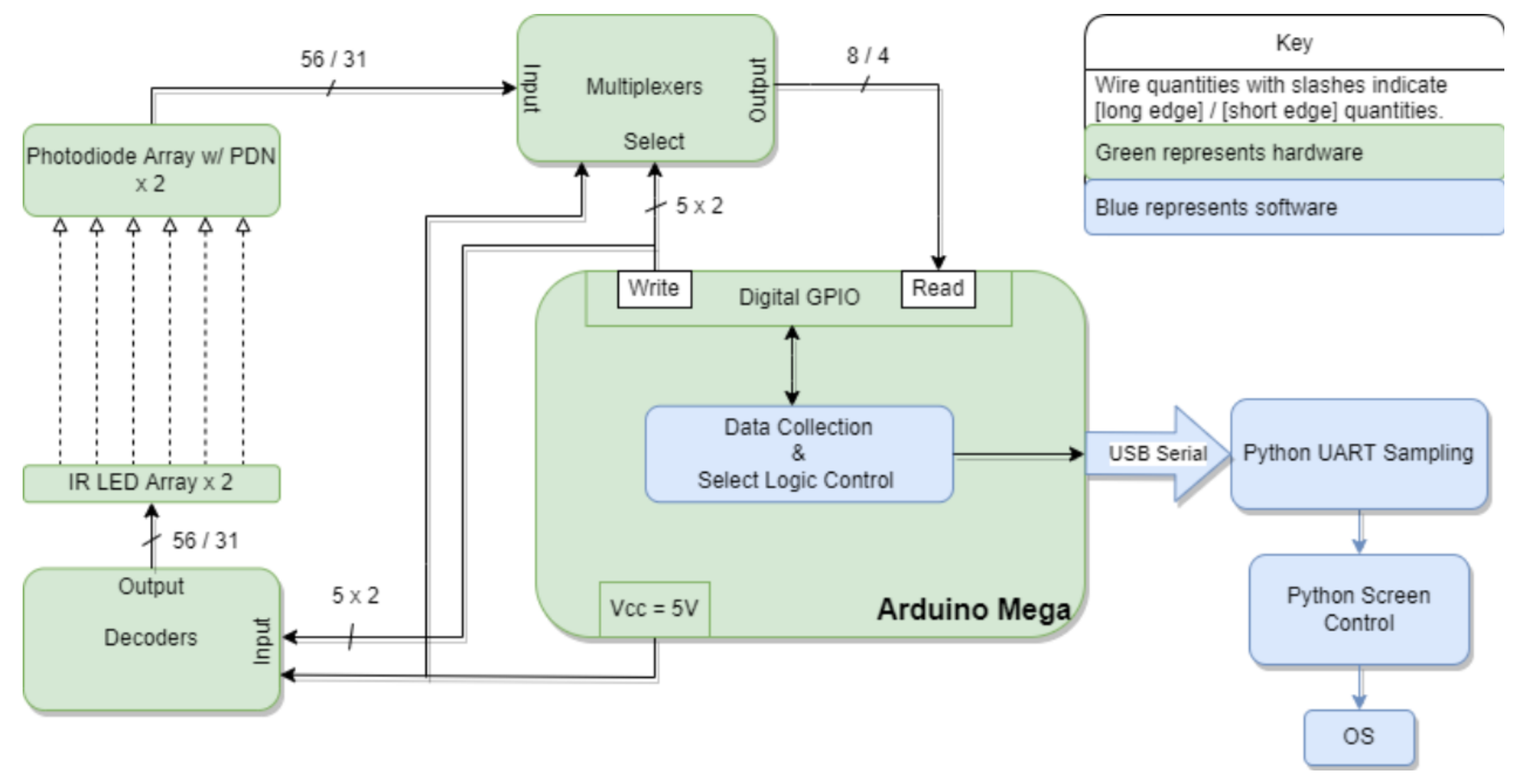
Touch TrackIR

Product Pitch

Most modern devices are touchscreen compatible, however laptops are one area of devices that usually are not. Our product allows consumers to make their non-touch compatible laptops work as touchscreen devices by simply placing a frame onto their laptop's screen. The product's design focuses on ensuring high touch precision, a low false positive/negative rate, a short response time, a high refresh rate, and that the device is lightweight. Currently, we can achieve a touch precision around an ⅛ inch, false positive/negative rates below 1%, and a refresh rate of over 50 Hz. Additionally, the product is only about a ½ lb, ensuring the screen stays stationary.

System Architecture

Our system uses 56 LED-photodiode pairs on the bottom/top and 31 pairs on the left/right to detect finger position. Arduino code controls the input/select values for decoders and multiplexers, which control the LEDs that are turned on and the photodiodes being read from. The photodiode values are sent over serial to Python code, which determines the finger position and sends an appropriate command to our screen control subsystem.



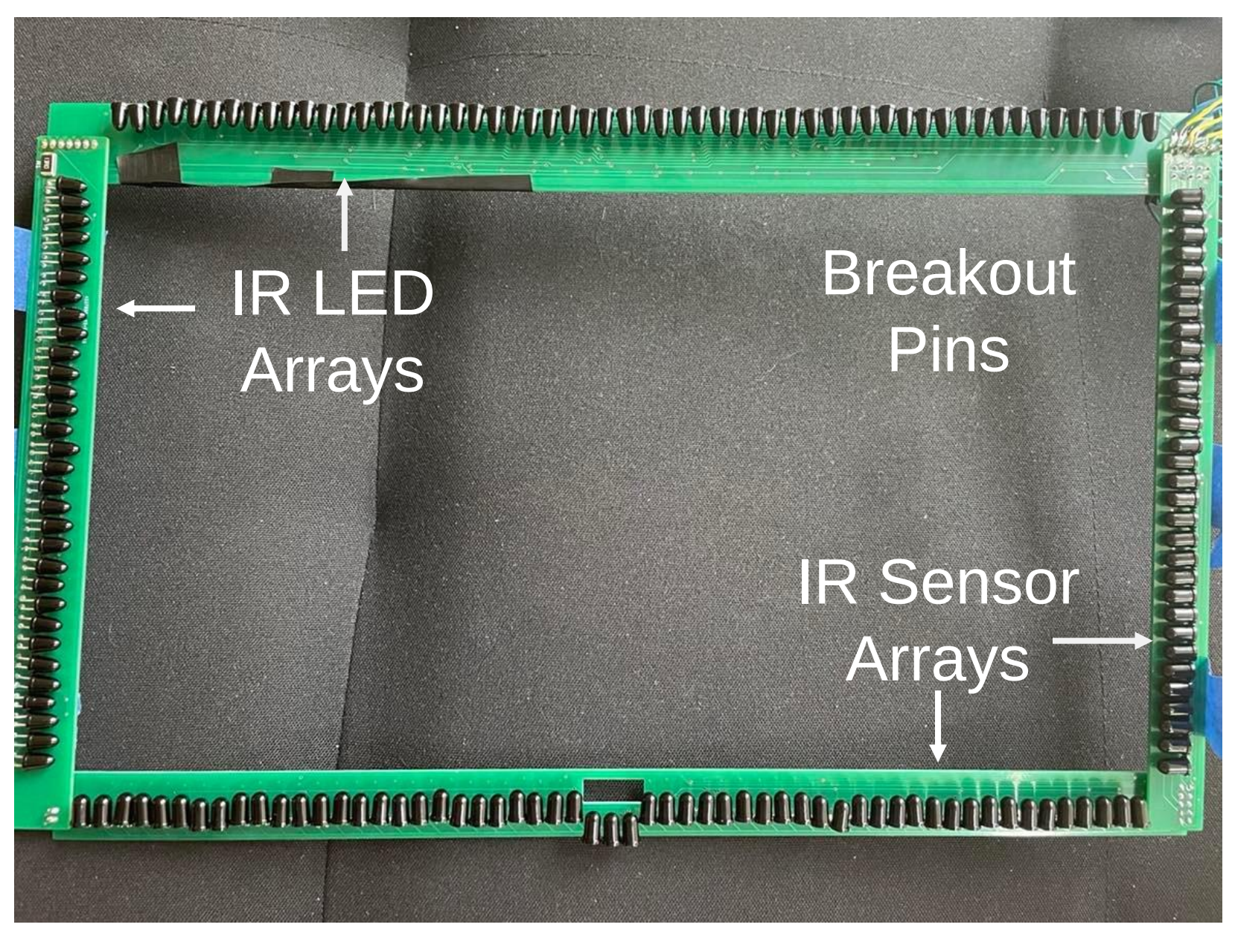
Conclusions & Additional Information

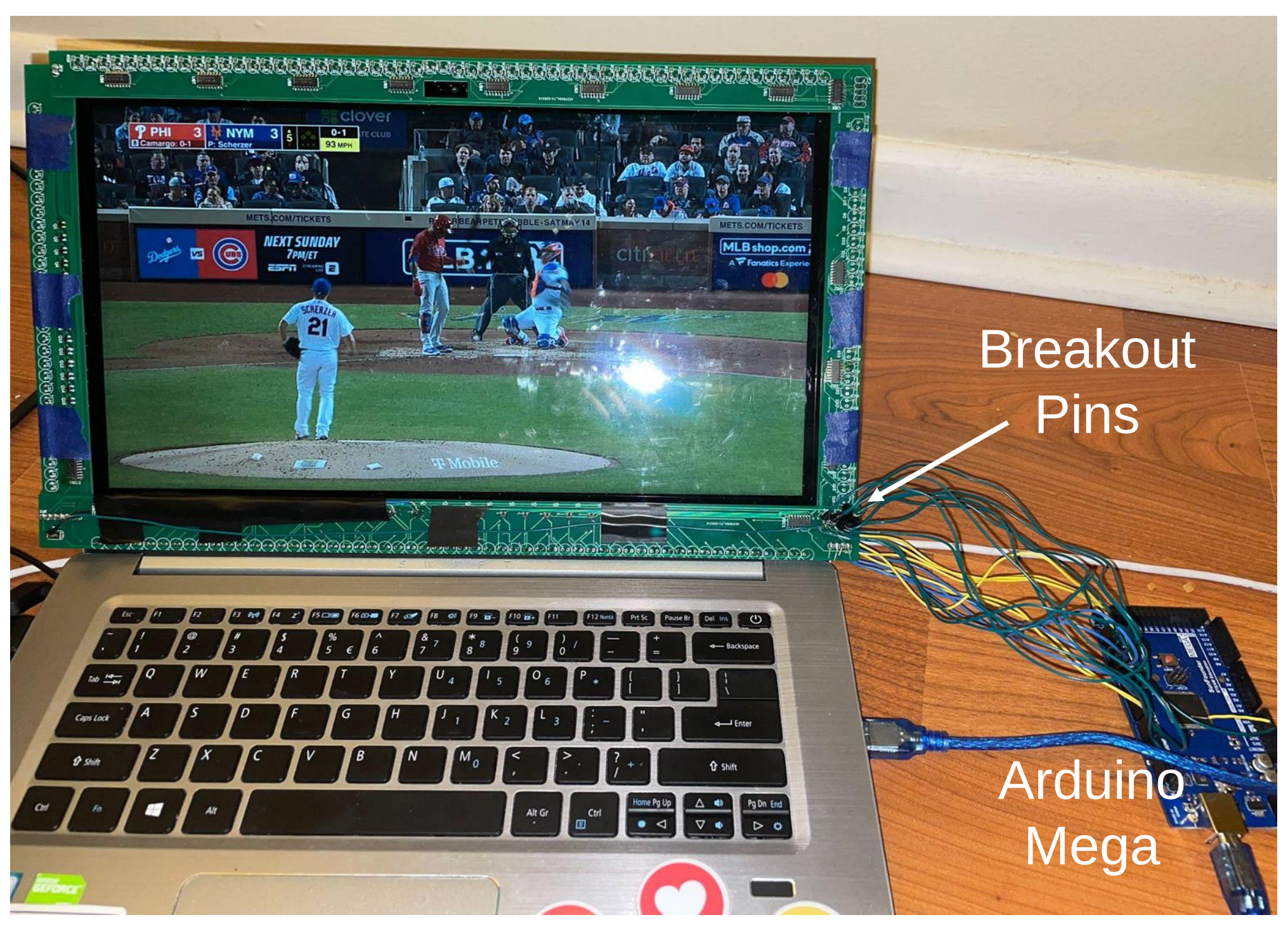
Our system ended up working as well as we wished for at the beginning of our process. We achieved our goals for accuracy, and even achieved some of our reach goals such as implementing drag gestures. This project taught us a lot about being patient with our design, and not to rush past critical unit testing early on. Due to our time constraints and budget, we treated the success of the PCB as a make-or-break. Thus, to increase the likelihood of our project's success, we ran several tests on a scaled-down version of our system to avoid any surprises when the PCB arrived.

If we had more time for our project, we would try to make accuracy improvements and create a casing for our frame that could improve the way our system attaches to screen.

System Description

Our system features a frame to be placed around a laptop screen. It uses infrared light to detect where on the screen a finger is touching to create a touchscreen experience. Our frame has light sensors on it, which send data to the computer to be translated into on-screen gestures.





System Evaluation

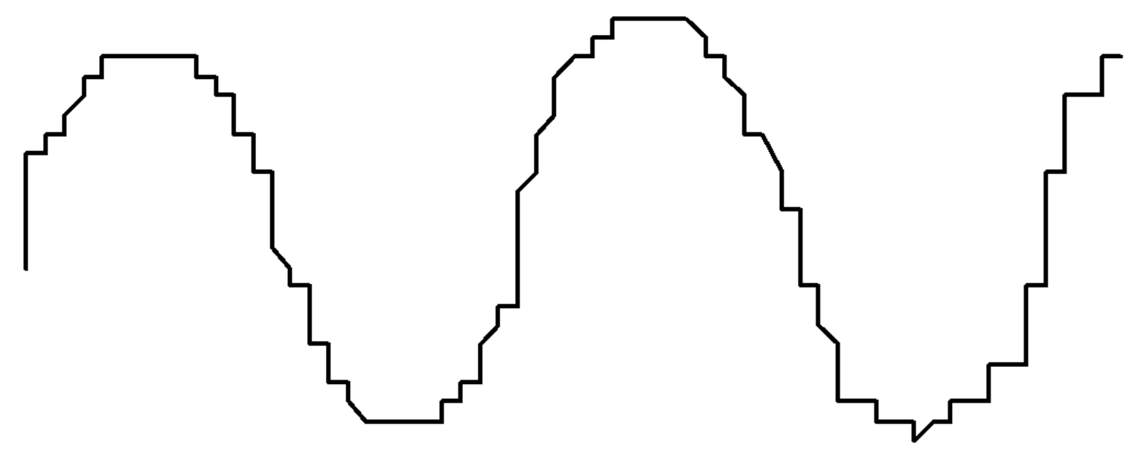
Requirement	Testing Method	Quantitative Goal	Measured Results
Touch Precision	Measure distance between center of finger and the actual click.	< .3 in	0.13 in
False Positive Rate	Let frame remain attached with screen untouched for one hour with script running, and count touches detected.	< 1 per 5 mins	0%
False Negative Rate	Repeatedly touch screen 100 times, and count how many requests were sent.	< 5%	0%
Response Time	Calculate time using frame-by-frame video analysis of touch and screen.	< 150 ms	120 ms
Refresh Rate	Analyze slow-mo video to determine cursor update time (temporarily disabling smoothing algorithm).	> 15 Hz	66 Hz
Weight	Measure on scale, and ensure screen stays open with frame attached.	< ½ lb	0.4 lbs

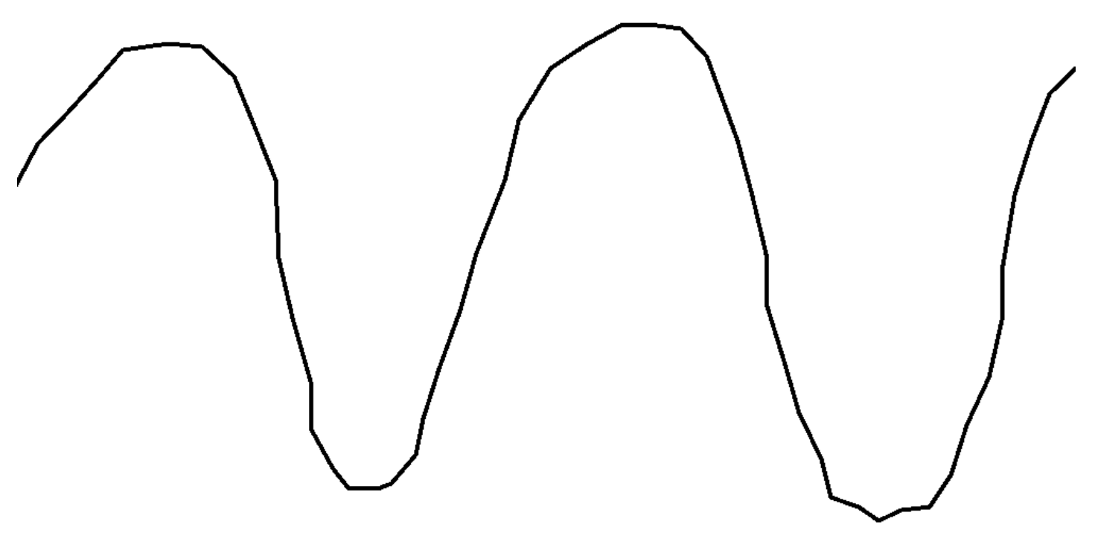
Tradeoffs that we considered in our design include:

- On the OS side, we chose to program in Python because despite its slower speed, it provides an intuitive library for serial communication. In the end, it was still more than fast enough for our use-case requirements.
- Smaller messages sent over serial allowed for faster communication at the cost of requiring more processing on the Python side.
- A faster refresh rate made our system smoother but could make it hard to differentiate between taps and small drags.
- To reduce the number of power FETs in our design, we illuminate more than one LED per NMOS switch. This required us to ensure that an LED's respective photodiode was tuned to be insensitive to a parallel LED's beam in order to prevent any interference.

We found that the MVP decided by the above requirements was usable but provided a suboptimal user experience due to tap/drag confusion and poor interpolation of touch coordinates detected between photodiode locations. Much of the work that we have done in addition to the MVP has been to improve the user experience by making drags smoother and single taps more recognizable.

The following curves were drawn in MS Paint using Touch TrackIR. The first curve is from our MVP. For the second curve, we interpolated the coordinates along a finger drag by averaging the detected positions in every 5 frames of updates.





Check out our blog site for Touch TrackIR!

http://course.ece.cmu.edu/~ece500/projects/s22-teama6/

Electrical & Computer ENGINEERING
Carnegie Mellon