Touch TrackIR

Team A6 - Matthew Kuczynski, Matthew Shen, Darwin Torres

Use Case

- Our product is intended to be used for non-touch-compatible laptops of a specific size to make them touch compatible. Activities such as drawing and website navigation are our specific target applications.

- We are trying to create a product for people who would appreciate a 2-in-1 laptop experience, but without the hassle of having to buy a new computer altogether.

- In a perfect world, we would like to expand our design to features such as 2-finger scroll and zoom, add programmable 3-finger gestures, and to have our product be functional in IR-noisy environments.

- Areas covered: Embedded & System Software, Analog & Power Circuits, Signal Processing

Use-Case Requirements

Key features and Metrics:

- Touch response time: This is crucial to any touch-operated device since it is frustrating to have to move at the speed of the device as opposed to moving at the speed of the user’s natural workflow.
- Precision: In order for a user to interact with objects on the screen, their finger position must be mapped precisely. Additionally, misclicks could have major consequences depending on the user’s application.

Use-Case Requirements

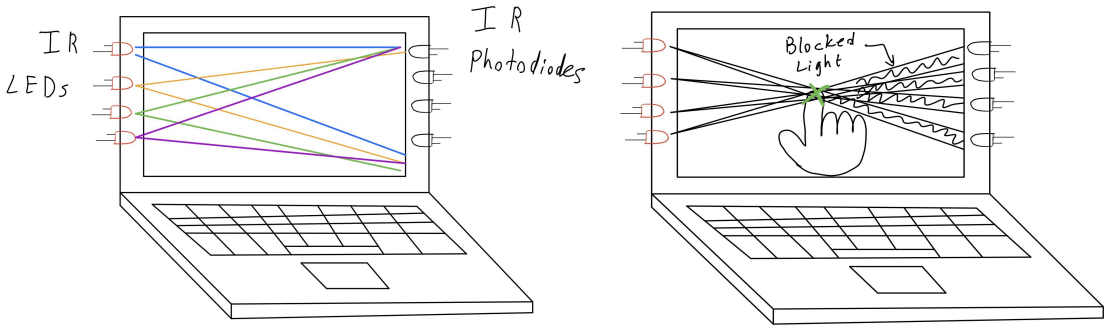
- Touch Precision: < 0.3 inch
  - The average adult index finger width is 0.6-0.8 inches, so we want to have precision no more than half the minimum
- Response Time: < 150 ms
  - MVP: 500 ms
  - Our goal of 150 ms touch latency is based on a low-performing, albeit flagship, Galaxy Tab 3.8.0 which released in 2013. A latency test reported a minimum response time of 168 ms.
  - For our MVP, we felt 500 ms would be the minimum bearable latency before no longer being deemed worth it.

Technical Challenges

- Technical challenges:
  - To meet our touch response goal of 150 ms, the speed of our code is of utmost importance, as all of the circuitry should be able to process data within microseconds. Ideally, we will run simulations starting with Python and transition to a lower-level language if Python proves too slow.
  - For our precision goal, the biggest challenge will be translating the analog voltages at each of our GPIOs to actual x-y coordinates on the screen. The second most challenging step would be determining optimal resistor values for the sensitivities of our sensors.
- Risk Mitigation
  - We plan to use inexpensive components at first to give ourselves flexibility to add new components to our design as we progress.
  - We will simulate our design entirely in software as an extra measure to ensure the math behind our design theoretically plays out as expected.

Solution Approach - Use-Case Requirements

- Touch Precision
  - We plan to use light triangulation to determine the 2D location of the finger on the screen.
  - Hardware Requirements:
    - IR LEDs
    - Photodiodes
    - Power Source – Battery



- Response Time
  - As explained before, the response time will be dependent on the quality of our software interface and implementation.
  - First, we will route the photodiode readings to digital GPIO pins on an Arduino Mega (or similar).
  - Then, pass data via serial USB to Python backend

Solution Approach - Design Requirements

- Stationary Frame
  - Wooden frame holding hardware on left and right sides of screen, connected across the width
- Coverage of Full Screen
  - LEDs placed some short distance from the edge of the screen in order to avoid dead areas
- Light Filtering
  - Modulation with duty cycle around 50%
- Software Processing
  - Implementation of the light triangulation method
  - Python OS Library and/or *win32api* to execute click

Testing and Verification - Use-Case Requirements

- Response Time
  - We will record a video of the product in use and then use a frame-by-frame analysis to determine the exact response time.
    - First Milestone: 1 second
- Position Accuracy
  - Similar to above, we can analyze a video to determine exact finger location and compare to the location determined by software.
    - First Milestone: ¾ inch
- Overall User Experience
  - User-friendly features
  - User feedback

Testing and Verification - Design Requirements

- Stationary Frame
  - Carrying around laptop and shaking does not cause LEDs to misalign or fall out.
- Coverage of Full Screen
  - Test edge cases of screen, including corners and spots equidistant from LEDs on left side of screen.
- Light Filtering
  - Place in indoor and outdoor environments, and in different light settings for both.
- Software Processing
  - Use simulations to test triangulation calculations and to experiment with different configurations of LEDs and photodiodes.

Tasks and Division of Labor

Matthew Shen's Tasks:

- Electronics research and ordering
- Initial Circuit design
- Breadboarding & Testing using only multimeter
- Test circuit design to test analog reads
- Develop Python codebase for analog reads from an Arduino Mega
- Hardware readjustments in later stages

Matthew Kuczynski's Tasks:

- Python simulation of hardware
- R&D of Python Windows screen control
- Develop Python codebase for analog reads from an Arduino Mega
- Develop Python codebase for Windows screen control
- *Reach goal:* add multi-finger compatibility into software

Tasks and Division of Labor

Darwin's Tasks:

- Python simulation of hardware
- R&D for Python/Arduino interfacing
- Develop Python codebase for Windows screen control
- Write script to test screen control
- *Reach goal:* add multi-finger compatibility into software

Fully-collaborative Tasks:

- Frame Design and Construction
- Integration of Hardware/Arduino/Windows Screen control
- Development of code for extra ADC if necessary
- Overall product testing
- *Reach goal:* Programmable 3-finger gestures

Schedule

	Week 1, Feb 7	Week 2, Feb 14	Week 3, Feb 21	Week 4, Feb 28	Week 5, Mar 7	Week 6, Mar 14	Week 7, Mar 21	Week 8, Mar 28	Week 9, Apr 4	Week 10, Apr 11	Week 11, Apr 18	Week 12, Apr 25
IR LED Python Simulation	Darwin											
	Matt K											
Research & order electronics	Matt S											
Circuit design												
Breadboarding & testing w/ multimeter												
Research & test Python screen control												
Research & test Py/Arduino interfacing												
Begin .py foundation to read analog values from Arduino												
Begin .py foundation for screen control												
Write simple script to test screen control												
Design simple circuit to test analog reads												
Frame Design Sketches				ALL								
Breadboard/Arduino Integration												
Breadboard/Arduino/Windows Integration												
Revaluation of accuracy, assess need for extra resolution												
Order 8-channel ADC if deemed appropriate												
If applicable, develop code for new ADC												
Adjust hardware for multi-finger functionality												
Frame Construction w/o electrical components												
Overall product testing and optimization												
Finalize frame construction w/ electrical components												
Reach goals:												
Programmable 3-finger gestures												
Begin adding multi-finger functionality into software												