TAPS: Touch-based Adaptive Predictive Screen

TEAM MEMBERS:

IRENE AGBECHA

DAVID BARRY

JONATHAN SIMMONS

Problem Statement

- Many touchscreen products are unable to adapt adequately to dynamic environmental changes, such as changes in light intensity and the introduction of vibration.
- These environmental factors introduce errors such as “false touches,” which result in the trigger of unintended screen functionality.
- These touchscreen products may be found in hospital equipment, aircraft, police vehicles, etc.

Project Goals

Main Goal

- The goal of this project is to develop enhancements for a touchscreen product. These enhancements enable the touchscreen to adapt dynamically to changes in light intensity and the introduction of vibration to the touchscreen system

Sub-goals

- To compensate for the presence of vibration
- To compensate for changes in light intensity
- To incorporate configurable compensation switch
- To design and conduct experiments to test system functionality

Design Needs

Needs	Importance
Compensates for Vibration	5
Compensates for changes in Light intensity	5
Has Compensation Switch	5
Is able to Collect Experimental Data	5

Design Specifications

Metrics	Units	Unit Alias	Marginally Acceptable	Ideal
Percentage of false touches	Number of false touches per 1,000 touches	FTT	20% decrease with compensation	80% decrease with compensation
Discernibility of On-screen Characters	Number of discernible characters per page	CPP	50% increase in CPP with compensation	90% increase with compensation

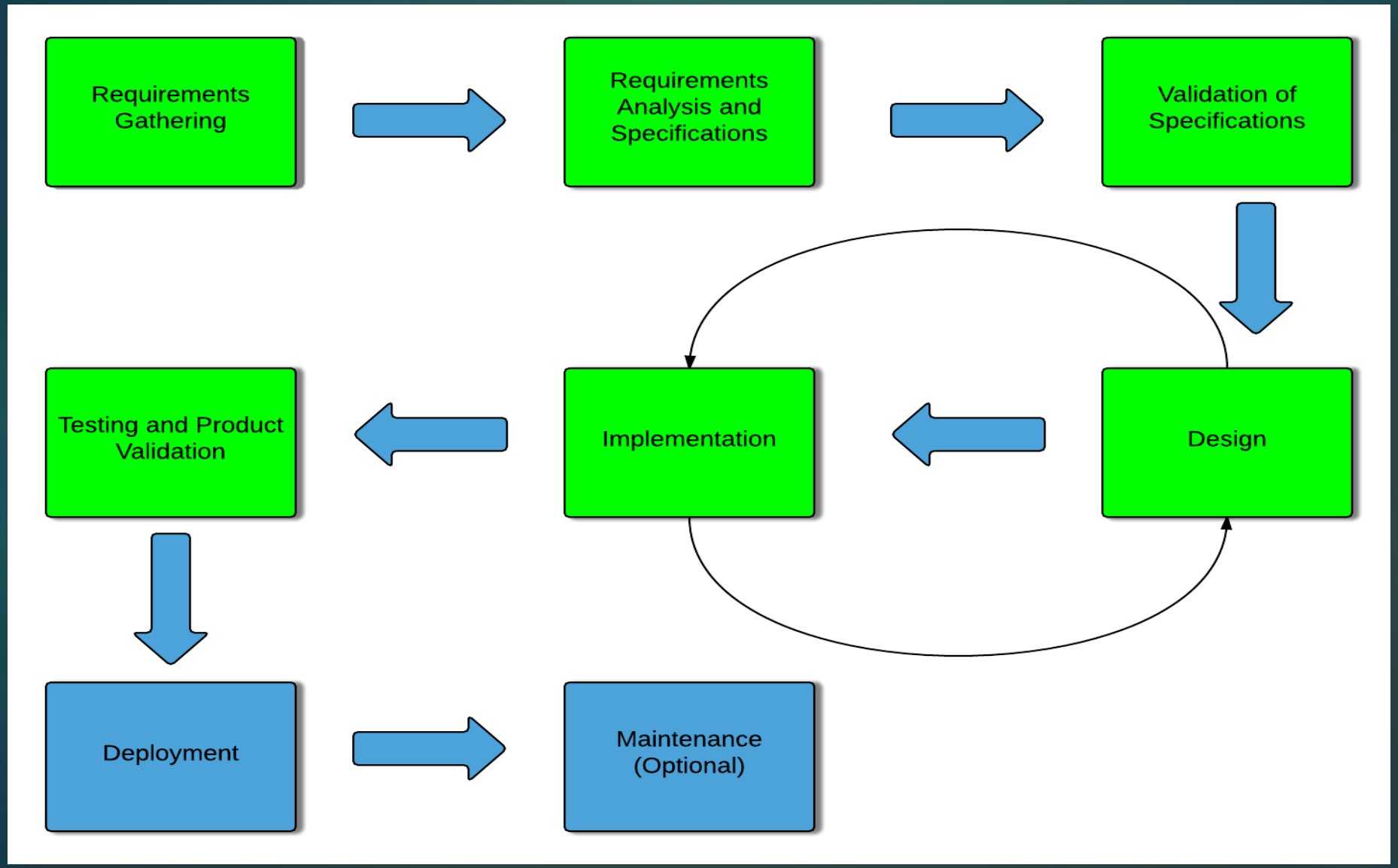
Project Deliverables

- System Design
- Algorithms
- Experimental data to prove system functionality
- System Documentation
- Prototype

Design Constraints

- Software based solution requirement(no mechanical solution)
  - Mechanical solutions were initially considered
  - 1. Springs to curb vibration
  - 2. Light shield to absorb some incoming light
- Testing Constraints
  - Light source for testing has limited light intensity range
  - Vibration plate for testing has varying frequencies, but limited amplitudes
- Hardware limitations
  - Touchscreen hardware does not allow for the modification of display light intensity

Design Process: process model



Design Process: design solutions

- Two Vibration Solutions considered initially
  - Steady state solution
  - Proportional magnification solution
- Light Solution
  - Modify display light intensity
  - Change contrast ratio with light intensity

Design Process: design solutions comparison

Proportional magnification

pros

- Can tolerate larger magnitudes of vibration
- Better for rhythmic vibration

cons

- Requires a proximity sensor
- Slightly intrusive to new users

Design Process: design solutions comparison

**** Steady state**

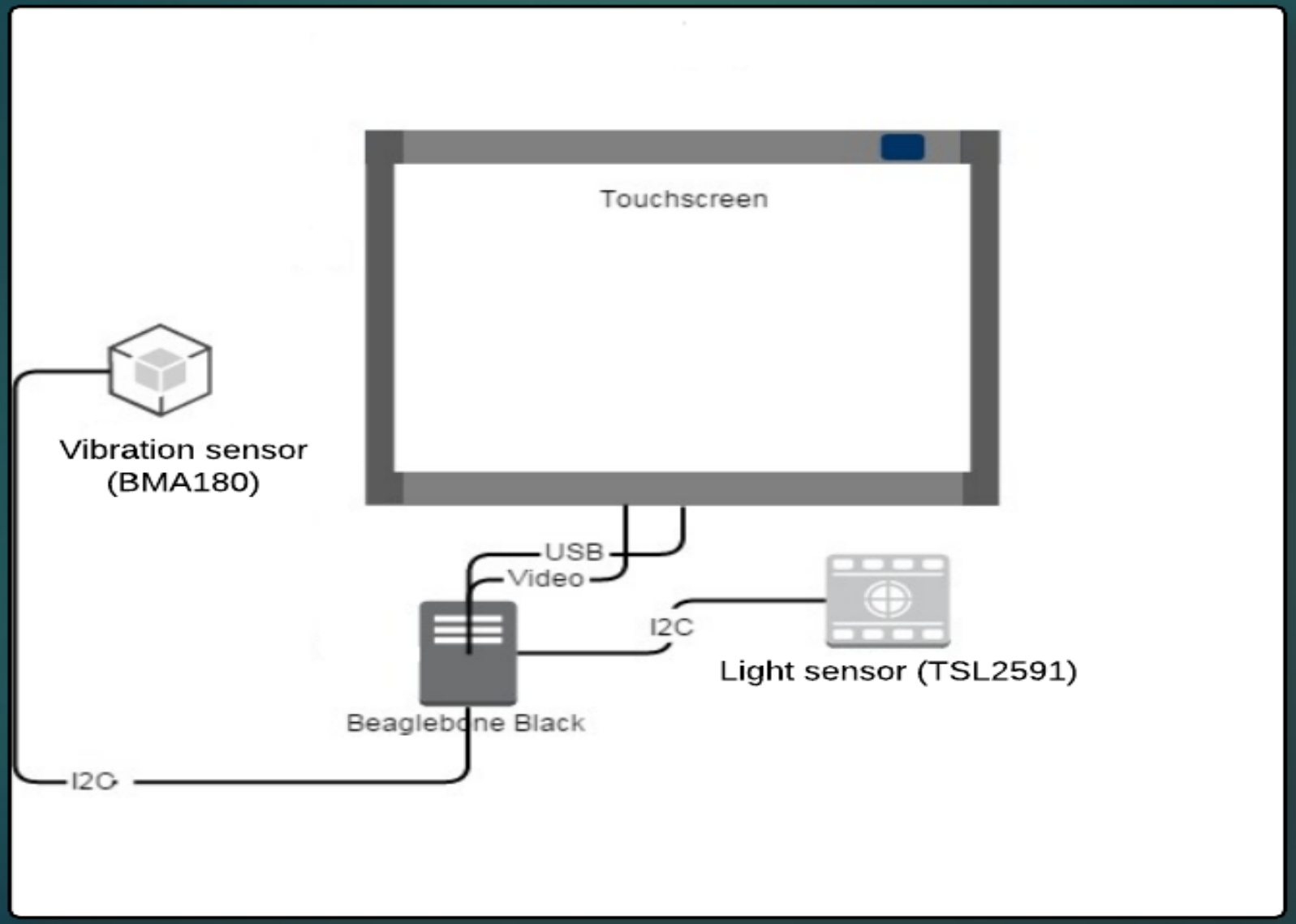
pros

- Non-intrusive
- Does not need proximity sensor
- Better for pulse vibration

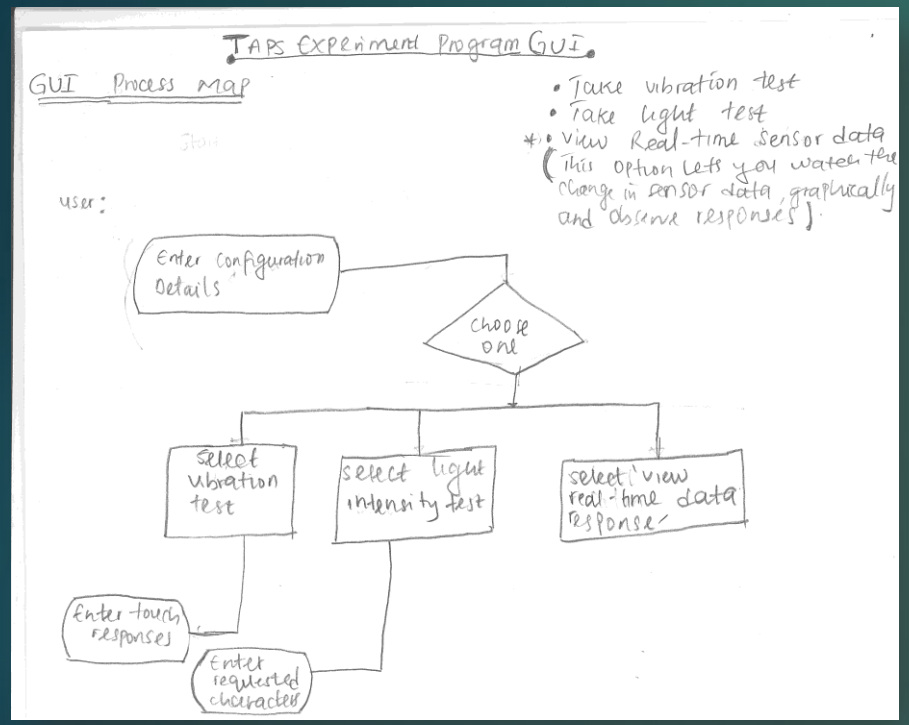
cons

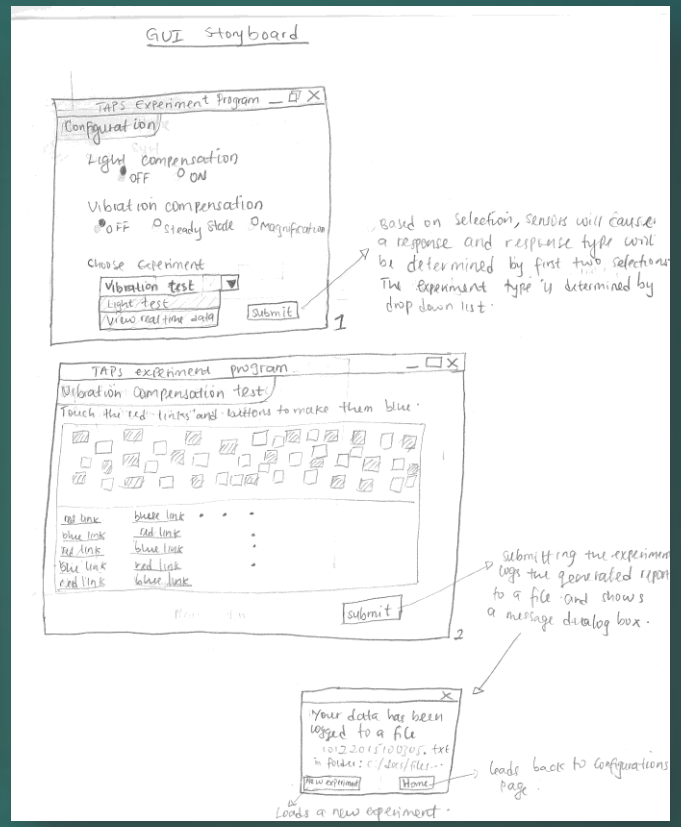
- Shrinking screen/on-screen components
- system speed requirement

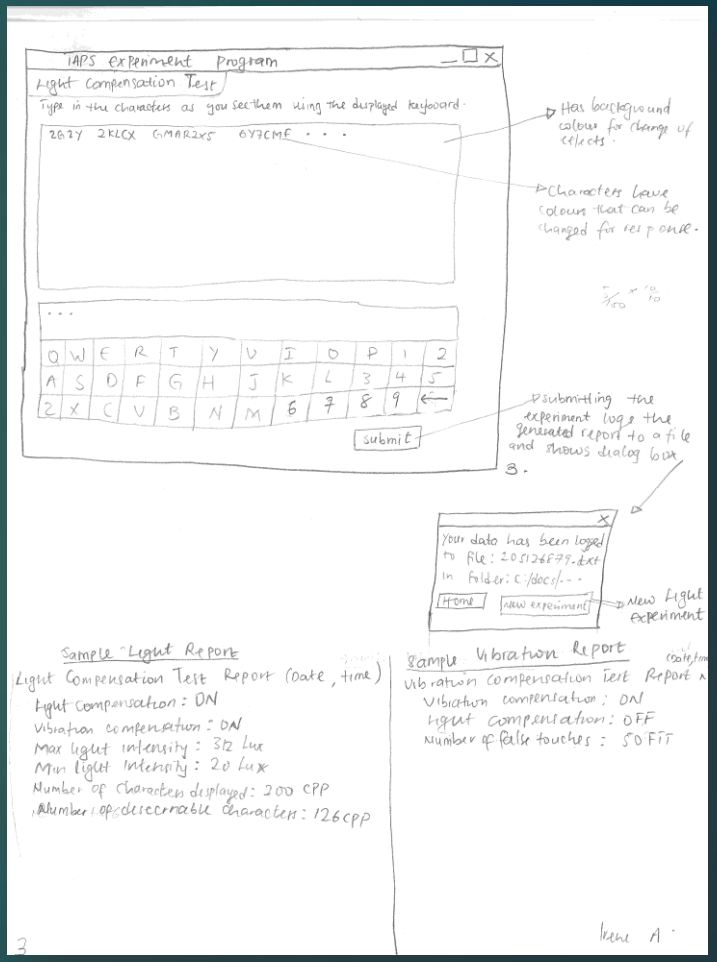
Design Process: physical configuration



Design Process: GUI storyboard







- Gui built using QT: a cross platform application framework for developing application software
- In C++

System Design: Light Algorithm

Based on

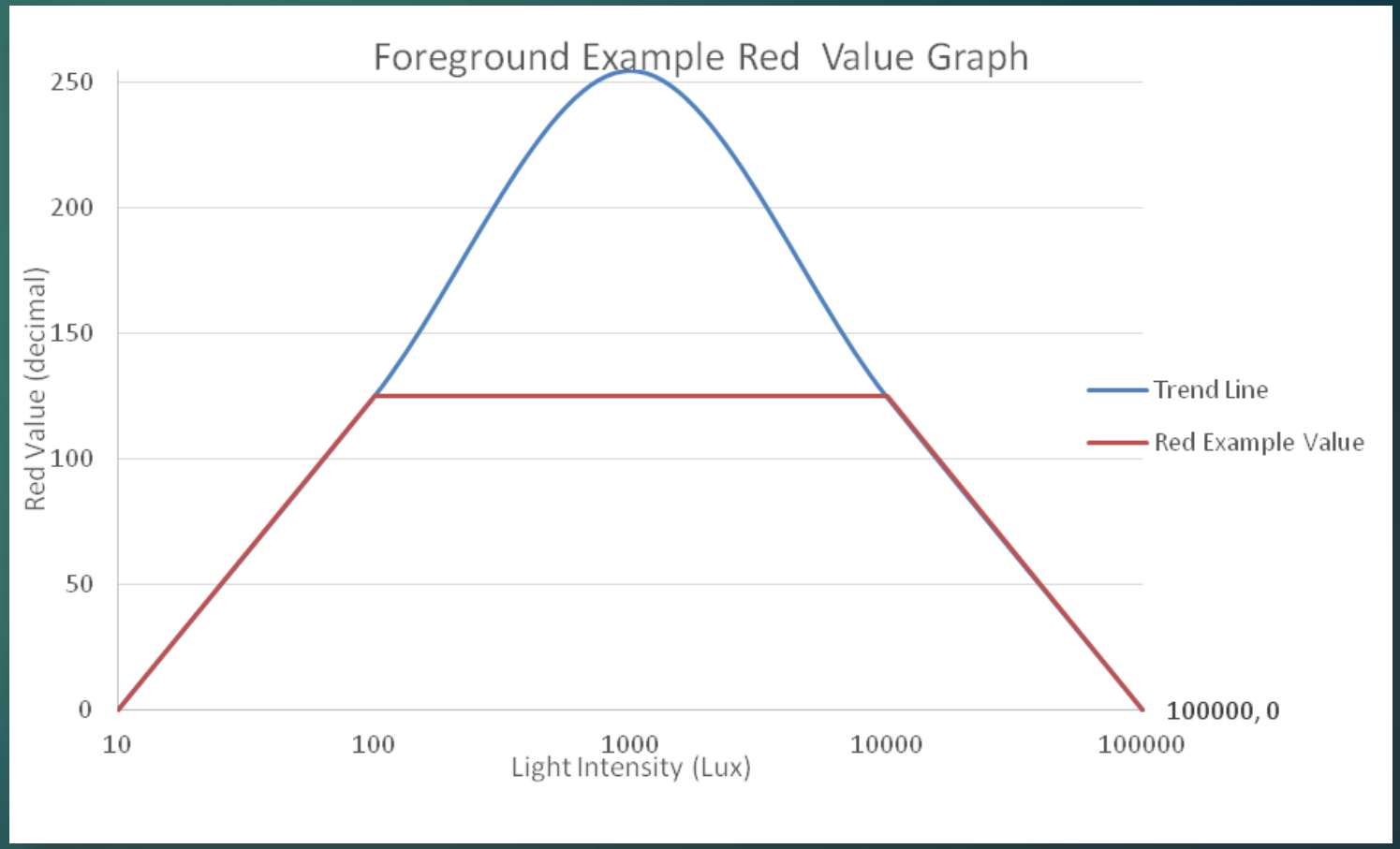
- RGB color changes
- Qcolor(int red, int green, int blue)
- Favors original color scheme

Background

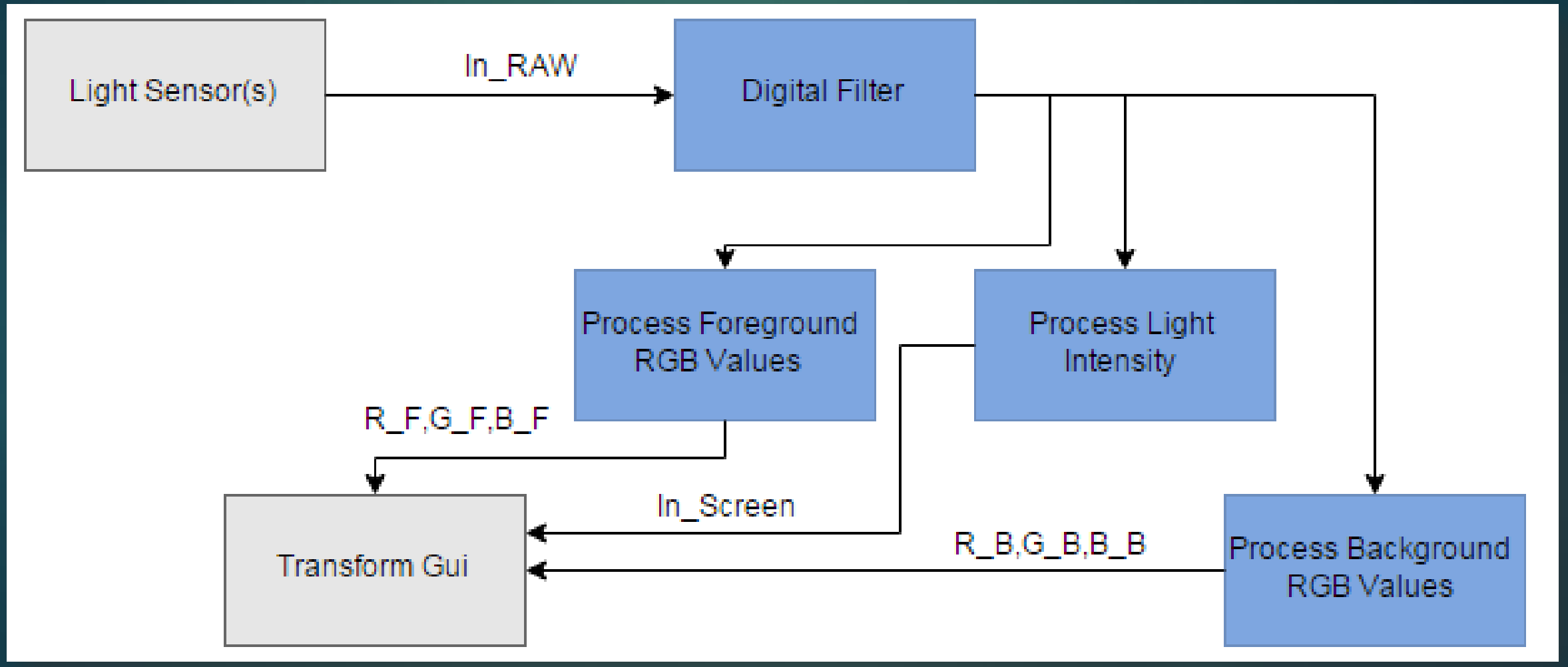
- Base #FF FF FF (White)

Foreground

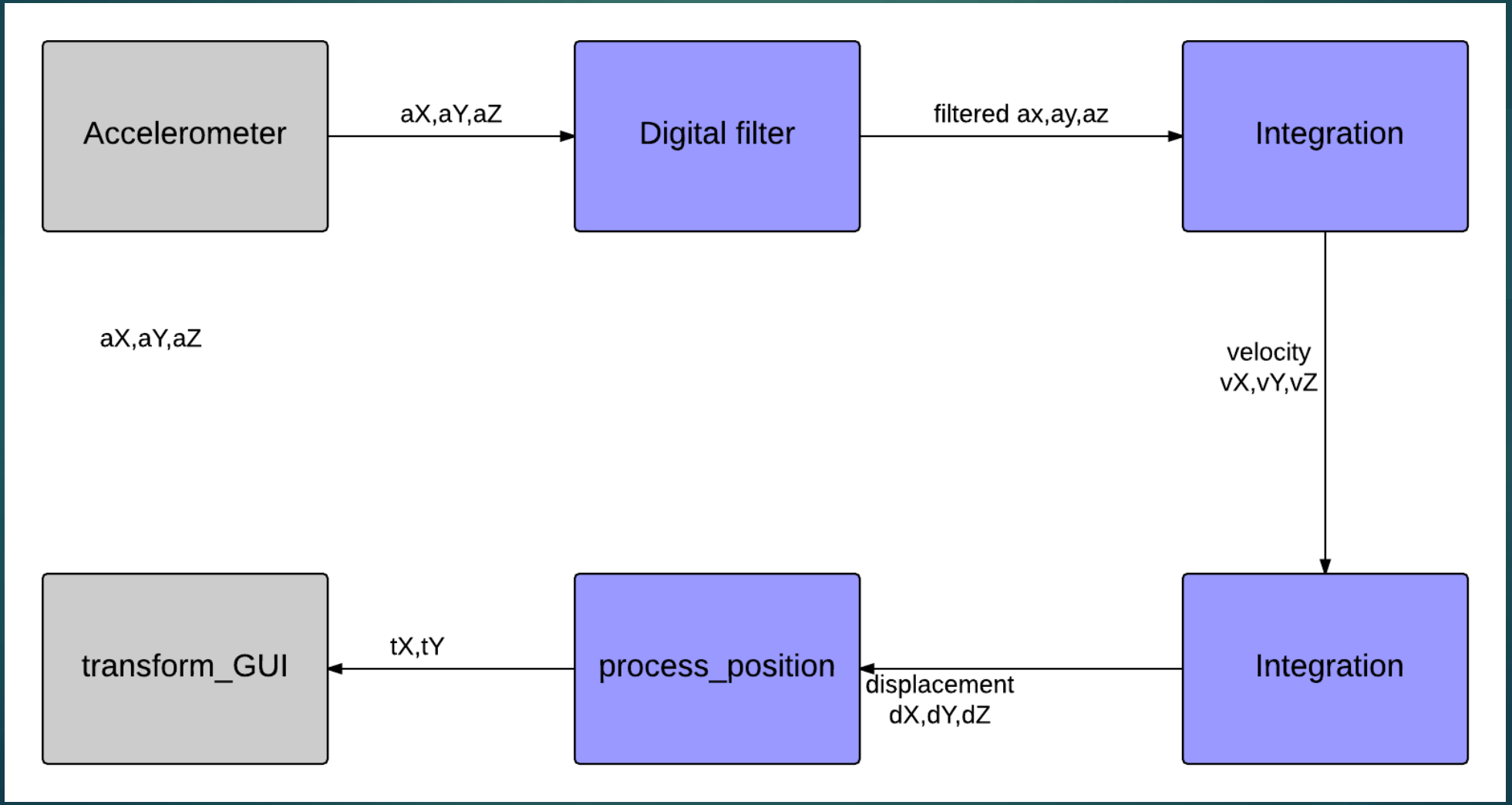
- Base #00 00 00 (Black)



System Design: light solution algorithm



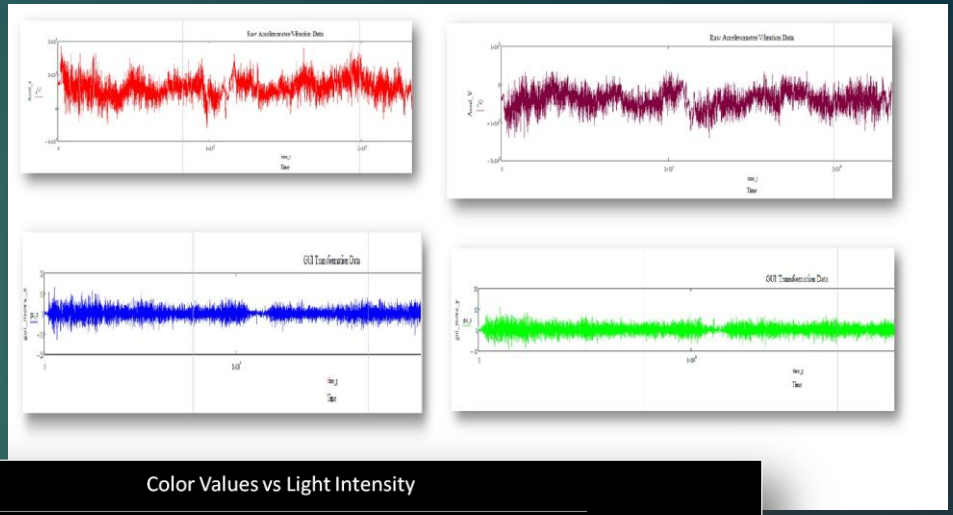
System Design: vibration solution algorithm



Data Collection and Testing

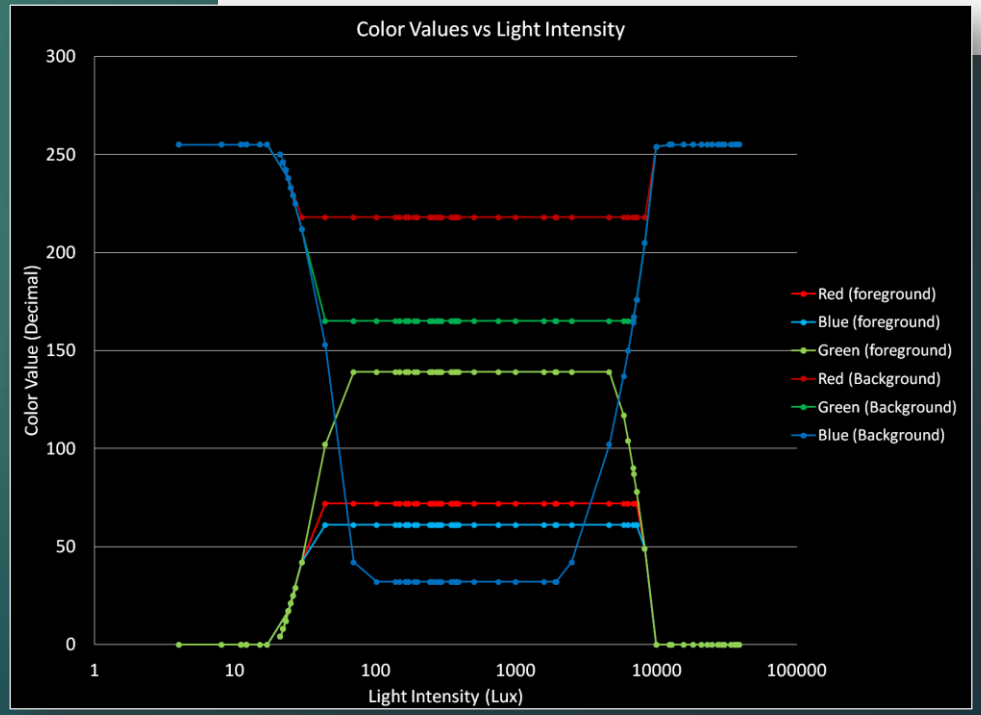
Vibration Data collection method

- Drive on a rough road with the touchscreen
- Using a vibration plate



Light Data

- Using a high light intensity lamp
- exposure to environments with different light intensities (dark room, in the sun, etc)



Results

Metrics	Units	Unit Alias	Marginally Acceptable	Ideal
Percentage of false touches	Number of false touches per 1,000 touches	FTT	20% decrease with compensation	80% decrease with compensation
Discernibility of On-screen Characters	Number of discernible characters per page	CPP	50% increase in CPP with compensation	90% increase with compensation

- Average discernible CPP **without** compensation: 65.7%
- Average discernible CPP **with** compensation: 95.8%
- %increase in CPP: 46%

- Average FTT without compensation: 0%
- Average FTT with compensation: 0%
- %decrease in FTT: 0%

DFMEA

Component	Symptom	Effect	Failure Mode	Probability of Failure	Severity of Effects
Vibration compensation module	Updates GUI at a slow rate	Vibration compensation might fail	Fast update rates cause system to crash	3	4

- probability from 1(improbable) to 4(very probable)
- Severity of failure from 1 (negligible) to 4(catastrophic)
- Sample rate limit: 200Hz

Conclusion

Our enhanced touchscreen product compensates for both the presence of vibration and changing source light intensities. Varying source light intensity is compensated for by varying the RGB color values of the GUI components to improve visibility. Vibration is compensated for by transforming the GUI to counteract the vibration experienced by the touchscreen system.

Recommendations & Evolution

- Improve the update speed for quality testing of the vibration solution
- Add a compass to more efficiently track directions of motion
- Incorporating functionality to compensate for shadows cast on the surface of the screen.
- Adding compensation for the user's motion
- compensate for light changes by varying display intensity