Fluorescence Intensity Detection Improvement for an Isothermal Diagnostic Detector

by

Yue Yu

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF APPLIED SCIENCE

in the

School of Engineering Science

SIMON FRASER UNIVERSITY

Fall 2023

Approval Page

Name: Yue (Mary) Yu

Degree: Bachelor of Applied Science (Honours)

Title of Thesis: Fluorescence Intensity Detection Improvement for an Isothermal Diagnostic Detector

_____

Dr. Cheng Li, P.Eng
Director, School of Engineering Science

Examining Committee:

_____

Dr. Michael Adachi, P.Eng (Supervisor and Chair)
Assistant Professor, School of Engineering Science

_____

Dr. Ash Parameswaran, P.Eng
Professor, School of Engineering Science

_____

Dr. Craig Scratchley, P.Eng
Senior Lecturer, School of Engineering Science

Date Approved: Jan. 18th, 2024

_____

Abstract

This thesis outlines the improvements made to the fluorescence-based isothermal diagnostic device, for the detection of COVID-19, that was previously developed by students.

The main goal of this thesis is to find a configuration that minimizes the amount of recorded noise in the baseline fluorescence sample. As the detector detects the presence of COVID-19 by comparing the amount of fluorescence within the presented sample against a baseline value, it is imperative that the baseline be as consistent and noise-free as possible to avoid an erroneous detection.

To find such a configuration, the thesis looks at various factors including heating, the use of optical filters, physical location, and variations in the number per round (count), number of rounds (cycle), and frequency (delay) of the recorded data.

Keywords: Isothermal Diagnostic Device; Fluorescence; Minimizing Error; Software

Acknowledgements

I would like to thank my supervisor, Dr. Michael Adachi, for his continued support and guidance throughout the project and for allowing me to participate in researching such an interesting topic. I'm also grateful for how he's been flexible and accommodating of the limits imposed by my circumstances.

I would also like to thank the previous students and professors who worked on the project for laying the groundwork for the system.

Lastly, I would like to thank my dad for helping me debug the problem when the PCB stopped working halfway through the testing process.

Table of Contents

List of Figures

List of Acronyms

COVID-19	Coronavirus Disease of 2019
IDD	Isothermal Diagnostic Device
LED	Light-Emitting Diode
PD	Photodiode
PCB	Printed Circuit Board
GUI	Graphical User Interface
PID	Proportional-Integral-Derivative
RMS	Root Mean Square
STD	Standard Deviation
RSE	Residual Standard Error
DC	Direct Current

Chapter 1. Introduction

The COVID-19 pandemic showed that having a cost-effective and convenient detection device is very important in helping to save the lives of many people. The fluorescence-based detection system improved on in this thesis is one such device.

The device is an isothermal diagnostic device (IDD) previously made by students under the supervision of Dr. Michael Adachi. It uses spectrophotometry to detect the presence of COVID-19 in a fluorescent-dyed biological sample by using light-emitting diodes (LEDs) of a particular wavelength. The refracted light is detected by photodiode sensors (PDs) and compared with a baseline value to diagnose the sample [1].

1.1. Background

The general system, shown in figure 1.1, includes a 3D printed enclosure, a printed circuit board (PCB), an Arduino Mega board, and a graphical user interface (GUI). The enclosure houses a heater, 8 LEDs, and 8 PDs, to allow for the diagnoses of 8 samples at once. It is connected to the PCB and the Arduino board, with a power supply connected directly to the PCB. The Arduino board can be controlled and monitored by USB and the output data is relayed to a python program which displays the data using the GUI.

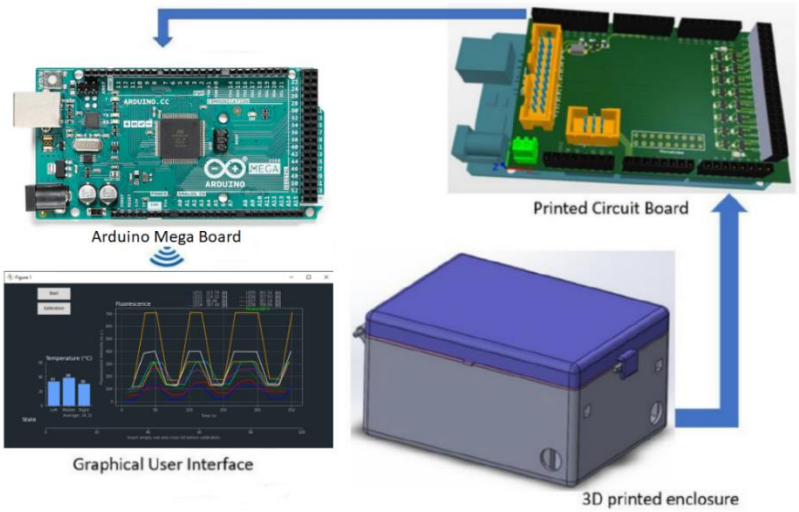

Figure 1.1: System Overview showing the 4 Main Components [2]

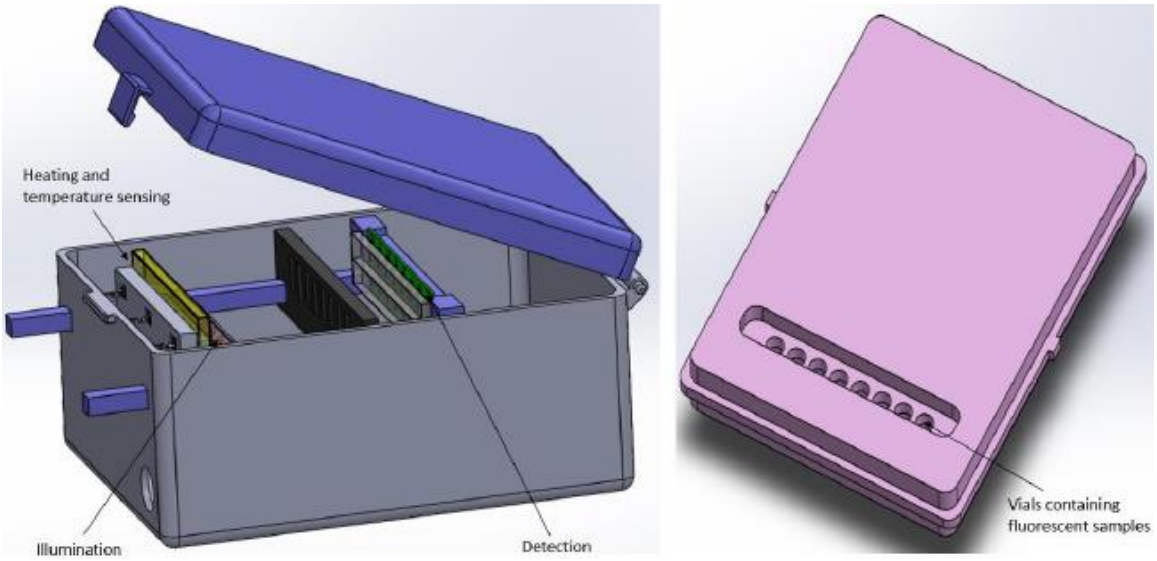

Figure 1.2: Design of 3D Printed Enclosure [2]

The left of figure 1.2 shows the design of the inside of the enclosure. The LEDs used for illumination are placed on the left side of the enclosure, by the heater. On the opposite side, the PDs are placed for detection.

The pink lid shown on the right side of the figure is used as an inner cover and is placed right below the blue lid of the enclosure. There are places on the pink lid to place 8 vials with fluorescent samples. The bottom of each of the 8 samples will line up with the 8 LEDs inside the enclosure and the heater will be right next to the samples so that they can be kept at the target temperature of 41 °C.

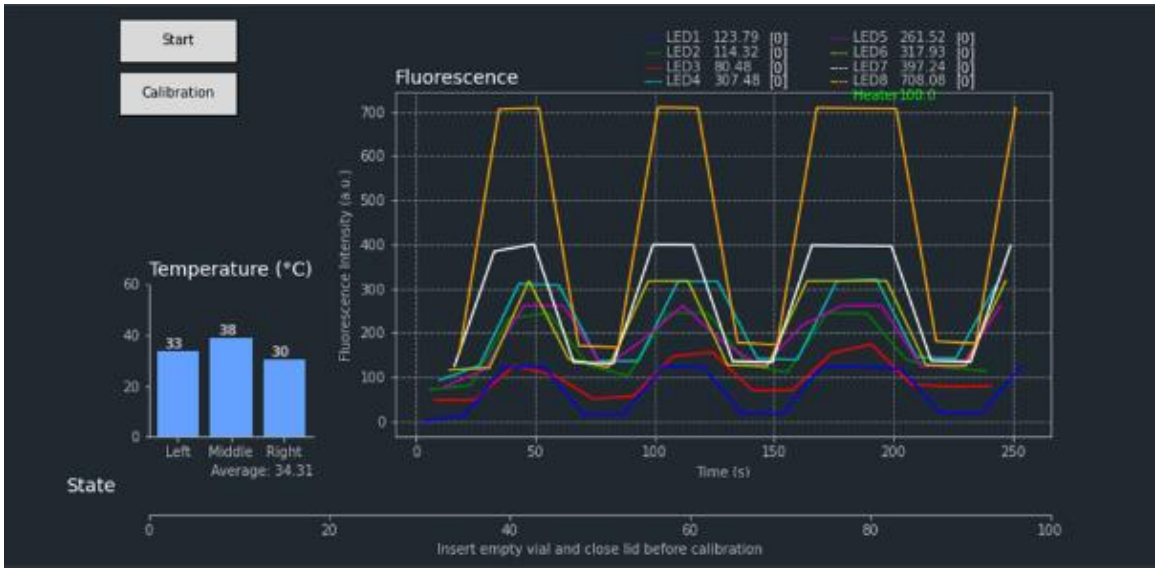

Figure 1.3: GUI, made using Python Code and displaying the data collected by the sensors.

Figure 1.3 shows the GUI for the system. It displays the Fluorescence graph, where the PD readings for each LED are graphed onto the same graph. It also includes a temperature bar graph that shows the most recent temperature readings from three sensors located at different locations inside the enclosure. The actual temperature is taken as the averaged of the three different temperature sensors. There is also a state bar that shows the current state of the system.

There are mainly 2 states, controlled by the “Start” and “Calibration” buttons. When the samples are placed into the vials, the “Calibration” button is used to set a baseline for each LED value (aka values recorded by the LED’s respective PD). The “State” label will change to “Calibrating” and the bar will fill up by 10% each time all 8 LED values are updated. When the bar is at 100% (when there’s 10 readings per LED), the new baseline for each LED will be calculated as the average of the 10 readings taken, and the graph will reflect the change. This will be done by graphing the difference between the raw data and the calibration value instead of just the raw data itself. As the graph can get quite messy, the ”Start” button resets the graph to a clean state while maintaining the calibration information.

The system has a few areas that can be improved upon, including the heater controller, general code organization, and GUI. But the most impactful improvement will be to find a configuration that would minimize the recorded baseline errors, because that would directly help to decrease the chance of an incorrect diagnosis.

Chapter 2. Problems and their Solutions

Before the optimal configuration can be found, the other areas in the system should be improved on to provide a better environment for the later tests.

2.1. Heater Improvements

Due to the sample's temperature sensitivity, the goal of the heater controller is to keep the samples within a 1-degree limit of the 41 °C target [3]. For this task, the system previously implemented a proportional-integral-derivative (PID) controller. However, from testing via manual manipulation of the PID value, it is found that an input value of 254 (out of 255) produced the same result as an input of 0; both putting the heater in an "off" state. The heater is in an "on" state only if the input is 255. This evidence suggests that the system only has 2 states and cannot be controlled with a PID system as previously thought.

In addition, when the heater changes state (from "on" to "off" and vice versa), either the light of the LED would dim, or the displayed temperature value would spike, as shown in figure 2.1. Figure 2.1 also shows that the fluorescence intensity readings correlate closely to the temperature reading graph and may also spike when the heater state changes.

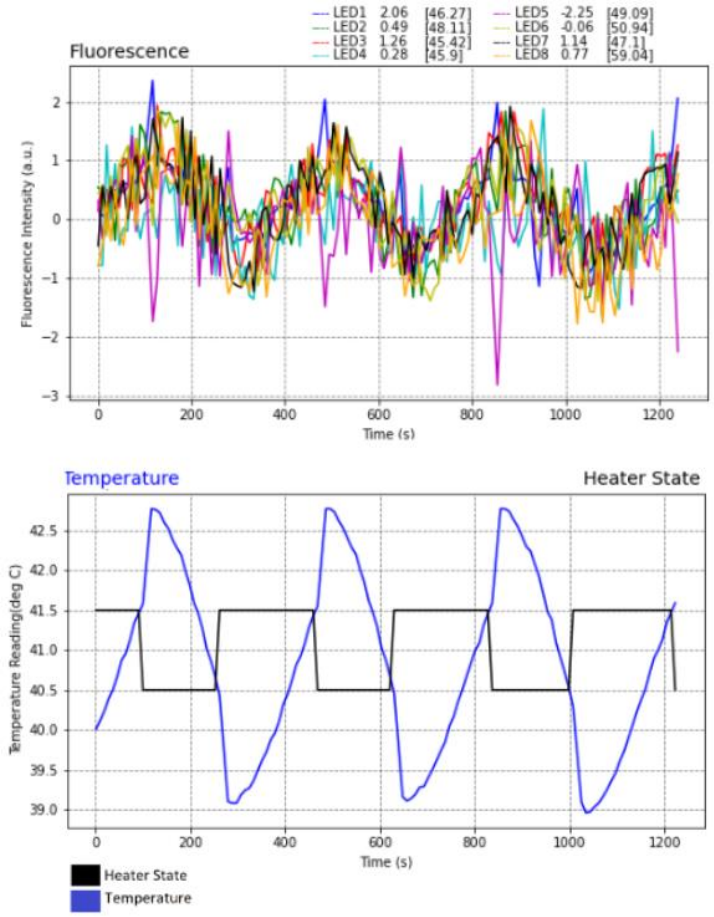

Figure 2.1: Temperature Reading Graph Spikes when the Heater State Changes. Also shows impact on Fluorescence Intensity Graph.

This behavior is due to the inductive or capacitive load placed on the power supply when the state changes. The effect of the power supply on the system is further shown in figure 2.2, where instead of changing the heater state from on to off, the power supply is plugged and unplugged. Here, during the unplugged state, the displayed temperature jumps up and down, behaving vastly differently from figure 2.1.

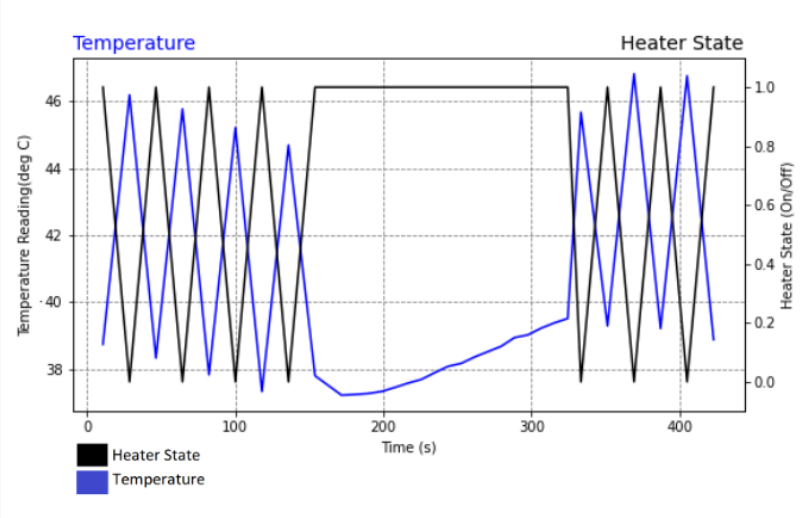

Figure 2.2: Temperature Graph for when the Power Supply is Unplugged vs Plugged. The temperature spikes up and down when the power supply is unplugged, which differs from simply turning off.

Thus, to lessen the impact of that heater state change has on the system, maintaining the system's state for as long as possible would be preferred. Furthermore, the solution should also allow the target temperature to be reached within an acceptable time frame (~10 mins) when starting from room temperature.

The first part of the solution is to improve the insulation of the system. In this project, this was done via the use of foam tape. It can be seen from figure 2.3 that the time it took to reach the target temperature decreased from about 1750 seconds to about 850 seconds. This still falls outside of the desired 10-minute time limit but is, nonetheless, a great improvement. Improving the insulation also had the added benefit of maintaining the temperature for testing and decreasing the amount of perceived noise in the temperature graph.

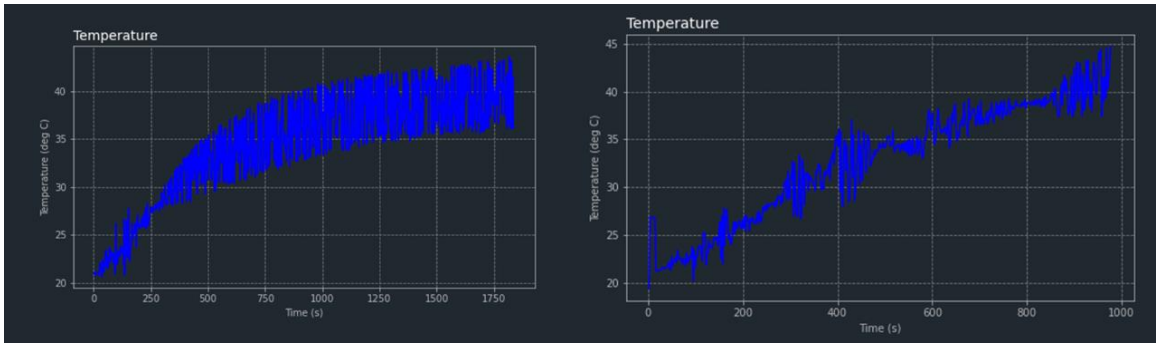

Figure 2.3: Temperature Graph Before (left) vs After (right) the use of Foam Tape

The other part of the solution, the implementation of the controller itself, was simple. To maintain the temperature, it was kept oscillating between a floor value of 40.5 °C and ceiling value of 41.5 °C. A heater state change is triggered when the temperature passes either limit.

2.2. Coding Improvements

The software of the device is divided into two sections: the Arduino code and the python code. The Arduino code's purpose is mainly to control the inputs and outputs related to the LEDs, PDs, and heater. It also passes the processed data to the python section of the code. The python section mainly deals with displaying the data provided onto a graph that updates in real-time. It also has a calibration function and a start button, to allow the user to specify when to calibrate and start (or restart) the test.

The changes made on the python side were relatively minor. It includes adding the data exporting function and making some GUI changes, which can be seen in figure 2.4. The GUI changes include color changes in the graph, the heater intensity label being removed (as it is always 0% or 100%) and adding a temperature graph and the heater state graph.

The temperature graph was probably the most important addition in this part. It allows the temperature vs time of the system to be seen instead of just being able to access the most recent temperature data. This change was fundamental in helping with the heater changes of the section above.

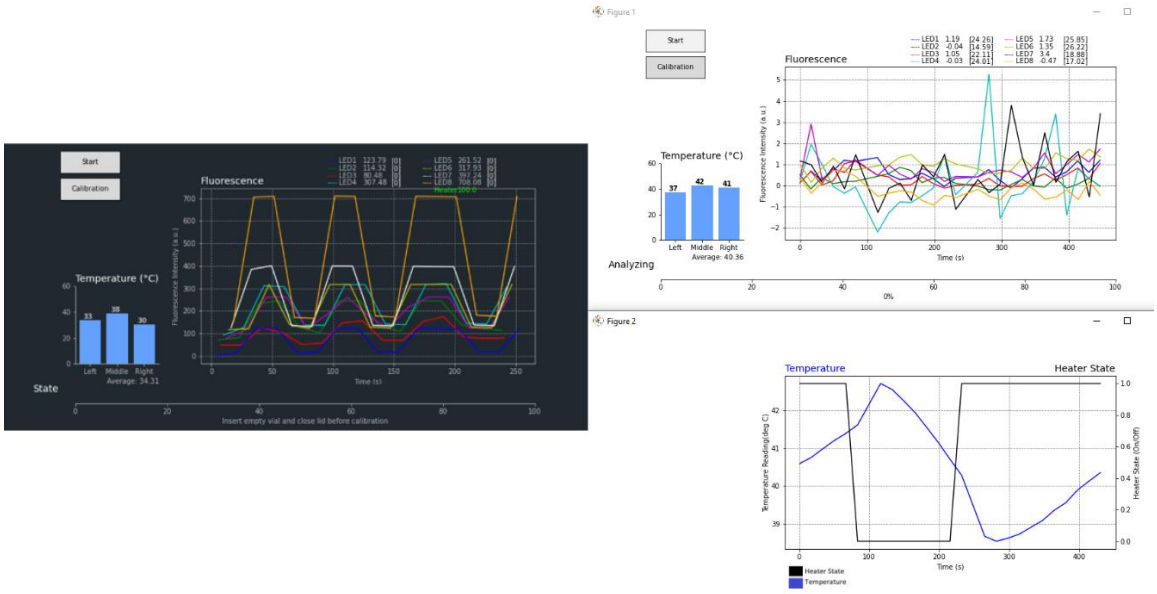

Figure 2.4: Comparing the Old GUI (left) and the New GUI (right), which added color changes and a separate graph.

As the Arduino code controls the system's inputs and outputs, it is important for the code to be concise and readable to decrease the chance of bugs and to allow for new features to be implemented easily. Therefore, the previous code needed to be cleaned and rewritten in many parts.

The old code used switch cases that dealt with each of the 8 pairs of LEDs and PDs in turn. However, it basically acted like a large loop with the variable "count" as the counter. The code for each of the 8 sections were long but were copy and pasted sections differing only in the pin numbers. This allowed for an increased change in error as every change that involved the LEDs/PDs need to be changed in 8 different places.

The code was condensed into an array that ran through a loop, so the functionality was the exact same. The long sections of code were also divided up into smaller functions with names that were clear and easy to understand. After the changes, the code was shortened to 54% of its original length.

When rewriting the code, there were a few sections that needed to be removed because they were found to be useless. An example of this is the code related to the PID controller, which was already discussed earlier. Another example is the code related to the variable "delaytime", which is a delay added between different photodiode reads. The variable was tested and found to not have any effect on the behavior of the system.

There were also bugs in the code. For example, while the LED values were averaged correctly to decrease the appearance of noise, the temperature vales were not averaged at all. This caused noisy data that would wrongly trigger heater state changes. After changing this, the temperature graphs showed a significant decrease in the amount of noise (figure 2.5).

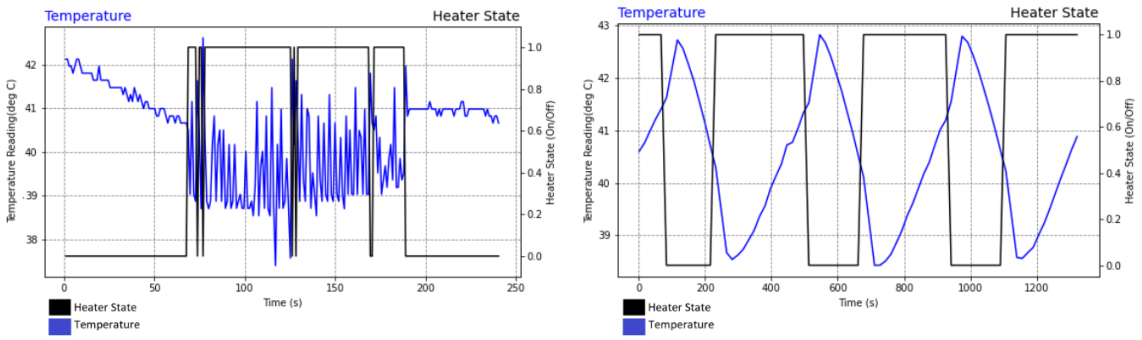

Figure 2.5: Comparing Old Temperature Graph (left) vs New Temperature Graph (right). The old, unaveraged graph is a lot noisier.

Figure 2.6 shows the block diagram for the rewritten Arduino code. First, the current PD and heater values are read and saved a set number of times (total count in the figure, or the “count” variable referred to later). After all 8 PDs’ values are read, the heater’s state is considered. If the heater reaches the temperature ceiling of 41.5 °C and is on, or the temperature floor of 40.5 °C and is off, the state will be changed. All the data are averaged and sent to the python part afterwards.

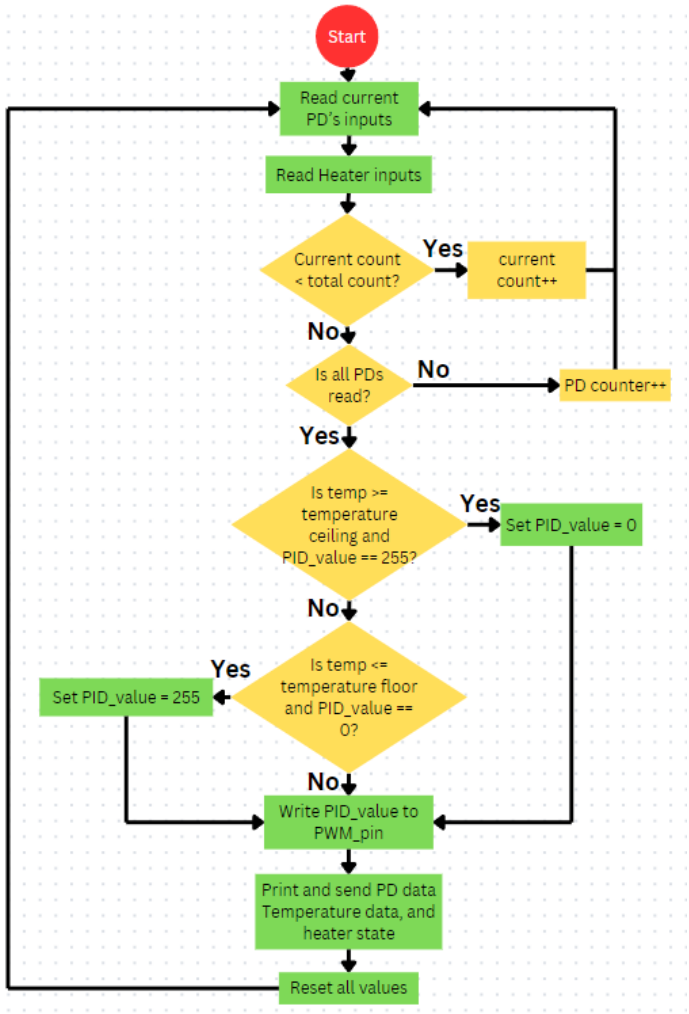

Figure 2.6: Block Diagram of the New Design of the Arduino Code, accounting for Temperature Averaging

2.3. Physical Sensitivity of the System

During testing, the system was found to be very unstable at times and would contain very large spikes in readings that could skew the rest of the data tremendously. It was found that the largest contributing factor is its physical placement.

The system was so sensitive that a slight bump into one of its many connecting wires would register a reading as that shown in the 3rd cycle of figure 2.7.

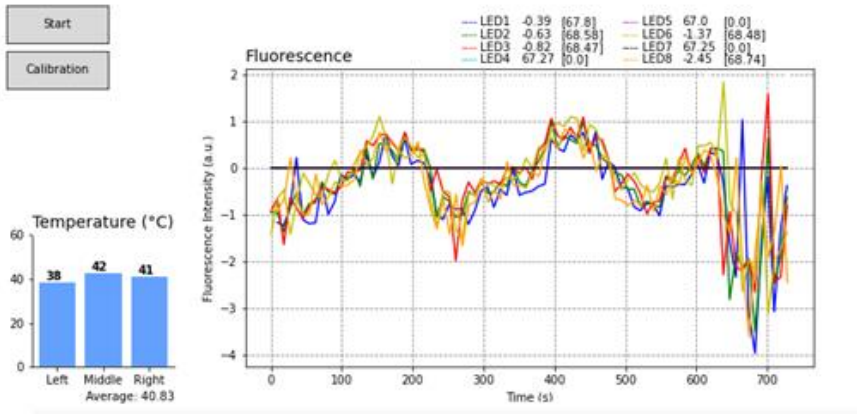

Figure 2.7: Fluorescence Reading with Slight Physical Interference, shows that the system is sensitive.

To limit the physical interference as much as possible, the testing area was readjusted so that the whole system is packaged in a box and anchored to a wall, leaving only the USB and power supply wires coming out from it. The wires were also in a location that would not be touched during regular testing.

This resulted in a vast improvement in consistency throughout the tests.

However, this is a temporary solution to the problem that only works in the context of the tests and does not help the goal of ultimately making the device a convenient detection device. Upon further investigation, the system's sensitivity is most likely due to how the wires are connected between the PCB and the sensors inside the 3D enclosure.

Figure 2.8 shows that the wires are connected by twisting together and insulated using blue tape. However, such connections may become loose at a touch. A better solution is to solder the wires together instead.

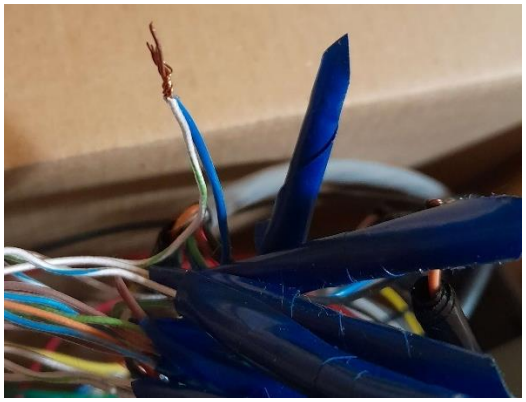

Figure 2.8: Suspected Cause of Unstable Connections and Sensitivity Issues

Chapter 3. Main Goal and Design

With the problems above dealt with, the plan to minimize the overall noise in the baseline fluorescence intensity readings can commence. Since there are no real samples placed in the device during testing, the main goal will be to minimize the fluorescence intensity of the outputs by altering the inputs.

Each test is referred to in the form count-cycle-delay. When displaying the results of a series of tests focused on a single output, the variable ‘x’ represents the variable that is altered. For example, if the line was 100-1-x, the count and cycle remain as 100 and 1 respectively, while the delay is changed.

3.1. Inputs

The inputs are the variables count, cycle, and delay.

3.1.1. Count

The count variable determines how many times a single LED’s value is recorded and averaged over before proceeding onto the next LED. For example, if the count is 200, then LED1 will be read 200 times in a row, the average value from the 200 readings will be calculated and recorded, and then it will proceed to LED2.

3.1.2. Cycle

The cycle variable determines how many cycles the value is averaged over before sending the value to the GUI. One cycle is complete when all the LEDs have been read “count” number of times once. So, for example, if the cycle is 4, all the LEDs’ values would be completely read and averaged 4 times over, and the average of the 4 averages for each LED will be sent to the GUI.

3.1.3. Delay

The delay variable is the amount of time (in milliseconds) between each “count”.

3.2. Outputs

The main outputs recorded are maximum (residual) peak-valley value, root mean squared (RMS), and standard deviation/residual standard error (RSE).

3.2.1. Maximum (Residual) Peak-Valley Value

The maximum peak-valley value is the difference between the maximum and minimum recorded value within a test. As the recorded fluorescence graph may trend upward/downward due to some uncontrollable factors affecting the system, the maximum residual peak-valley value is actually calculated instead and taken to be the maximum peak-valley value under normal circumstances.

3.2.2. Root Mean Squared

The root mean squared value presents the root mean squared of the original recorded data.

3.2.3. Standard Deviation and Residual Standard Error

The standard deviation considers the fact that the values may not center around zero. The residual standard error is actually calculated instead, to account for possible sloping, and taken as the standard deviation under normal circumstances.

3.3. Main Design

The main goal of the experiment is to limit the fluctuations detected in the fluorescence intensity through the manipulation of count, cycle, and delay. The tests are designed to consider each variable in turn. For example, one test may hold cycle and delay the same while altering count, and another would hold count and cycle the same and alter delay.

As the data appeared cyclical and contained random spikes, the data is mainly taken over a minimum of a 3-cycle duration so that the instability due to the different configurations would be factored into the result. Right after the heater changes state, the first value is filtered out as there would be an expected spike caused by the constraints of the hardware.

Each set of data is taken multiple times (up to 4) and in a different order each time. The order for the second set is always the reverse of the first set, while the orders of the subsequent sets are random. The final result is an average taken over all the sets to factor in the changes in the system over the course of a few days, as well as the changes over the course of the test itself.

After exporting the data into excel, the trendline (slope) for each LED's data is found and the maximum (residual) peak-valley value, RMS, and STD (RSE) are calculated. A value representing all 8 LEDs is calculated and the result is recorded and graphed.

The LED's representative value is calculated by ignoring the maximum and minimum of and averaging over the rest. The reason why the maximum and minimum values are ignored is to give some leniency when it comes to spikes, so that the appearance of one rogue LED would not affect the result that much. This is especially true when there are certain LEDs that tend to spike more than others and seemingly randomly. However, if there are more than one such LED, then it should be considered that it may be part of the instability caused by the configuration itself.

At the end, the graphs for each of the different tests are compared over the variable of time per read, which is the amount of time it takes for all the LEDs to update once on the GUI. Not only does this variable provide a common link between all the tests, but it also provides important information within itself. If the time per read is short, the time averaged over is not very large, so it is expected that there would be more noise. However, if the time taken is too long, the datapoints per heater state could be too small a number and may obscure information.

3.4. Other Considerations

There are also several other factors that needs to be taken into consideration when designing the test. These include the computing capabilities of the system, the power supply used, and the optical filters.

3.4.1. Computing Limitations

There are a few computing limitations that need to be aware of. Firstly, if the Arduino sends too much data to the python's GUI, the GUI will freeze indefinitely. Therefore, this puts a limit on how small the variables can be for the test.

Secondly, if the Arduino sends too low an amount of data to the python, the GUI sleeps and may randomly crash if too many user inputs are registered while sleeping (such as pressing the calibration or start button or moving the window). This problem doesn't affect the making of the test much but should be noted if the system were to be used by others in the future.

3.4.2. Power Supply

There are two available power supplies – an adjustable lab power supply and a direct current (DC) constant 7.5V 750mA power supply. When using the lab power supply, the LEDs would dim when the state of the heater changes, affecting the fluorescence intensity detected. The other power supply did not show this problem. However, it did cause the recorded temperature to spike whenever the heater changes state (though the actual temperature remained unchanged).

As fluorescence intensity is the main unit of measurement when it comes to determining the result, the 7.5V DC power supply was chosen.

3.4.3. Optical Filter

The optical filters used are a short pass and a long pass filter, located as shown in figure 3.1.

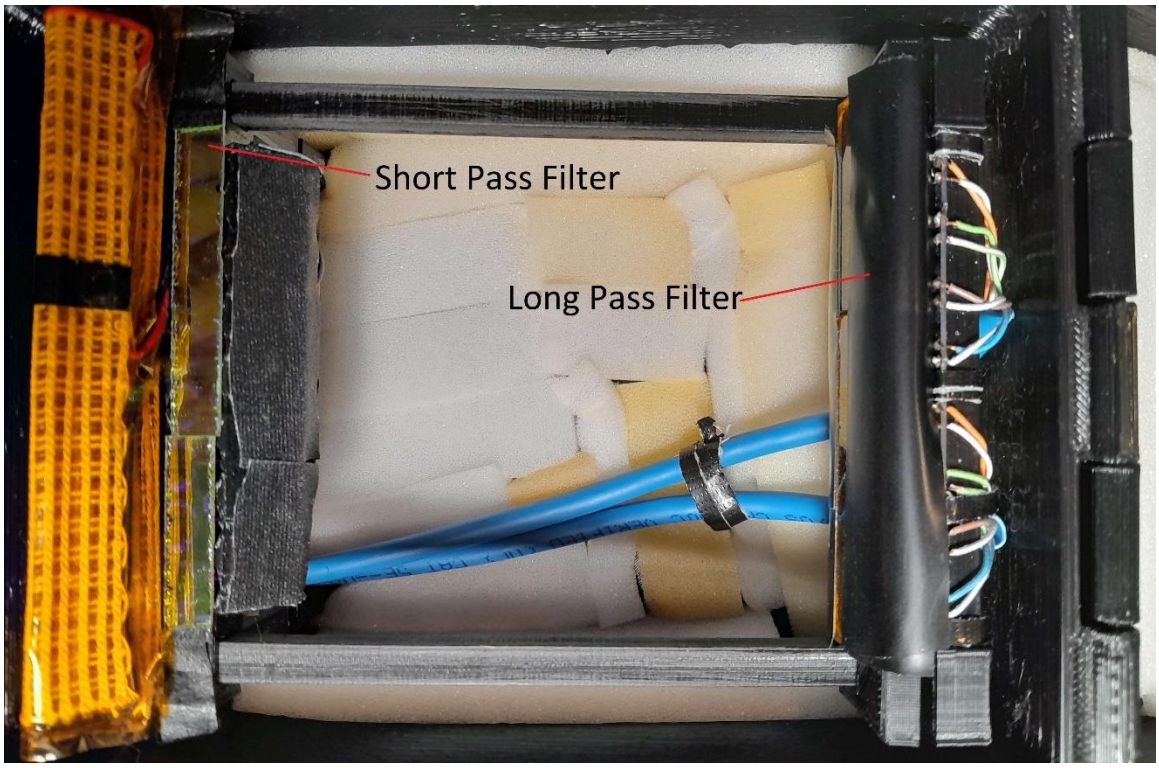

Figure 3.1: Placement of Short Pass and Long Pass Filters inside the 3D Enclosure

The filters narrow in on the selected wavelength and dramatically changes the magnitude of the fluorescence data, as shown in figure 3.2.

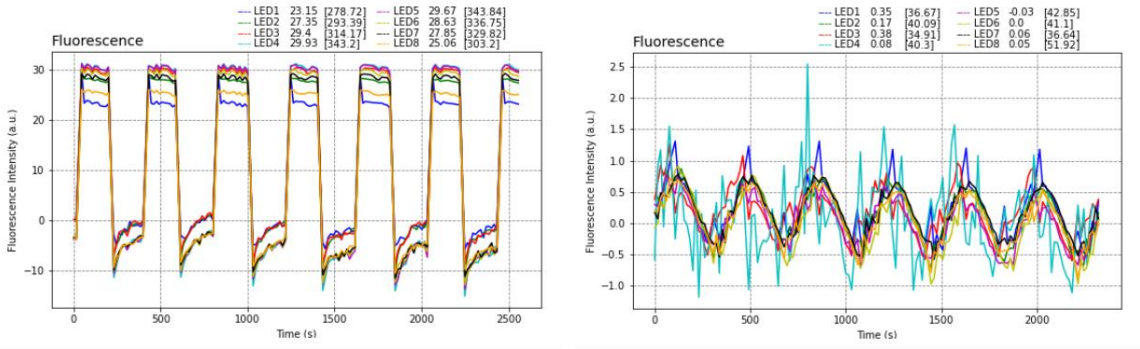

Figure 3.2: Comparison of Fluorescence Graph without Optical Filter (left) and with Optical Filter (right) for the Configuration of 300 counts, 1 cycle, and 9 delay. The amplitude of the unfiltered graph is higher.

An increase in count and delay also has a different effect depending on whether the optical filters are used or not.

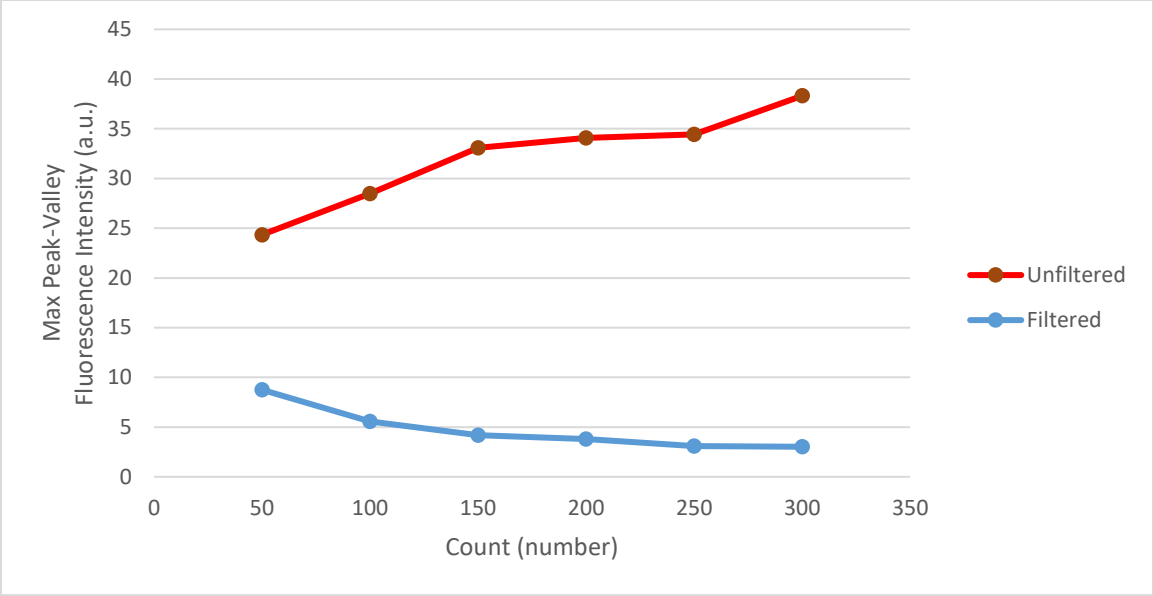

Figure 3.3: Max Peak-Valley Graph for Unfiltered vs Filter for the Count-Varying x-1-5 Line. As count increases, the max peak-valley increases for the unfiltered line and decreases for the filtered line.

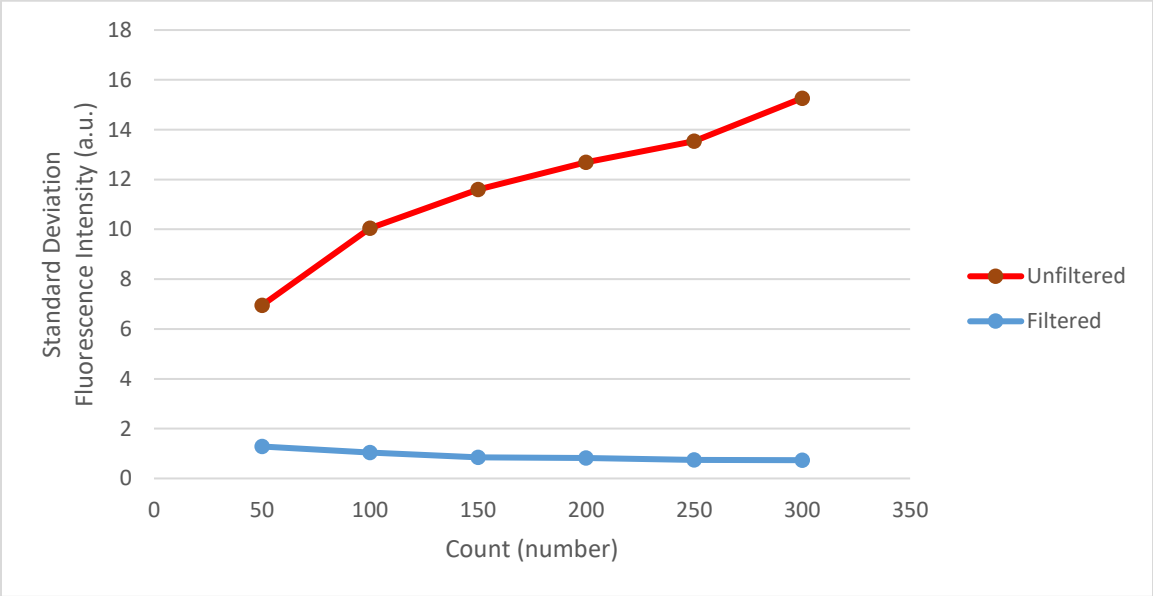

Figure 3.4: Standard Deviation Graph for Unfiltered vs Filter for the Count-Varying x-1-5 Line. As count increases, the standard deviation increases for the unfiltered line and decreases for the filtered line.

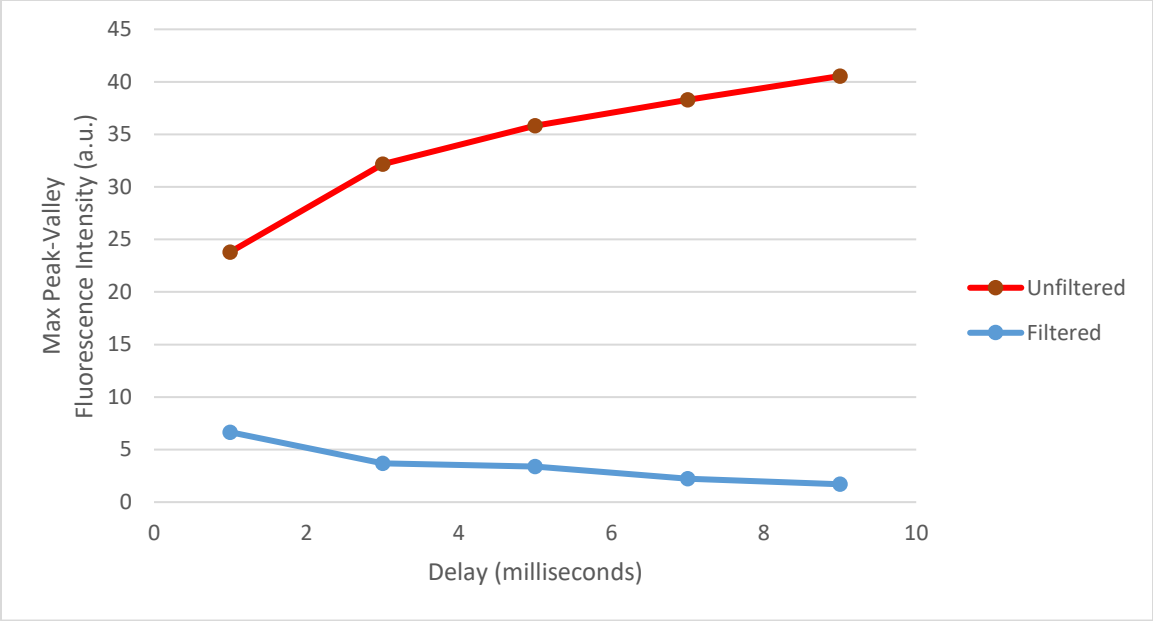

Figure 3.5: Max Peak-Valley Graph for Unfiltered vs Filter for the Delay-Varying 200-1-x Line. As delay increases, the max peak-valley increases for the unfiltered line and decreases for the filtered line.

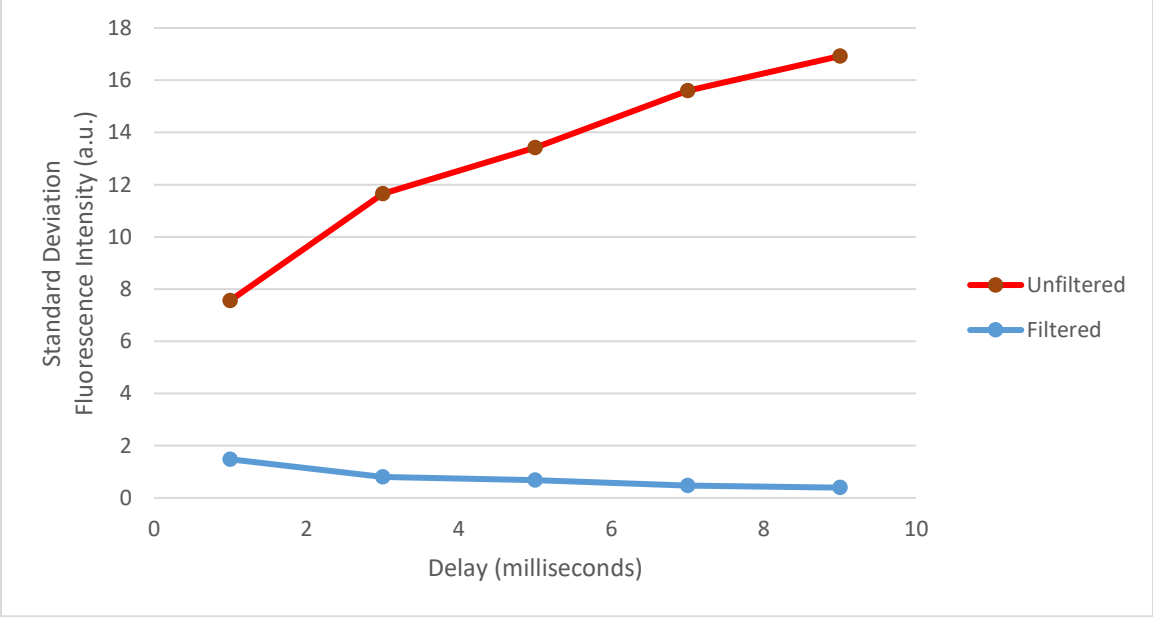

Figure 3.6: Standard Deviation Graph for Unfiltered vs Filter for the Delay-Varying 200-1-x Line. As delay increases, the standard deviation increases for the unfiltered line and decreases for the filtered line.

As can be seen here in the comparison of the graphs above (figure 3.3 and figure 3.5 for the max peak-valley fluorescence intensity and figure 3.4 and 3.6 for the standard deviation of the respective graphs), when count and delay increases, the max peak-

valley and standard deviation both decreases in the unfiltered case, but both increases in the filtered case.

These results are expected because the filters filter out large amplitudes and decreases the likelihood of extreme spikes in the data. When count or delay is low, there is more jitter in the data because the data is averaged over less points, but the overall amplitude is smaller because the data is more responsive and can detect changes in the temperature faster. As the count and delay increases, the unfiltered result sees an increase in error because of the increase in amplitude, while the filtered result sees a decrease because of the decrease in jitter.

The main problem with using the filters is that it was hard to secure the long pass filter. Tape was used to put it in place but, as the enclosure needed to be maintained at a high temperature, the tape often fell, taking the filter with it. This is even more problematic since it is hard to notice and would produce vastly different results. The solution implemented was to use an elongated piece of tape and secure it between two sliding frames.

Chapter 4. Results

4.1. Main Results

Here are the results of the experiment, comparing several series of data to each other. The series were chosen based on preliminary testing and for easy comparisons.

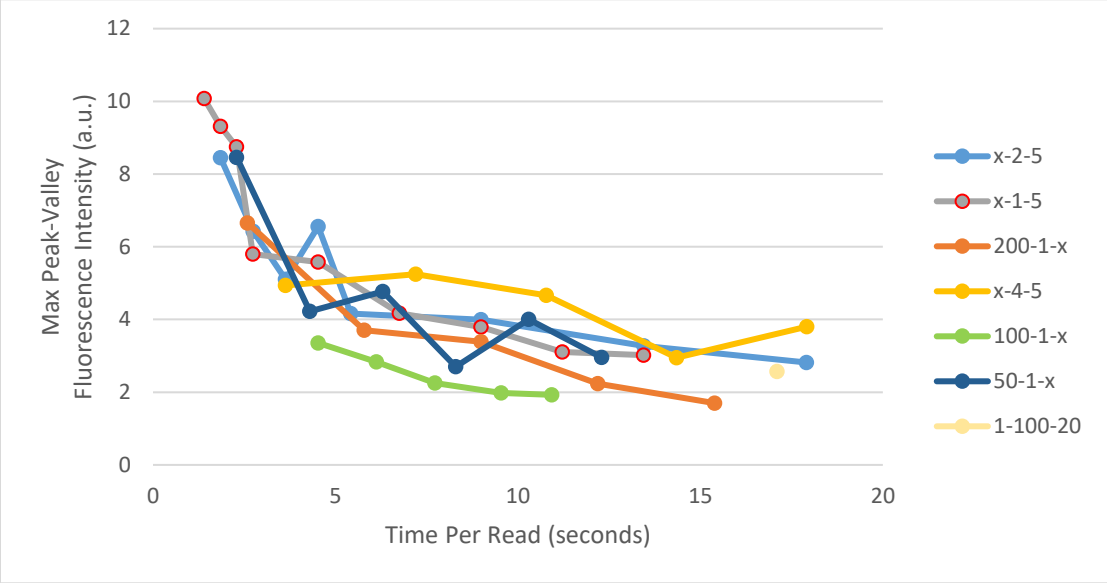

Figure 4.1: Max Peak-Valley Comparison Graph. Comparing seven series with each other over the variable of Time Per Read. Six series have multiple datapoints.

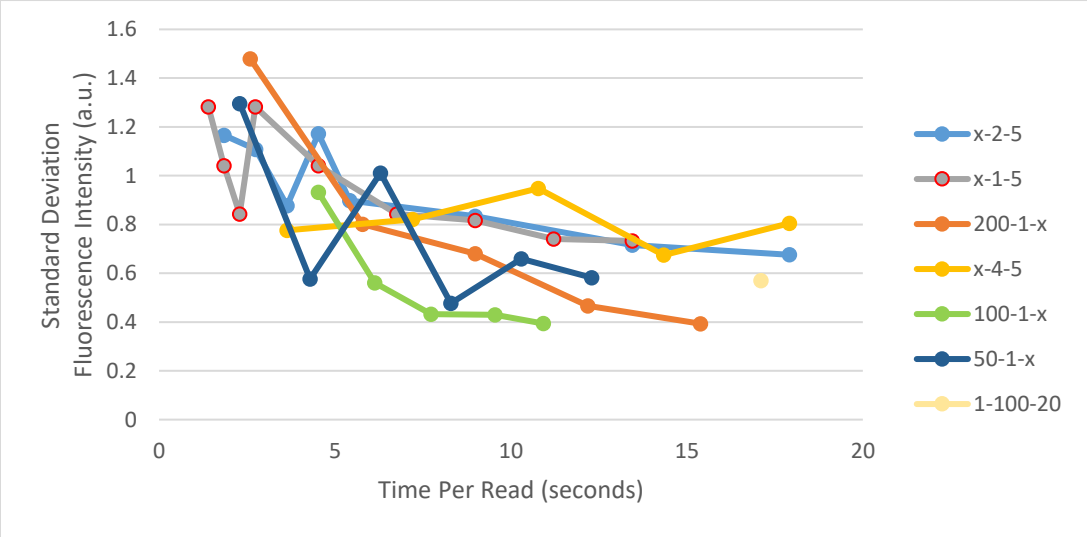

Figure 4.2: Standard Deviation Comparison Graph. Comparing seven series with each other over the variable of Time Per Read. Six series have multiple datapoints.

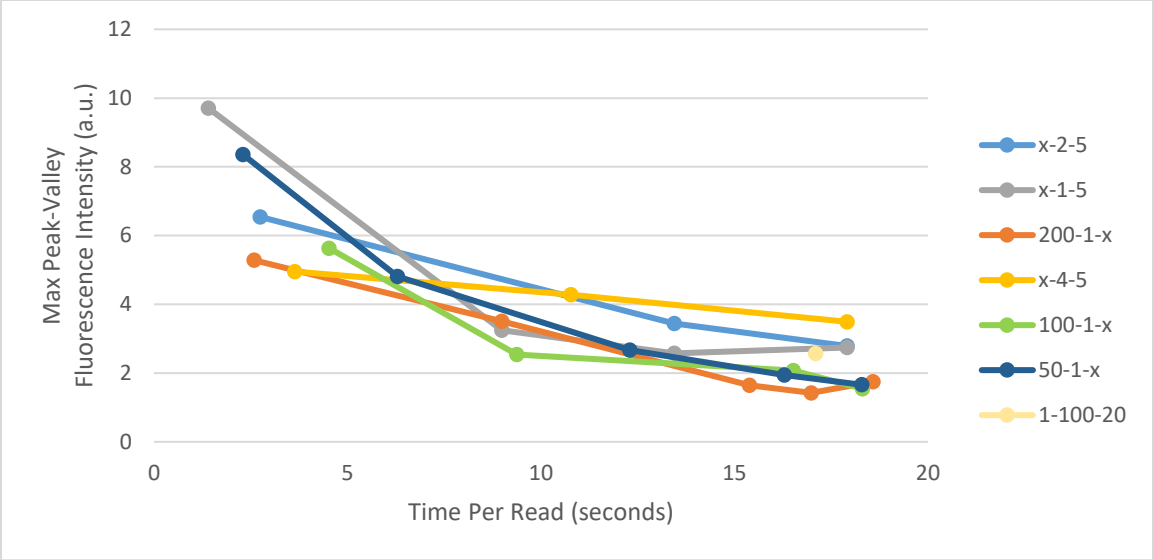

Figure 4.3: Averaged Max Peak-Valley Comparison Graph. Comparing seven series with each other over the variable of Time Per Read. The series have less datapoints, but each are averaged up to four times.

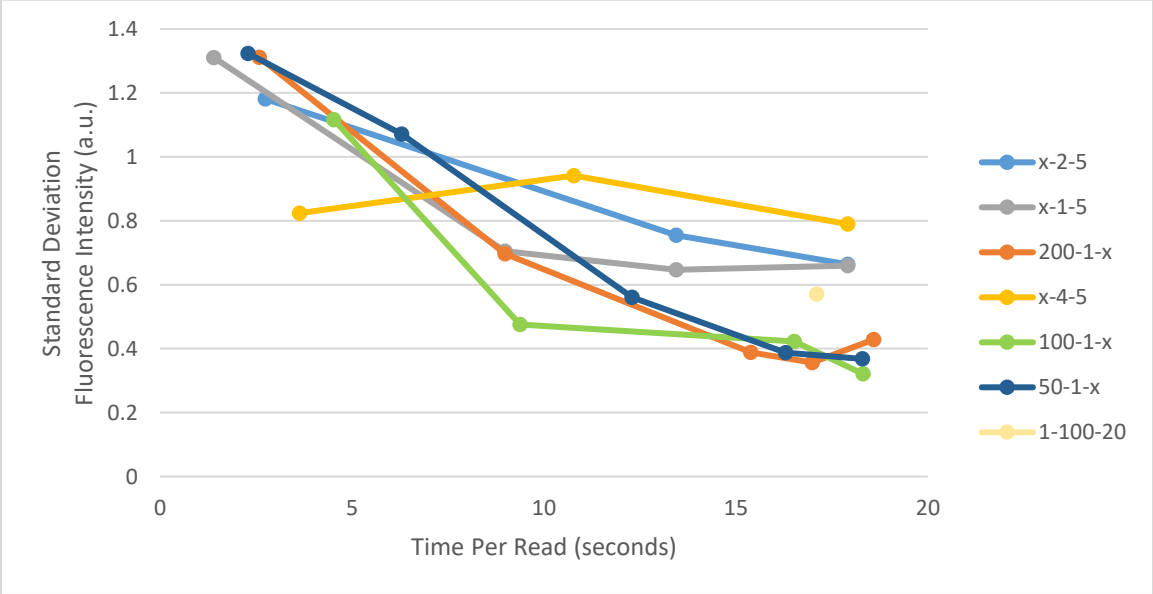

Figure 4.4: Averaged Standard Deviation Graph. Comparing seven series with each other over the variable of Time Per Read. The series have less datapoints, but each are averaged up to four times.

The first pair of max peak-valley and STD graphs (figure 4.1 and 4.2) are taken with more datapoints to give a more detailed look. The second (figure 4.3 and 4.4) pair are taken with less datapoints, but most of those datapoints are values averaging from up to 4 different tests. This includes the data from the first pair. Some additional datapoints with longer time per read values are also added to the second pair to make each line

more comparable (therefore, some of the later datapoints averaged over fewer tests). Both pairs of graphs produced similar results.

From the general shape of all the lines, as the time per read increases, the fluorescence intensity decreases. However, as mentioned in sections before, if the time between each read is too long, the device may be slow to respond to inputs, such as the heater's need for a state change or the user's press of a button. Also, with the current implementation of the GUI, it increases the chance of a crash. Therefore, a maximum allowed time per read was set at 18 seconds. (The reason why 18 seconds was chosen was mainly due to what felt acceptable during the testing process.)

With that limit in mind and taking a look at the various max peak-valley and STD lines, increasing the delay has the most impact on the data. Furthermore, from the lines 50-1-x, 100-1-x, and 200-1-x, which differ only in the count constant, the actual choice in the value of count decreases in importance as the time per read increases. The flatness of the x-1-5 line also supports this conclusion.

But in general, from the lines available on the graph, the 100-1-x line yielded the best result, with the configuration of 100-1-22 being the one to produce the lowest fluorescence intensity fluctuations.

4.2. Signal-to-Noise Ratio Test

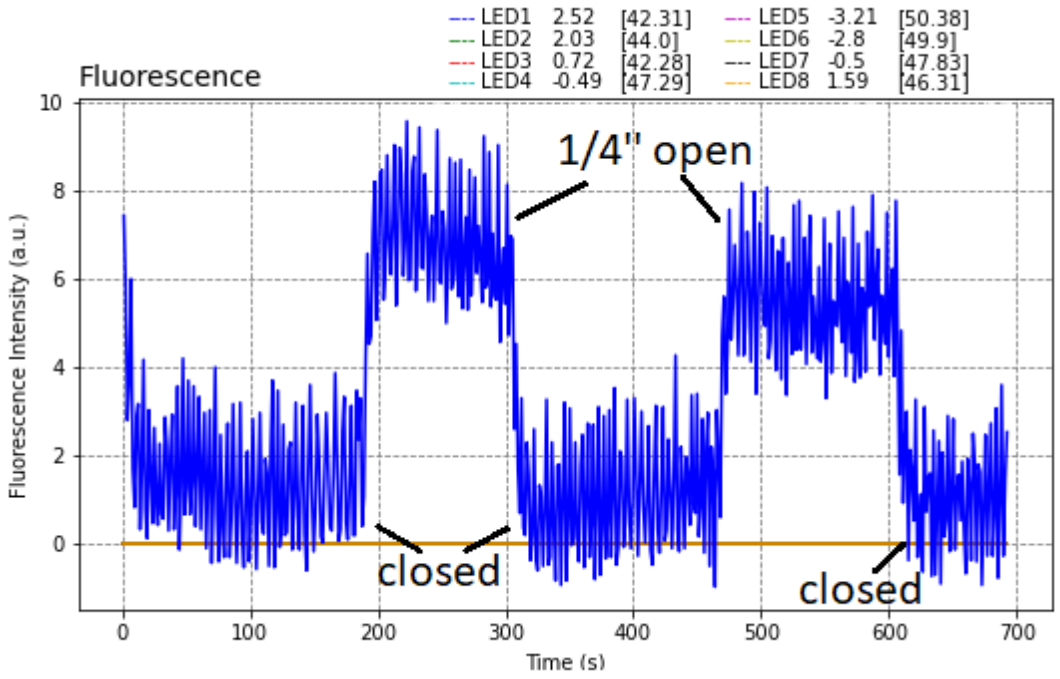

Figure 4.5: Signal-to-Noise Ratio Test for 30-1-5. This configuration has one of the worst max peak-valley and STD values, and this translates to how noisy the graph is.

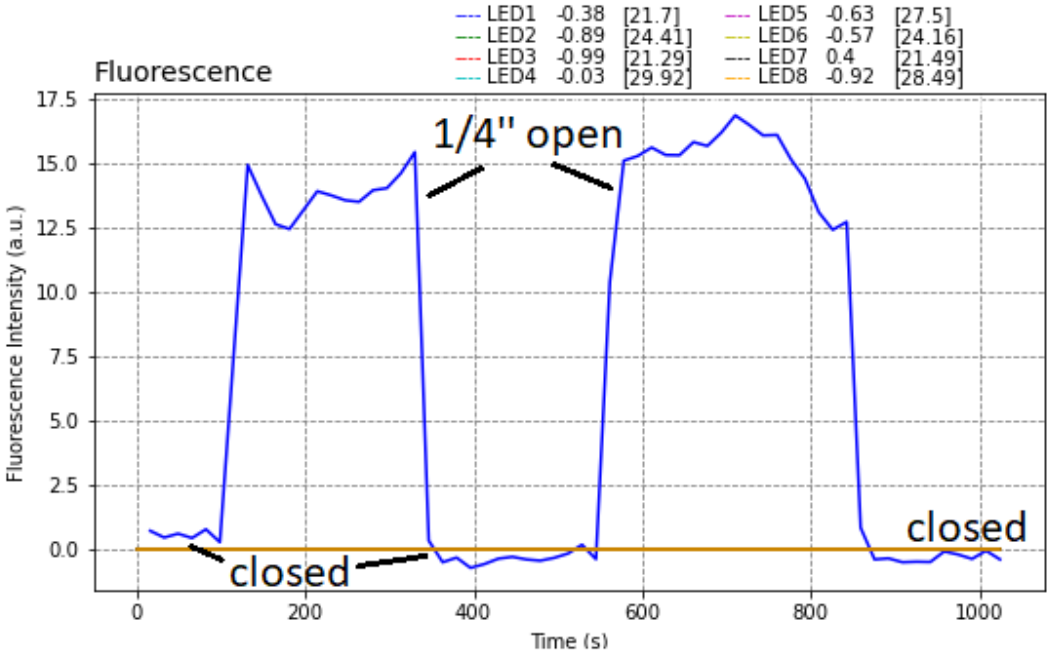

Figure 4.6: Signal-to-Noise Ratio Test for 100-1-20. This configuration has one of the best max peak-valley and STD values and is significantly less noisy than figure 4.5.

Figure 4.5 and 4.6 shows the signal-to-noise ratio test for the configuration of 30-1-5 and 100-1-20, respectively. 30-1-5 and 100-1-20 were selected because one had one of the worst max peak-valley and STD graphs, while the other had one of the best. Comparing the two shows the resulting improvement made to the system by the testing of different configurations.

The tests were done by opening the lid of the 3D printed enclosure by ¼'', to let ambient light into the system, and detecting the resulting change in fluorescence intensity. The areas where the enclosure was closed and where it was opened are labelled. Only one LED (LED1) was used in this test so that the result will be easier to observe.

Even though in both cases, the ¼'' difference could be observed, the 100-1-20 case seem a lot less noisy overall. This shows that changing the configurations did, indeed, improve the signal-to-noise ratio by quite a bit.

Figure 4.7 also shows the light sensitivity of the 100-1-20 configuration. It shows that even with the lid 1/8'' open, it's clearly differentiable.

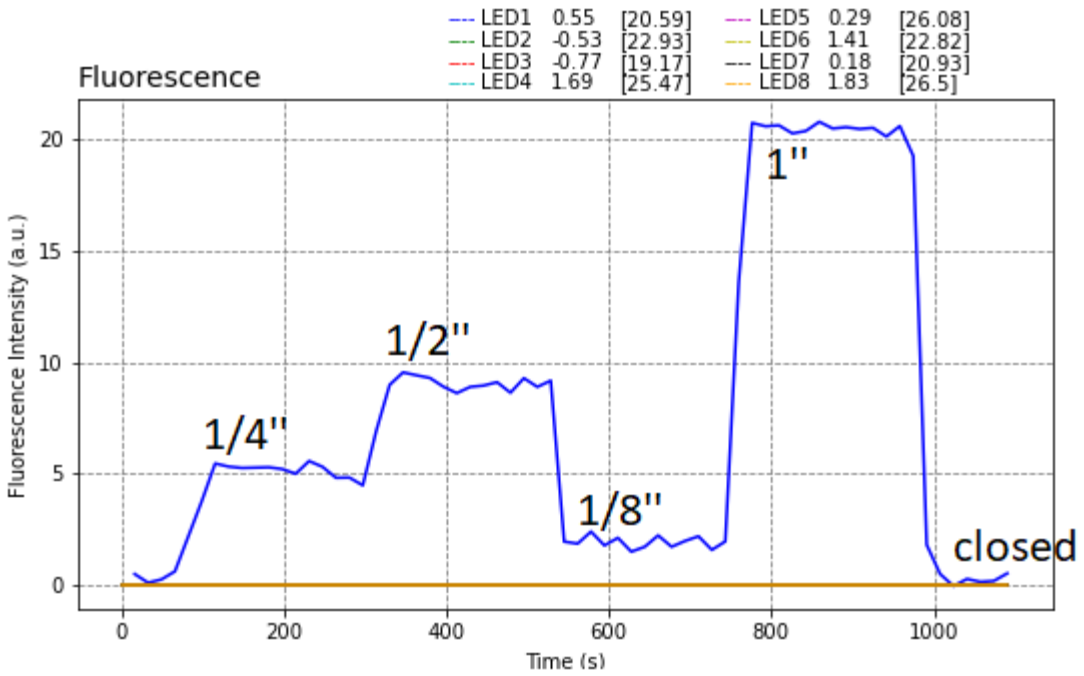

Figure 4.7: Light Sensitivity test for 100-1-20. Shows that 1/8'' opening of the lid can still be differentiated as a signal rather than plain noise.

4.3. Discussion on Cycle

The cycle variable was not singled out on any of the lines, but it can be seen from comparing lines x-1-5, x-2-5, and x-4-5 that an increase in cycle had an adverse effect on the max peak-valley and standard deviation graphs, especially as the time per read increases.

This is a surprising observation, because count and cycle could be seen as counterparts to one another; both variables determine the amount of data taken, with no alteration except for the order at which they were taken. Theoretically, it could be argued that increasing the cycle should provide more stability as the data is averaged across a longer length of time. However, the reason becomes clearer when looking at the figure 4.8.

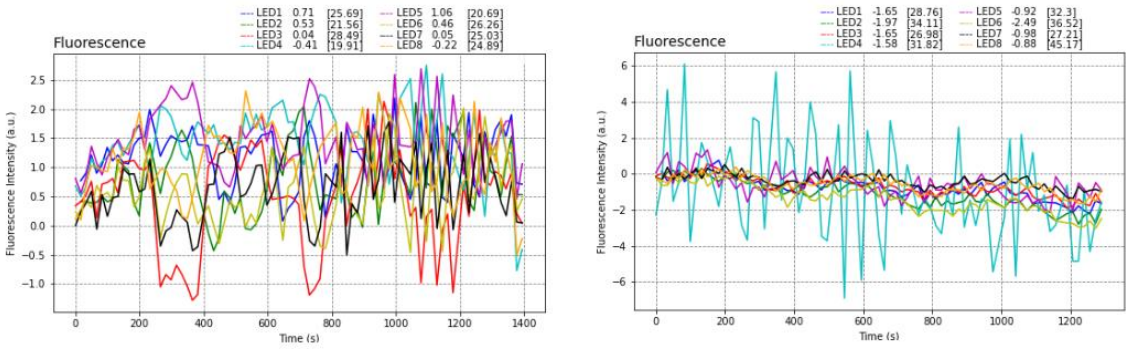

Figure 4.8: Comparison between 1-100-20 (left) and 100-1-20 (right). Shows how the noise can be concentrated on one single LED or be spread out to all the LEDs.

This figure shows that the configuration with higher count have noise that is concentrated on a single LED while the configuration with higher cycle spreads the noise out among all the LEDs.

This means that, even though the datapoint of 100-1-20 appears to be better than that of 1-100-20 in all respects (figures 4.1 to 4.4) it is mainly due to the choices made when parsing through the data. If the stability of every single LED been a major factor in the decision, higher cycle might have been determined as the better choice.

4.4. Discussion on RMS

The RMS graphs corresponding to figures 4.1 to 4.4 are shown below.

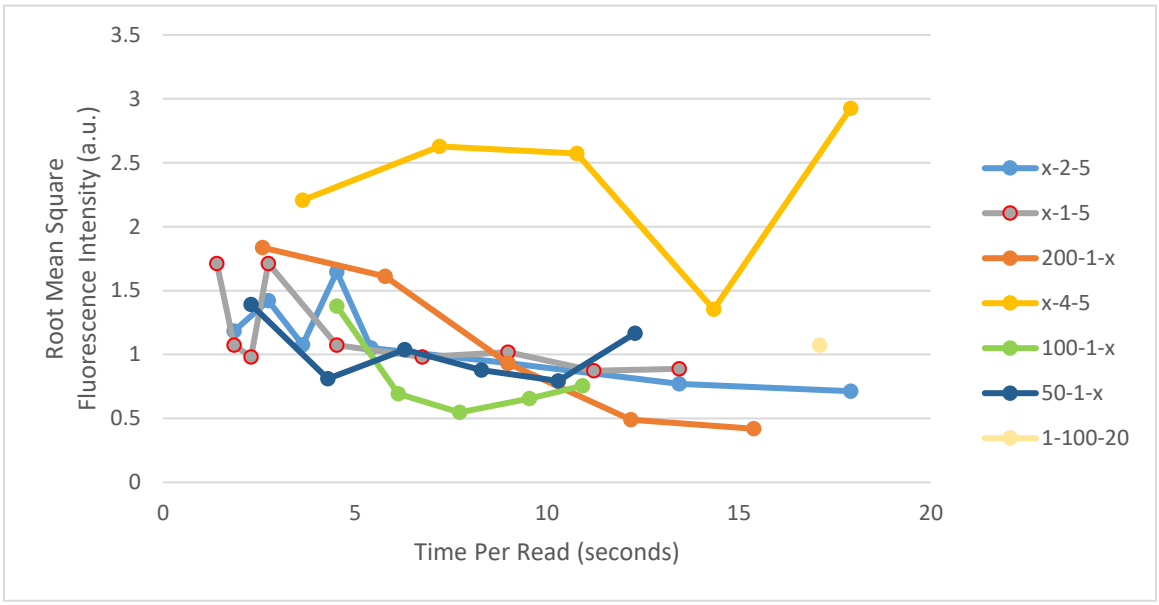

Figure 4.9: RMS Comparison Graph. Comparing seven series with each other over the variable of Time Per Read. Behaves differently from figures 4.1 and 4.2 due to errors in calibration.

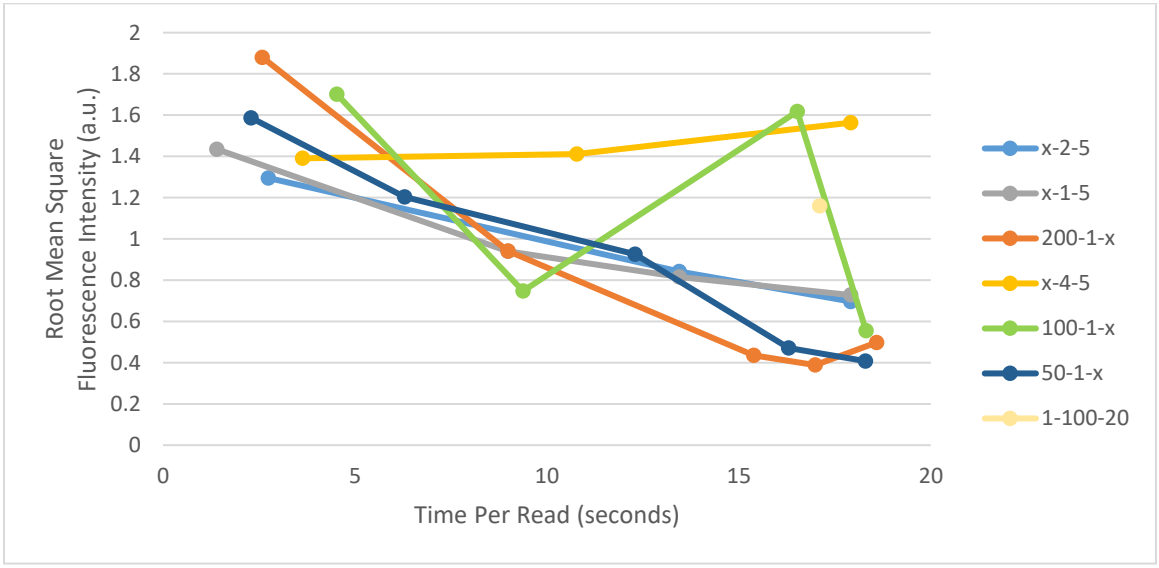

Figure 4.10: Averaged RMS Graph Comparison Graph. Comparing seven series with each other over the variable of Time Per Read. Behaves differently from figures 4.3 and 4.4 due to errors in calibration.

As can be seen in both figure 4.9 and 4.10, the graphs produced are vastly different compared to their respective STD counterparts. This is the expected result because of how the calibration is coded in the system.

Since the calibration is determined from 10 consecutive reads, the calibration can produce a very different result depending on which section of the heater cycle it is averaging over and whether the time per read is short (like 2 seconds) or long (like 18 seconds). Also, given that the time it takes for a full heater cycle is about 400 seconds, it can never properly center at zero unless the calibration is equally long.

The difference between the graphs is further increased due to the possibility of a sloping trendline in the data.

4.5. Discussion on Repeatability of Data Taken

Even though the pair of max peak-valley graphs (figures 4.1 and 4.3) and standard-deviation graphs (figures 4.2 and 4.4) are similar in shape, there are some slight differences between them. This is due to errors embedded in the system itself.

To get an overview of the magnitude of the errors, the maximum and minimum values calculated for each datapoint are plotted as error bars below in figure 4.11 and 4.12.

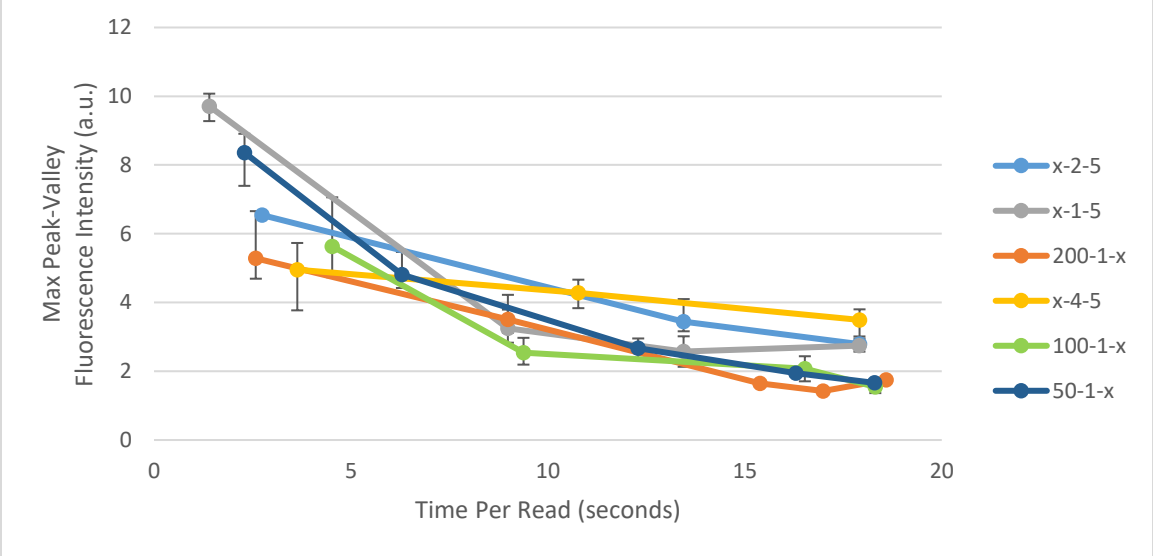

Figure 4.11: Figure 4.3 with Error Bars. Higher Timer Per Read resulted in less error.

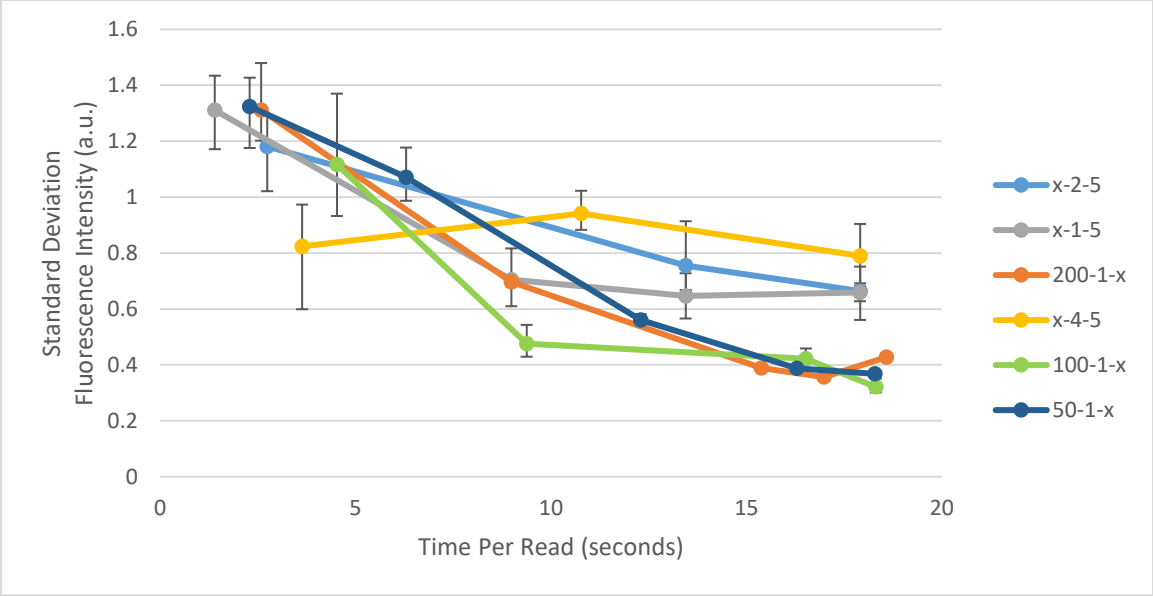

Figure 4.12: Figure 4.4 with Error Bars. Higher Timer Per Read resulted in less error.

To standardize the errors, each point was taken in a different order for each of the 4 tests. But it can still be seen that the magnitude of the error bars is greater in configurations with lower time per reads. The main cause of this should be the same as the reason why longer time per read configurations produced lower fluorescence intensity fluctuations. That reason is that the amount of jitter in the system is decreased due to the data being averaging over a longer period of time.

Because of this, it means that selecting the best configuration also gave the added benefit of selecting the comparatively more stable one.

Chapter 5. Conclusion & Future Work

The purpose of this project was to improve upon the IDD. Such improvements were made in sections dealing with the heater, the Arduino code, and most importantly, the fluorescence intensity error detection. The physical limitations of the system were also investigated.

However, there are still room for improvement for the system in the future.

5.1. Heater

5.1.1. Separation of Heater Circuit

As mentioned in before, the fluorescence intensity and temperature reading graphs are directly impacted by the heater's state. Furthermore, the temperature reading graph shows that there is a spike in the temperature graph whenever the heater's state changes due to the effects of the power supply on the circuit.

Therefore, to decrease the effect that the power supply has on the circuit, it may be better to separate the power supply and heater from the rest of the circuit (LEDs, PDs, and temperature sensors), which will be powered separately.

5.1.2. Refined Heater State Considerations

The temperature reading would reach a maximum a little after the heater is turned off and not right after. The delay in reaching a maximum can be explained by how, even after the heater is turned off, the temperature may keep rising due to the insulation of the enclosure itself. Also, the temperature of the samples themselves may not be the same as the temperature registered by the temperature sensors, especially if the samples are chilled before testing. Therefore, the limits of 40.5 °C and 41.5 °C may need to be reconsidered after further testing.

5.2. Software

5.2.1. Python Program

The main problem with the python program, especially if it'll be used by others in the future, is its knack for freezing and randomly crashing. This is especially true if the time per read is very long, because the program will show as not responding during the entire length of that time.

5.2.2. Calibration

The calibration for the fluorescence intensity can also be improved. As mentioned in the RMS section, the calibration averages the value over a fraction of the heater's cycle, which results in a baseline that is close to zero but not exactly. However, improving this could be tricky, as either the calibration has to take a very long time (the entire heater cycle of 400 seconds, or at least from state changed to state change), or if has to be solved in another way altogether, like dynamically locating the calibration point.

An implementation of calibration for the heater's state change based on time per read should also be considered. As the experiment is now, if time per read is as high as 18 seconds, the heater could be trapped in its current state for up to 18 seconds longer than expected, which may drive it out of the 1-degree limit.

5.2.3. Addition of Software Filters

The consistent cyclical patterning of the fluorescence intensity makes filtering by software a possibility. Even though one state may appear longer than the other in a heater on-off cycle, the pattern could still be learned and applied to future cycles. However, the pattern will need to be dynamically updated in case there's a slope (especially if the slope isn't constant). This may not be possible without drastic changes, especially with how much the current system already freezes and crashes.

5.3. Hardware

The printed circuit board (PCB) may need to implement some additional safeguards as a PCB was destroyed in the process of this project. Also, further investigation could be done to mitigate, or filter, the spikes found when there is a heater state change.

References

[1] Hicks, Trevor, MEng Project Report. (2020). Project Title: Electronic design and assembly of an Isothermal Diagnostic Detector. Published by SFU Library.

[2] Kaur, Amandeep, MEng Project Report. (2020). Project Title: Design of hardware components for fluorescence-based Isothermal Diagnostic Detector. Published by SFU Library.

[3] A. Abdolahzadeh, E.V.Dolgosheina, and P.J. Unrau, “RNA detection with high specificity and sensitivity using nested fluorogenic Mango NASBA,” *RNA,* vol. 25, no. 12, pp.1806-1813, Sept. 2019