MICRO-SCALE LASER-INDUCED FLUORESCENCE THERMOMETRY FOR MULTIPHASE FLOW IN POROUS MEDIA

A Thesis

Presented to

The Faculty of the Department of Mechanical Engineering

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Samuel J. Simmons

December 2023

The Designated Thesis Committee Approves the Thesis Titled

MICRO-SCALE LASER-INDUCED FLUORESCENCE THERMOMETRY FOR MULTIPHASE FLOW IN POROUS MEDIA

by

Samuel J. Simmons

APPROVED FOR THE DEPARTMENT OF MECHANICAL ENGINEERING

SAN JOSÉ STATE UNIVERSITY

December 2023

Farzan Kazemifar, Ph.D.	Department of Mechanical Engineering
Sang-Joon (John) Lee, Ph.D.	Department of Mechanical Engineering
Crystal M. Han, Ph.D.	Department of Mechanical Engineering

ABSTRACT

MICRO-SCALE LASER-INDUCED FLUORESCENCE THERMOMETRY FOR MULTIPHASE FLOW IN POROUS MEDIA

by Samuel J. Simmons

In this thesis the use of Laser-Induced Fluorescence (LIF) thermometry was evaluated as a temperature measurement for multiphase flow in porous media. This optical temperature measurement technique utilizes the temperature-dependent emissive properties of fluorescent dyes to measure temperature. This research evaluates the accuracy, spatial resolution, and temporal resolution of LIF thermometry compared to existing temperature measurements. In this research water-soluble and oil-soluble fluorescent dyes are evaluated in terms of their temperature sensitivity. The ability of these dyes to measure temperature is compared to an Ansys FEA simulation of a fixed temperature gradient. For multiphase flow, the fluorescent dyes were both simultaneously measured within the porous media micromodel and distinguished by the cameras based on their differing fluorescence response for two cameras. The results of this research determined that the technique was able to produce a spatial resolution of $(5.2\ \mu m)^2$, a temporal resolution allowing for a complete measurement to be taken in 4.5 seconds, and a root-mean-square error of at most 4.30 °C and at beast a root-mean-square error of 1.62 °C. While the spatial resolution of the technique is much greater than most existing temperature measurements, the accuracy of the temperature measurement must be greatly improved for this technique to be practical.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

1.1 Literature Review

Microfluidic devices are commonly used in biology, chemistry, physical sciences, and medicine to handle volumes of fluids ranging from picolitres to microliters [1]. Understanding of surface tension, capillary force, laminar flow, and viscous drag are critical to studying the mechanics of microflow within these devices. Despite the difficulty of the physics of microflows, microfluidics research has rapidly advanced in recent years [1]. The use of microfluidics devices to study porous media is a common application of microfluidics. Porous media micromodels have been instrumental to advances in the studies of single and multiphase phenomena in porous media applications, including: enhanced oil recovery, colloidal transport, microorganism transport, and fuel cells [2].

Measurement of temperatures within these porous media micromodels is difficult with the temperature measurement techniques that are commonly used currently, such as thermocouples, infrared thermometers, and thermochromic liquid crystal probes. One alternative to these more common temperature measurement techniques is laser-induced fluorescence (LIF) thermometry, an optical temperature measurement technique that relies on the temperature-dependent fluorescence response of many fluorescent dyes [3]. While there is a wealth of literature available on measuring temperatures in micro-scale fluid flows, very little literature exists on the measurement of temperatures in multiphase flows through porous media. No literature covering the simultaneous measurement of the

temperature within two immiscible fluids (an aqueous fluid and a non-aqueous fluid for this research) was able to be found.

This thesis will utilize LIF thermometry to measure temperature distributions within multiphase flows through porous media in different heat transfer scenarios. Planar temperature fields within porous media micromodels will be determined experimentally and compared with simulations to determine the validity of applying LIF thermometry to multiphase flow in porous media. The results of this research are intended to allow for future investigation of the heat transfer processes within porous media, which is applicable to the fields of mechanical engineering, chemical engineering, environmental engineering, petroleum engineering, and geology [4]. For example, LIF thermometry could be used for temperature measurements when designing microchannel heat sinks that use sintered porous media for electronics cooling applications [5].

1.1.1 Existing Temperature Measurement Methods

Thermocouple probes are used in many heat transfer applications as a reliable method of temperature measurement due to their low cost, simplicity, and relatively small size for most applications [3]. However, their applicability to measuring temperature in micro-scale flows is limited by their limited spatial resolution and physical intrusiveness to the flow [6]. Even the smallest micro-thermocouples currently sold are between 13 and 50 micrometers, far too large for many micro-scale applications [7]. Thermocouples typically have response times less than 1 second depending on many factors and standard Type T Thermocouples

typically have an accuracy of ±1 °C or 0.75% of measurement (whichever is greater) [8]–[10].

Another temperature measurement technique involves the usage of thermochromic liquid crystal probes to measure temperature, utilizing the selective reflection of visible light due to the local temperature [6]. Temperature measurement using thermochromic liquid crystal probes requires point-by-point calibration of any test field due to the effects of illuminating light intensity variation within the test field. Additionally, each microencapsulated thermochromic liquid crystal bead has an approximate size greater or equal to 10 micrometers, much too large for high-resolution temperature measurement in microscale applications [6].

Planar temperature fields can be determined using infrared thermography, a technique that utilizes the emission of thermal radiation to measure temperature [3]. While this technique features an immediate response capable of achieving high surface temperature spatial resolutions, this technique is unfortunately limited to outside surfaces, making it ill-suited for measuring temperatures within porous media [3].

1.1.2 Single-Dye LIF Thermometry Principles

In LIF thermometry, excitation of temperature-sensitive fluorophores found in fluorescent dyes by specific wavelengths of photons causes the fluorophores to emit photons at another wavelength [7]. The wavelengths being emitted are then optically filtered to measure the fluorescence intensity within a specific range of wavelengths. The intensity of light being emitted can then be correlated with temperature for a temperature-

sensitive fluorescent dye, allowing for temperatures to be determined based on the fluorescence intensity measured [7]. Quantum efficiency, a characteristic that is different for every fluorescent dye, may be either temperature-dependent or temperature-independent depending on the fluorescent dye used [11].

This emerging measurement technique has spatial resolutions up to 0.1 to 0.3 micrometers and thermal resolutions up to 0.01 K, with typical accuracies of ±1.5 K over ranges of at least 40 K [7], [11]. The fluorescence intensity is essentially only dependent on temperature provided the excitation light intensity and dye concentration remain constant [6]. Additional problems arise due to dye molecules absorbing or scattering the laser light, thus decreasing the laser sheet intensity even with an ideal setup. The cameras used offer yet another source of error, with optical limitations and uncertainties in their responses producing inconsistent intensity levels between tests [6]. Another source of error is the inability of the single-dye technique to distinguish between intensity variations due to temperature variations and local concentration variations [12].

1.1.3 Application of Single-Dye LIF Thermometry to Porous Media and Micro-Scale Configurations

Single-dye LIF thermometry has been applied to porous media in optimizing the evaporative performance of a biporous copper wick, like those commonly utilized in two-phase cooling devices [7]. Submicron level evaporation rates were evaluated utilizing Rhodamine B as the fluorescent dye with a spatial resolution of approximately 0.3 micrometers. To calculate the evaporation rate, the surface fluorescence intensity was

evaluated for a heated solution in a steady-state and converted into a surface temperature profile. Based on this surface temperature profile, the local evaporation rate could be calculated [7].

Regarding micro-scale applications, single-dye LIF thermometry using Rhodamine B has been applied to measure temperatures within acrylic microchannels ranging from room temperature to 90°C [13]. Each microchannel had a trapezoidal cross-section, with nominal dimensions of 30 microns deep, 20 microns wide at the bottom of the channel, and 75 microns wide at the top of the channel. This technique produced a measurement uncertainty between 2.4 and 3.5°C with spatial and temporal resolutions of 1 μm and 33 ms, respectively [13].

1.1.4 Ratiometric LIF Thermometry Principles

The ratiometric (or two-color) LIF thermometry technique uses a temperature-sensitive and a temperature-insensitive fluorescent dye to compensate for uncertainties introduced in the single-dye LIF thermometry by variations in excitation light illumination [14]. A temperature-insensitive dye is used together with a temperature-sensitive dye to compensate for fluorescence intensity variations in the temperature-sensitive dye caused by incident irradiation variation. The fluorescence intensity ratio between the fluorescent dyes is essentially dependent on temperature assuming a constant dye concentration [14].

Both the two-color and single-dye LIF thermometry techniques suffer from photobleaching effects in which the fluorescent response degrades as the fluorophores are repeatedly excited [15]. For example, eosin Y is a fluorescent dye that exhibits “significant

photo-dissociation effects under pulsed laser illumination", making it likely a poor choice of fluorescent dye as the laser being used is pulsed [16]. This photo-dissociation means that the eosin Y molecules are broken down by the photons created by the laser. This effect, commonly referred to as photobleaching, leads to fluorescence emission decreasing over time during repeated laser exposure. This can be seen in Figure 1, with the fluorescent emission of eosin Y rapidly decreasing within just a minute of exposure leading to a 30% reduction in emission intensity when dissolved in water. This is less of a problem when eosin Y is dissolved in ethanol, as emission intensity decreased about 10% after 20 minutes of exposure. While photobleaching is a problem with all fluorescent dyes, eosin Y is the only dye used for this research where photobleaching was mentioned as being a particular large issue.

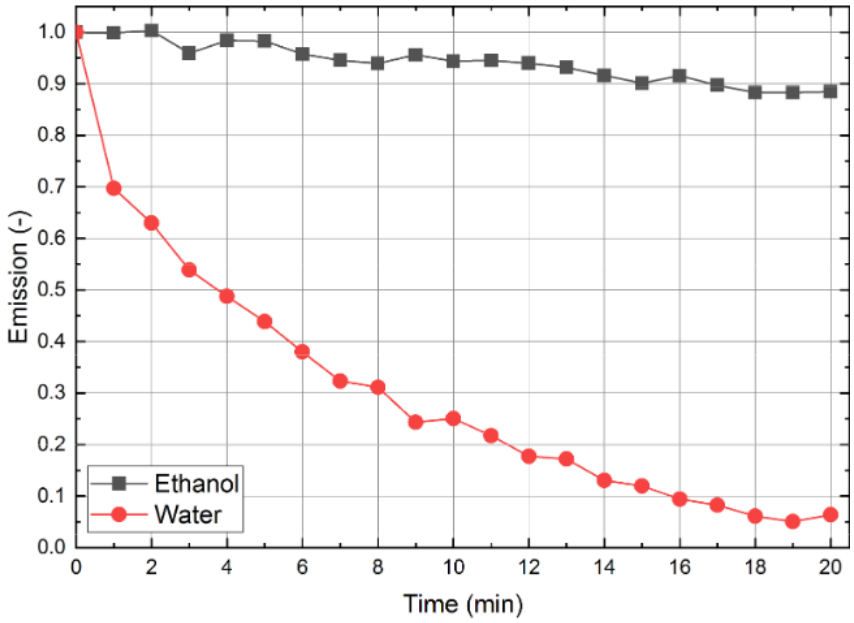

Figure 1. Fluorescence emission versus time for eosin Y (0.01586 g/L) dissolved in ethanol and water at 20 °C using a continuously illuminated 10 Hz, 45.8 mJ/cm^2 laser [16].

1.1.5 Application of Ratiometric LIF Thermometry to Multiphase Flows, Porous Media, and Micro-Scale Configurations

The two-color LIF thermometry technique has been utilized to study heat transfer due to a constant, linear temperature gradient applied to a polydimethylsiloxane (PDMS) device consisting of seven parallel microchannels [17]. A hot and a cold reservoir, with temperatures maintained by circulation water baths, were placed 2 millimeters apart and the PDMS device was placed between the two reservoirs. Mean steady-state temperatures were measured to be within ±0.4°C and ±0.3°C of their predicted temperature in water and ethanol using Rhodamine B and Sulforhodamine-101 as the temperature-sensitive and temperature-insensitive dyes, respectively. This technique was shown to have a spatial resolution of 22.2 x 22.2 micrometers with experimental uncertainties of between ±0.48 and ±0.59°C for ethanol and between ±0.41 and ±0.49°C for water. Photobleaching effects were mitigated by continuously circulating the dye solution [17].

In regards to LIF thermometry applied to multiphase flows, a research paper by Labergue et al. investigated the thermal mixing of two water sprays at different temperatures using the technique [18]. A three-color LIF thermometry technique was used, with the additional color used to also compensate for the effects of droplet size, relating temperatures with two fluorescence ratios due to a quadratic relationship between the temperatures and the fluorescence ratios. Various spray flow rates were tested along with different spray heating conditions and it was found that this technique could successfully determine the evolution of droplet temperature as a function of droplet diameter and

distance from the sprays [18]. No available literature was found applying ratiometric LIF thermometry to porous media.

1.1.6 Significance

Multiphase flow through porous media is found in many practical applications, such as oil recovery from petroleum reservoirs and natural gas recovery [19]. While single-phase fluid flow through porous media can be characterized by Darcy's law, multiphase flow modeling has proved challenging. In industrial applications, an approximation of Darcy's law is utilized that fails to model the true complexity of multiphase flow through porous media [19]. LIF thermometry could be applied to multiphase flow in porous media to assist in the experimental characterization of the temperature distributions through these flows, allowing for engineers to further improve their designs in these applications.

1.2 Hypothesis

The hypothesis of this investigation is that laser-induced fluorescence thermometry can achieve similar temporal resolution and accuracy to thermocouples with a significantly greater spatial resolution for simultaneous measurement of temperature in two immiscible liquid phases in microfabricated 2D porous models with. Multiple fluorescent dyes are to be tested in multiphase flow configurations with fabricated porous media micromodels to demonstrate the high temperature sensitivity, wide temperature range, and high spatial and temporal resolution of this technique for pore-scale investigation of heat transfer within porous media. The final fluorescent dyes selected should be capable of accurately capturing the temperature distributions within the porous media micromodels for a

temperature range between 20 °C to 65 °C, have a temporal resolution capable of producing one measurement every 4.5 seconds, and spatial resolution allowing for measurements every 5.2 µm by 5.2 µm region within the devices.

2 METHODOLOGIES

2.1 Experimental Apparatus

The experimental apparatus consists of a microscope, cameras, and lase for imaging. A dual-head 2 mJ 532-nm EverGreen neodymium-doped yttrium aluminum garnet (Nd:YAG) laser was used as the illumination source. Two Andor Zyla Scientific Complementary Metal-Oxide-Semiconductor (sCMOS) cameras with a 2560 x 2160-pixel sensor (5.5 megapixel) and 6.5 µm pixel size were used to record images. A 10X 0.25 Numerical Aperture objective was used (0.65 µm/pixel). Optical filters were used such that one camera captured fluorescence from 547.5-572.5 nm (Chroma ET560/25x filter) and the other camera captured 610-640 nm light (Chroma ET625/30m filter). Figure 2 shows an example of the fluorescence emission of two fluorescent dyes and the range of wavelengths captured by each camera. In this example, camera 1 (green) mostly captures the fluorescence emission of eosin Y and camera 2 (red) mostly captures the fluorescence emission of sulforhodamine 101. Additionally, thermocouples were used for reference temperature measurements, water baths were used to set the temperature of the porous media device, and syringe pumps were used to pump the fluorescent dye through the device to avoid the effects of photobleaching.

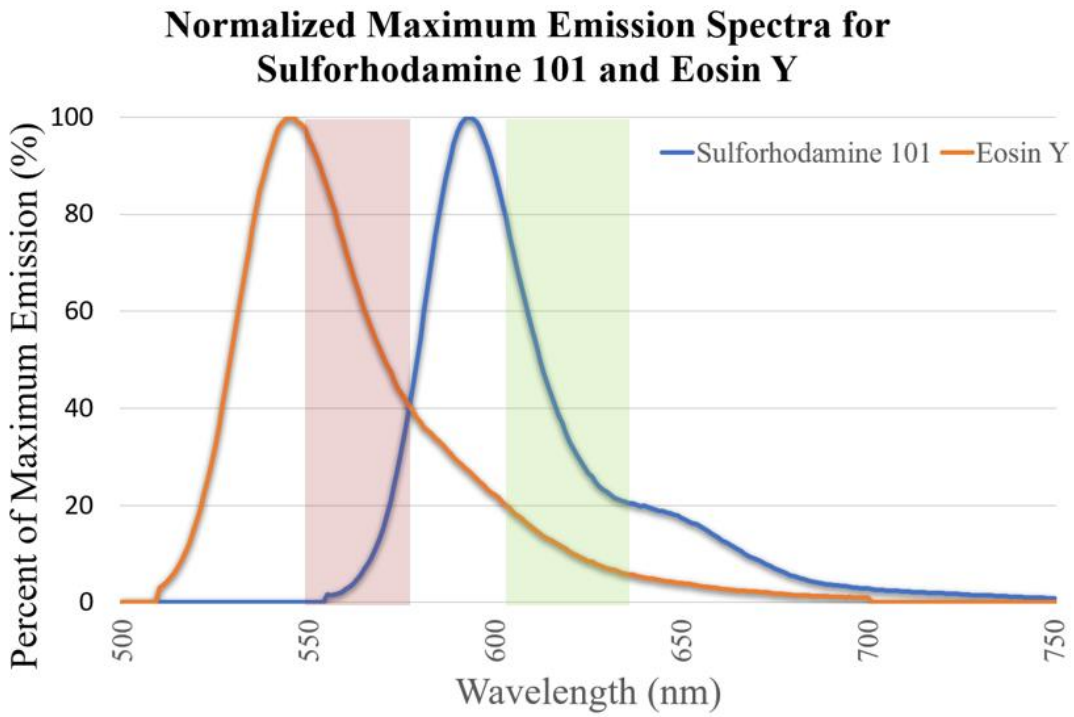

Figure 2. Plot of the normalized maximum emission of sulfrhodamine 101 and eosin Y showing the approximate wavelength band captured by camera 1 (red) and camera 2 (green) [20].

Suitable fluorescent dyes were determined based on their fluorescence emission spectra measurements obtained using a spectrofluorometer. The two sCMOS cameras were used to capture images taken of the fluorescence of various fluorescent dyes at known temperatures. MATLAB codes (shown in **B.1 Data Processing MATLAB Code**) were written to determine the relationship between the fluorescence intensities measured by both cameras and temperature.

A microscope stage was used for this experiment and a diagram of it is shown in Figure 3. The pulses from the Nd:YAG laser were directed through the microscope and into the micromodel. The fluorescence emission from the PDMS micromodel was then captured by

each camera after being split into two separate beams with a dichroic mirror and traveling through the corresponding filters for each camera. The emission was captured by the same microscope lens that the laser pulses travel through rather than being placed above the micromodel as shown in the diagram. Insulation material was placed over the micromodel to reduce the effects of natural convection on the device.

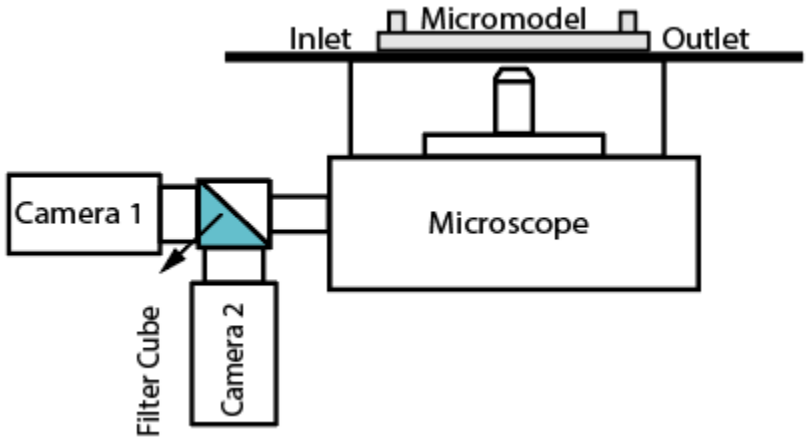

Figure 3. Microscope stage setup used for this research. Nd:YAG laser beam is directed through the microscope lens.

The micromodels used have the approximate dimension shown in Figure 4 and Figure 5, with a 30-micrometer channel depth. Temperature baths were used to maintain the temperature within the PDMS using an aluminum block that surrounds the device. For the temperature calibration process used to determine the temperature sensitivity of the fluorescent dyes (mentioned in 2.3.2 Fluorescent Dye Selection), both temperature baths were maintained at the same temperature and a thermocouple was placed in the outlet port to measure the fluid temperature. Thermal paste as used where the sides of the PDMS

device contact the sides of the aluminum block to allow for conduction. A picture of the experimental setup is shown in Figure 6.

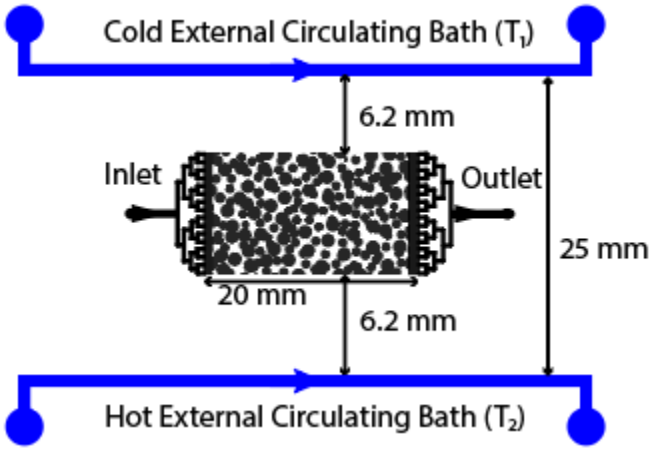

Figure 4. Top view diagram of PDMS micromodel, flow channel dimension widths may vary, with narrower flow channels utilized for temperature calibration.

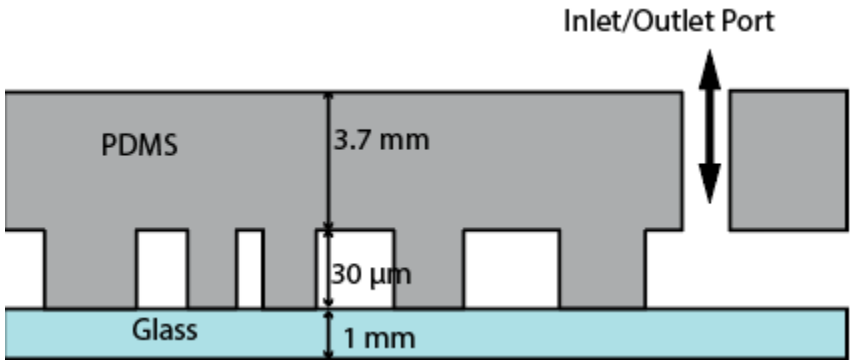

Figure 5. Side view diagram of the PDMS device.

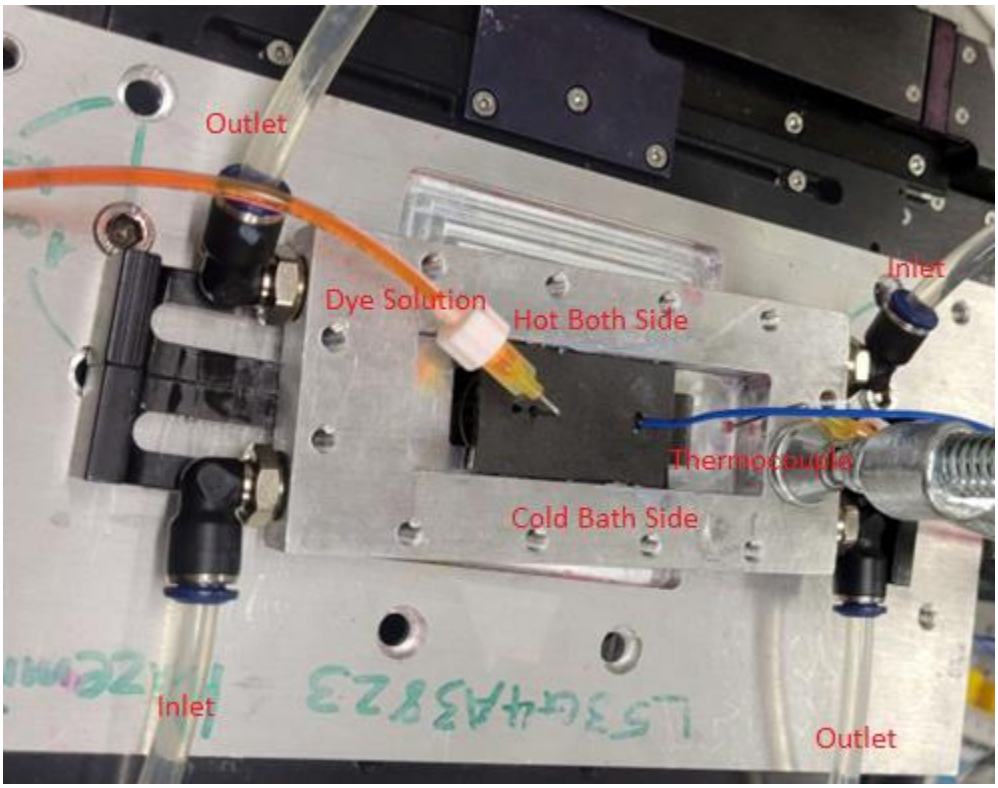

Figure 6. Picture of the experimental setup used during the temperature calibration process.

2.2 Microchannel Design and Fabrication

2.2.1 Flow Device Geometry

Flow device geometries were designed using Adobe Illustrator before the design was placed on a Mylar mask for photolithography. Two flow device geometries were used, a narrow channel device was used for temperature calibration and a wider channel device was used for all other experiments. The narrow channel device is shown in Figure 7. The width of the flow channel inside the device is 3.752 mm. The larger pores of the device have a 300 µm diameter, are spaced out every 79.0 µm, and have a 45-degree angle relative to the diagonally adjacent pores. The smaller pores located on the left and right side of the

flow channel are 150 µm in diameter and spaced out 79.0 µm vertically. The total length of the flow channel is 65.0 mm with a width of 3.72 mm. The overall dimensions of the flow device are 25 mm by 40.0 mm.

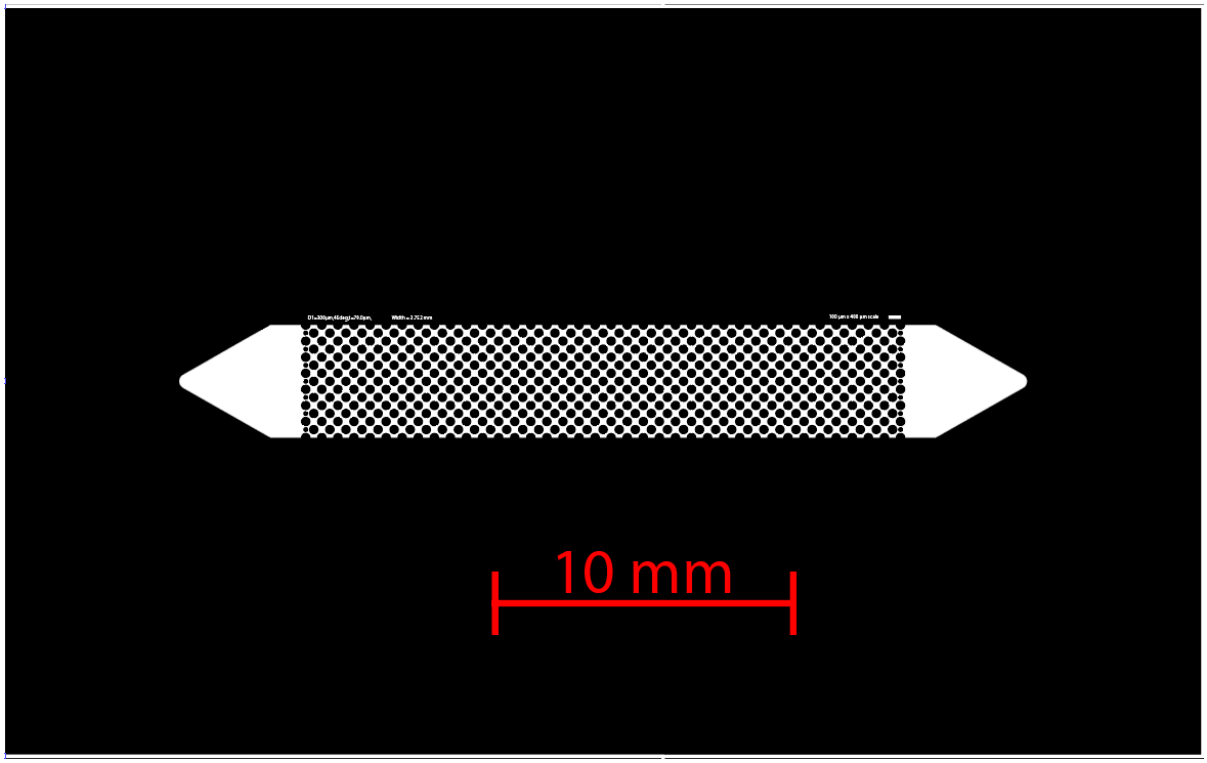

Figure 7. Flow device geometry for narrow channel PDMS device used for temperature calibration.

The geometry of the wide channel flow device is shown below in Figure 8. The overall size, spacing and angle of the pores was mostly the same, but pores have been randomly displaced 60 µm in x or y and scaled to 105% or 95% of the original size. Additionally, triangles were added near the inlet and outlet of the device with a height of 1.5 mm and a base of 1 mm. These triangles were added to ensure that the inlet and outlet of the PDMS device maintained a consistent 30 µm channel height after manufacturing. The channel has

an overall width of 12.5 mm and was made wider to allow for more temperature measurements to be taken along the width of the device and capture more of the temperature variation across the device when each side of the device was set to a different temperature.

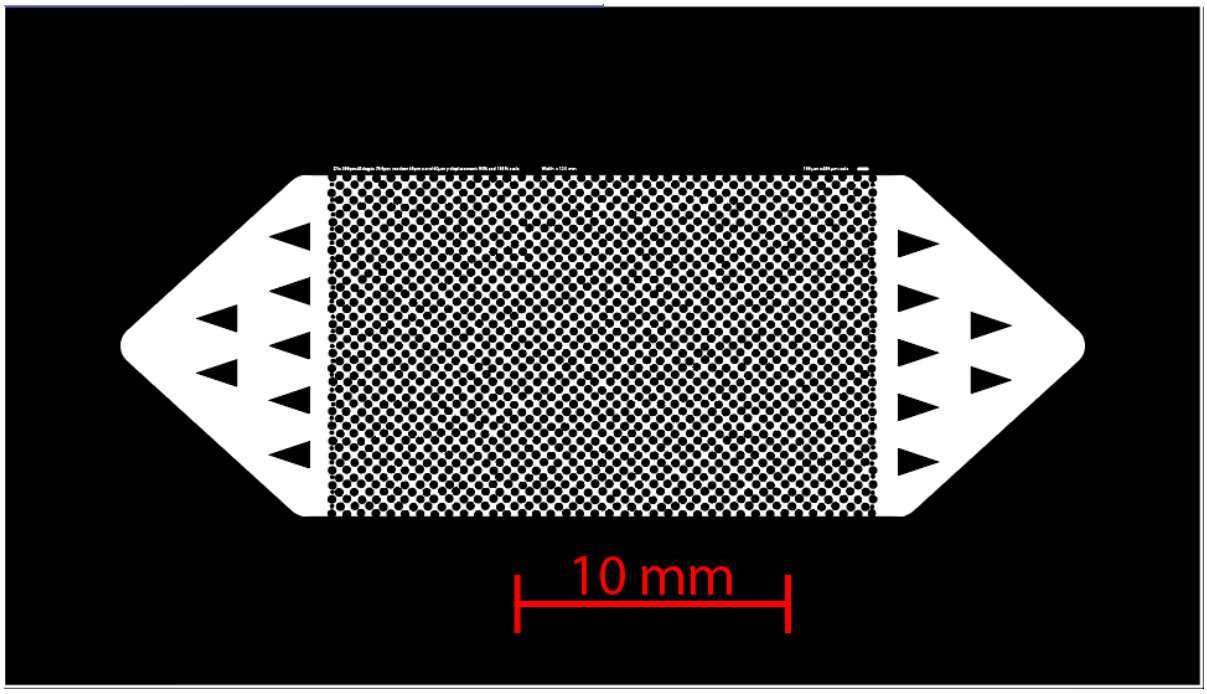

Figure 8. Flow device geometry for wide channel PDMS device used for temperature calibration.

2.2.2 Flow Device Manufacturing

Photolithography was used to manufacture a silicon mold for the flow devices using a Mylar mask. To do this, silicon wafers were coated in SU-8 photoresist, an epoxy-based negative photoresist that is cross-linked by UV exposure [21]. Silicon was spin coated with SU-8 2035 photoresist to produce a 30 µm coating on top. The silicon was then soft baked at 65 °C then 95 °C for 5 minutes each. A mercury lamp was used to expose the SU-8 coating

to UV with the Mylar mask containing the device design placed over the silicon. This UV exposure cross-linked the SU-8 photoresist, transferring the geometry on the mask to the photoresist coating. An SU-8 developer was then spun on the device to remove the parts of the SU-8 coating that were not cross-linked. The silicon wafer was then soft baked at 65 °C then 95 °C for 5 minutes each. Then, SU-8 developer was used again to remove any remaining excess material and isopropyl alcohol was used to wash the device before drying. The silicon wafer was then annealed at 165 °C for 5 minutes, resulting in 30-µm thick SU-8 photoresist mold of the design.

Once the SU-8 mold was manufactured, PDMS was poured over the silicon wafer such that the PDMS was approximately 2 mm above the surface of the wafer. A vacuum pump was used to remove any air bubbles from the PDMS before curing the PDMS in an oven at 100 °C for 1 hour. The PDMS device was then cut out following lines placed around the edge of the geometry, producing a PDMS device that is 25 mm by 40.0 mm with a 30 µm high flow channel on one side of the device. Rapid-core microfluidic punches were used to cut small holes for an inlet and an outlet.

A plasma bonder was then used to bond the PDMS device to a microscope slide. The side of the PDMS device containing the flow channel and one side of the glass microscope slide were exposed to the plasma bonder for 5 minutes at a pressure of roughly 350 mTorr. Once bonded, the PDMS device was annealed for 1 hour at 100 °C using a hot plate.

2.3 Fluorescent Dyes

2.3.1 Fluorescence Spectra of Dyes

Fluorescence spectra were used, when available, to confirm that fluorescence measurements match expectations based on the emission spectra for that dye. For example, based on the emission spectra. For example, based on the emission spectra for eosin Y shown in Figure 9 and the filters being used, eosin Y should have a strong positive temperature sensitive within the 547.5 to 572.5 nm band captured by camera 1. In addition, that fluorescence intensity measured by camera 1 should be significantly higher than that measured by camera 2, as the emission intensity is significantly lower in the 610-640 nm wavelength band.

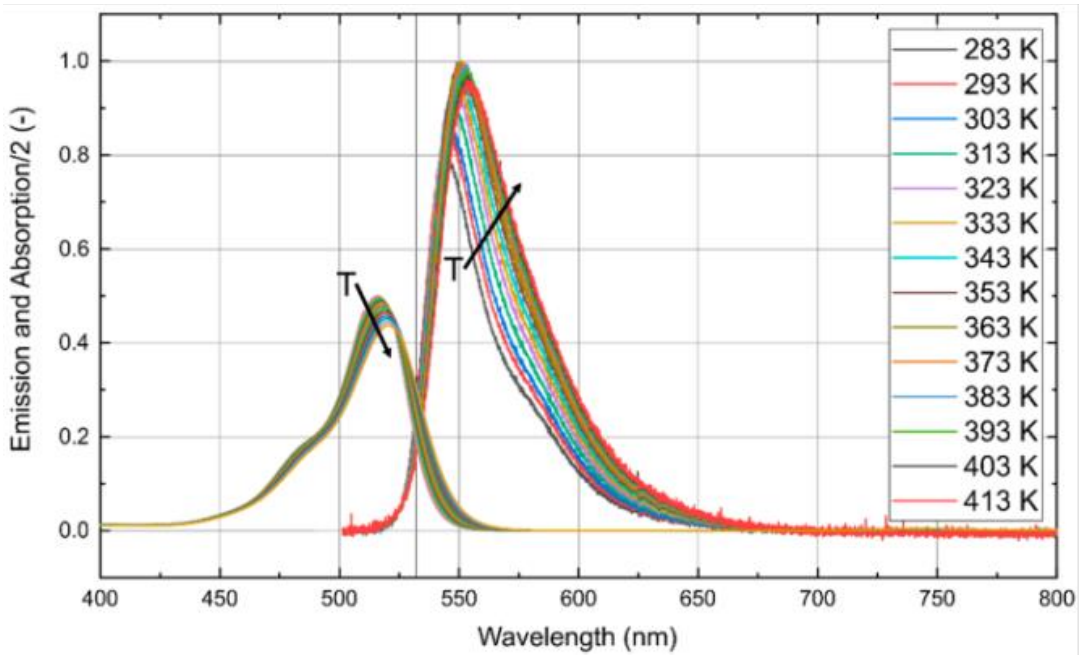

Figure 9. Absorption spectra (left) and emission spectra (right) of eosin Y compared to temperature when dissolved in water [16].

The combined fluorescence spectra for a mixture of pyrromethene 567 and pyrromethene 597-8C9 is plotted below in Figure 10. Fluorescence spectra for these two

fluorescent dyes individually were unable to be found, so individual data on their emission spectra's temperature sensitivity cannot be used for comparison.

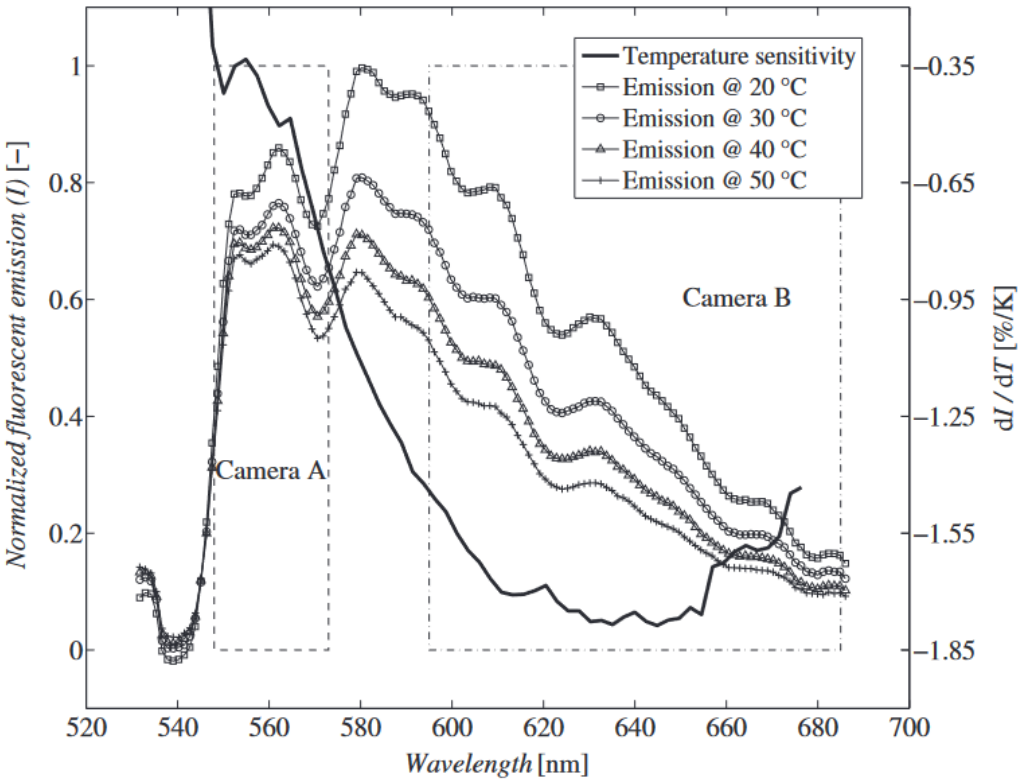

Figure 10. Emission spectra of pyrromethene 567 (0.176 g/L) + pyrromethene 597-8C9 (0.483 g/L) dissolved in hexadecane compared to temperature. Graph shows the emission spectra of both dyes mixed together along with the wavelength bands these researchers selected [22].

The emission spectra of sulforhodamine 101 dissolved in ethanol is plotted below in Figure 11. This emission spectra indicates nearly no emission intensity within the band of wavelengths captured by camera 1 and a negative temperature sensitivity in the 610-640 nm wavelength band captured by camera 2. The emission spectrum for fluorescein disodium salt hydrate is plotted in Figure 12 and indicates a strong positive temperature sensitivity at all wavelengths. The fluorescence of fluorescein disodium salt hydrate is most

intense in the 547.5-562.5 nm wavelength band captured by camera 1. Emission spectra for sulforhodamine B sodium salt and pyrromethene 567 and pyrromethene 597-8C9 individually were unable to be found.

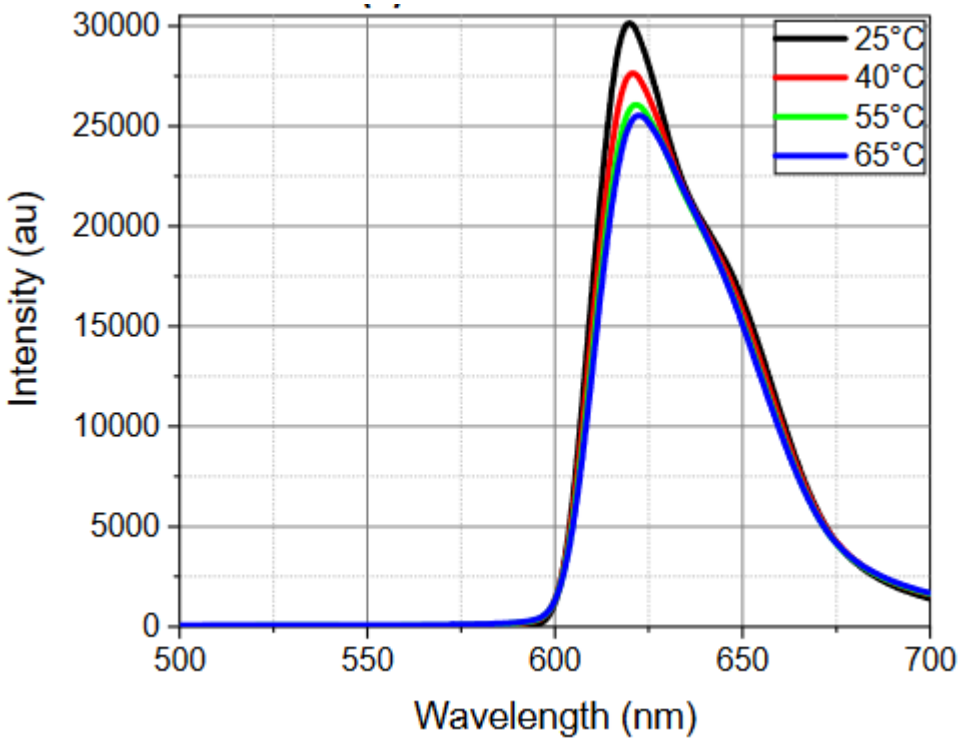

Figure 11. Emission spectra of sulforhodamine 101 compared to temperature for ethanol [23].

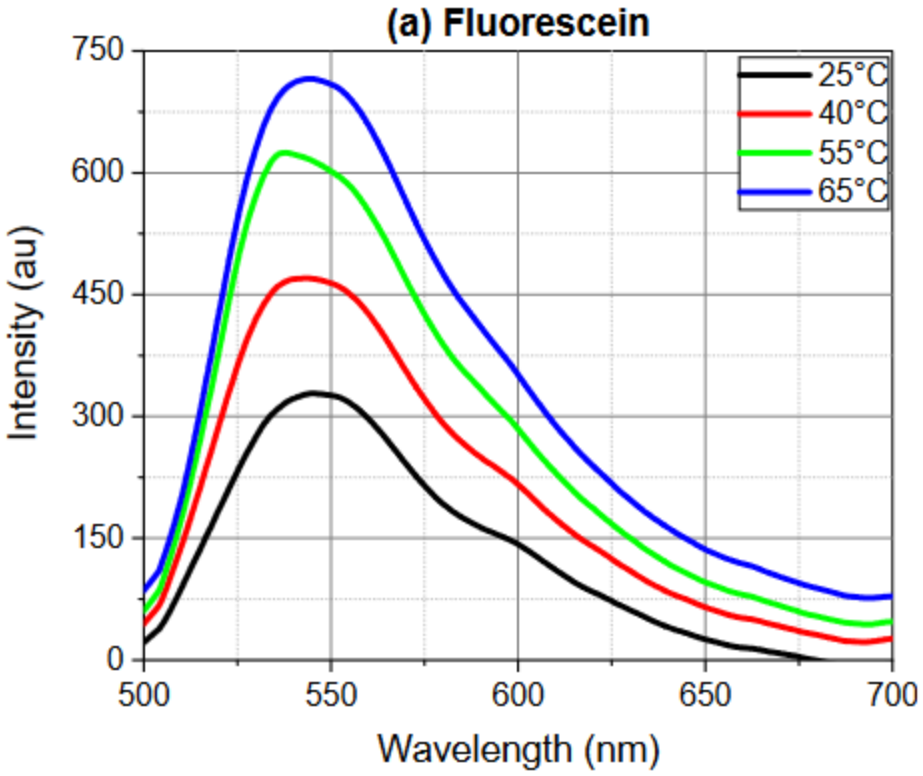

Figure 12. Emission spectra of fluorescein disodium salt hydrate dissolved in ethanol compared to temperature [23].

2.3.2 Fluorescent Dye Selection

Excitation of fluorophore molecules within fluorescent dyes produces a temperature-dependent emission of photons at a wavelength that is different from the wavelength of light that was used to excite it [7]. The relationship between the fluorescence intensity of this emission and temperature can be mathematically represented as

$$I = I_0 C \varepsilon \phi \qquad (1)$$

where I_0 denotes the incident light flux (W m^{-2}), C denotes the concentration of the dye solution (kg m^{-3}), ε denotes an absorption coefficient (m^2 kg^{-1}), ϕ denotes the quantum efficiency of the dye, and l denotes the optical path length (m) [11]. Optical path length will

be kept approximately constant by using microfluidics devices each manufactured to approximately the same depth.

For this equation to be valid for measuring temperature, I_0 and C must be kept constant, as variations in these values can be misinterpreted as changes in fluid temperatures [17]. At large enough dye concentrations, fluorescence emission deviates significantly from this equation due to self-quenching, where part of the emission of the dye is reabsorbed by the dye [24]. The equation above is a linearized version of the relationship between fluorescence intensity and dye concentration and is only valid for sufficiently low dye concentrations [25]. This linearized equation is not valid for flows with local high concentrations, such as those found in turbulent flow regimes [25]. In this research, concentrations were kept low enough that the linearized relationship between fluorescence intensity and concentration could be used.

The incident light flux is never entirely constant, necessitating a way to compensate for uncertainties that are introduced by variations in excitation light intensity [14]. A two-color LIF thermometry technique may be used to account for this variation, which uses a temperature-sensitive and temperature-insensitive dye [2]. The relationship between the fluorescent intensities may be represented mathematically as

$$\frac{I_A}{I_B} = \frac{\varepsilon_A C_A \phi_A}{\varepsilon_B C_B \phi_B} \qquad (2)$$

in which ε, C, and ϕ refer to the absorption coefficient, dye concentration, and quantum efficiency of temperature-sensitive fluorescent dye A and temperature-insensitive

fluorescent dye B, respectively. The absorption coefficient is approximately constant for a given fluorescent dye, while quantum efficiency is a temperature-dependent or temperature-independent quantity depending on the fluorescent dye. The fluorescence ratio has no dependence on incident light intensity but each dye's emission must now be optically separated to be picked up by only one camera [11].

There is always some overlap of fluorescence spectra due to the broad spectra of most fluorescent dyes and the imperfect filtering of spectral filters. This necessitates a way to compensate for the small fraction of each dye's fluorescence spectra that is picked up by the other camera [17]. This fraction is determined by measuring the fluorescence intensity of fluorescent dyes individually with the same two-camera setup. The voltage outputs of each sCMOS camera can then be expressed as

$$\frac{V^1}{V^2} = \frac{I_A + \Pi_B I_B}{I_B + \Pi_A I_A} \tag{3}$$

where the Pi functions for each dye are obtained from the relationships

$$\Pi_A = \frac{V^2_{C_B=0}}{V^1_{C_B=0}} \tag{4}$$

and

$$\Pi_B = \frac{V^1_{C_A=0}}{V^2_{C_A=0}} \tag{5}$$

determined experimentally.

The camera sCMOS voltage outputs are expressed as 16-bit numbers in a grayscale TIFF image and are used instead of determining fluorescence intensity directly through another

method. This equation can be rearranged in terms of camera voltage outputs to determine the ratio of the fluorescence intensity of each dye as

$$\frac{I_A}{I_B} = \frac{V^1 - \Pi_B V^2}{V^2 - \Pi_A V^1} = f(T) \qquad (6)$$

in which the fluorescence intensity ratio of temperature-sensitive dye A to temperature-insensitive dye B is determined by measuring the voltage output of each camera and subtracting out any overlap in the emission spectra using the Pi function [17]. For example, Figure 2 showed the emission spectra plot of sulforhodamine 101 and eosin Y which had considerable overlap in the wavelength bands captured by each camera. While camera 1 would primarily capture the fluorescence emission of sulforhodamine 101, the Pi function would be used to subtract any effects caused by the small amount of fluorescence emission of eosin Y within that same wavelength band. The Pi functions must be obtained for temperature-sensitive dye A and temperature-insensitive dye B to account for any overlap in fluorescence spectra picked up by each camera.

The Pi functions are determined by measuring the voltage output of each individual dye for both cameras at various temperatures and taking the ratio of the voltages measured by the two cameras. Additionally, even if the emission spectra do not overlap significantly, it is also possible that the absorption spectra of one dye overlaps with the emission spectra of the other dye. This results in the fluorescence of one dye being partially absorbed by other and can be a critical error in dye selection, as the associated errors in measurement cannot be corrected for [26].

First, suitable fluorescent dyes had to be selected for both water and oil. Each phase must have its own fluorescent dye with the fluorescence emissions being each captured by primarily by one of the two cameras. By selecting temperature-sensitive fluorescent dyes with differing fluorescence emission properties, the MATLAB code can be more easily written to apply to the appropriate relationship between fluorescence intensity and temperature. For example, if one dye has nearly no response within the wavelength band captured by one camera it can be determined that any significant fluorescence captured by that camera must be due to the other dye being used.

2.3.3 Fluorescent Dye Temperature Relationship Calibration

Calibration must be done to determine the relationship between temperature and fluorescence intensity at known temperatures for each fluorescent dye, as literature on every fluorescent dye available when using the exact same wavelength bands is not available. To obtain this relationship for calibration, the experimental setup mentioned in 2.1 Experimental Apparatus was used (except for oil-based dyes, which using a silicon device instead of the PDMS device). The temperature of the dye solution as a result of heat transfer from the water bath to the aluminum block and dye solution. To ensure the dye solution will be constantly pumped through the dye reservoir channel, a syringe pump was used at a constant flow rate of 0.1 mL/min, though results were not found to depend on the flow rate. A thermocouple was used to determine the temperature in the flow channel while the images are taken of the fluorescence of the dye by each camera.

MATLAB code was created to determine the fluorescence intensity ratio between the two images obtained during the same laser pulse from each camera. Additionally, the images must be aligned between the two cameras using a calibration slide with pictures taken from the cameras shown in Figure 13. A rotation transformation is done in MATLAB to rotate the image from the second camera to align with the image from the first camera. This was done for all images obtained at every temperature and the relationship between fluorescence intensity and temperature will be determined for each fluorescent dye for both cameras. A background image of the device exposed to laser light with no fluorescent dye in the flow channel was subtracted from all the fluorescence images. These temperature plots for each fluorescent look similar to that shown in Figure 14 for fluorescein disodium salt hydrate dissolved in water. In this example, the fluorescence intensity measured by camera 1 is related to temperature by the equation $I_1(T) = 14.914 - 27.6932$.

Figure 13. Images of calibration slide for Camera 1 and 2 separately (left and middle, respectively) and superimposed (right) to show differences in alignment of images.

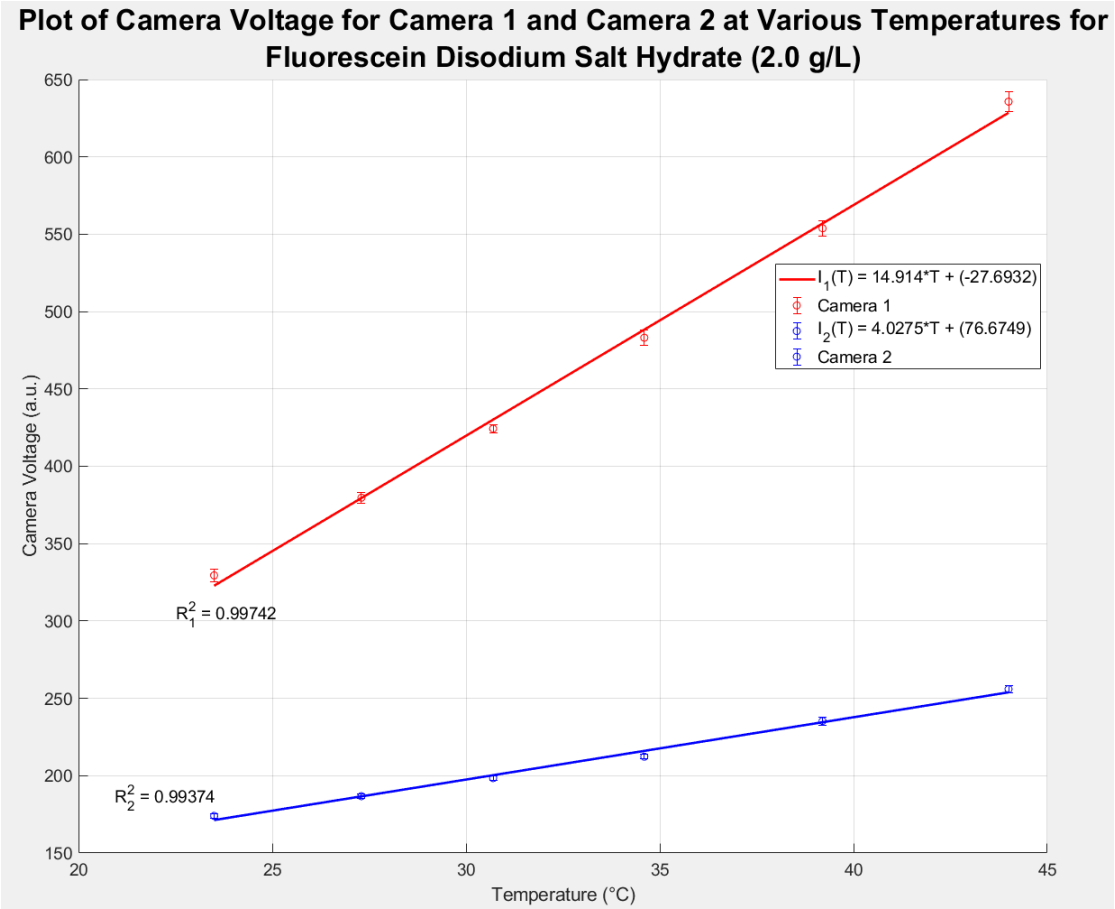

Figure 14. Camera voltage versus temperature for both cameras for fluorescein disodium salt hydrate. Error bars represent a single standard deviation.

2.3.4 Micro-scale and Porous Media Micromodel Device Single-Phase and Multiphase Flow Temperature Measurement

The temperature and fluorescence intensity calibration functions that were determined were then applied to micro-scale devices and porous media micromodels made of PDMS or silicon. The dye combinations are tested for each fluid phase for simple heat transfer scenarios that have known analytical solutions, such as a linear temperature gradient. Fluorescence intensity was determined at various points in the device, used to determine the temperature, and compared to the expected values based on an Ansys simulation. The

expected temperature values at 12 locations within the devices are shown below in Table 1 and will be compared with the temperature values measured using LIF thermometry.

Table 1

Summary of Ansys simulation results presented in A.1.2 Temperature Gradient Applied to PDMS Device when one side of the device is at a measured temperature of 66.8 °C and the other is at a measured temperature of 21.8 °C.

Simulated Temperature Gradient Results	
Distance From Cold Reservoir (mm)	Temperature (°C)
6.60	26.67
7.58	29.09
8.57	31.58
9.55	34.11
10.53	36.71
11.52	39.45
12.50	42.27
13.48	45.24
14.47	48.39
15.45	51.65
16.43	55.15
17.42	58.72

After testing the individual fluid phases, multiphase flow configurations were then tested. Both the oil and the water must be tested as primary and secondary phases using the most temperature-sensitive fluorescent dyes. This multiphase flow is tested in the silicon porous media micromodel device due to the oil-phase making the PDMS device not a viable option. MATLAB code is used to detect which regions of the images captured contain which fluid and apply the corresponding temperature calibration equations based on the fluorescent dyes used.

Due to the small droplets of the secondary phase having a low flow rate compared to the primary phase, the effects of interactions between the two fluids in multiphase flow should be small, therefore similar results should be obtained compared to those obtained for just the primary phase for the micro-scale devices. However, due to the complexity of flow through porous media, temperature distributions may differ greatly between the individual phases and the multiphase flows.

3 RESULTS AND DISCUSSION

3.1 Temporal and Spatial Resolution

3.1.1 Temporal Resolution

To determine the temporal resolution of the LIF thermometry technique, the number of images required to achieve consistent, low variance results with a consistent mean value was determined. To test this, 4 records were taken of a series of 24 images of sulforhodamine B sodium salt at a concentration of 0.2 g/L in water at room temperature (roughly 17.2 °C for this experiment) inside a PDMS micromodel. An averaging window of 10 pixels, corresponding to $(6.5\ \mu m)^2$ was used to average the voltage values in the region of the image being measured. This size of the region to use for this pixel averaging window is determined in ***3.1.2 Spatial Resolution*.** The camera voltage values determined for each camera are shown below in Figure 15. For this fluorescent dye, the fluorescence response is strongest for the wavelength band captured by camera 2 (610-640 nm). Voltage ratios of the two cameras were used instead of the voltages of the individual cameras to ensure that the temporal and spatial resolution were not the reason ratiometric LIF did not work well.

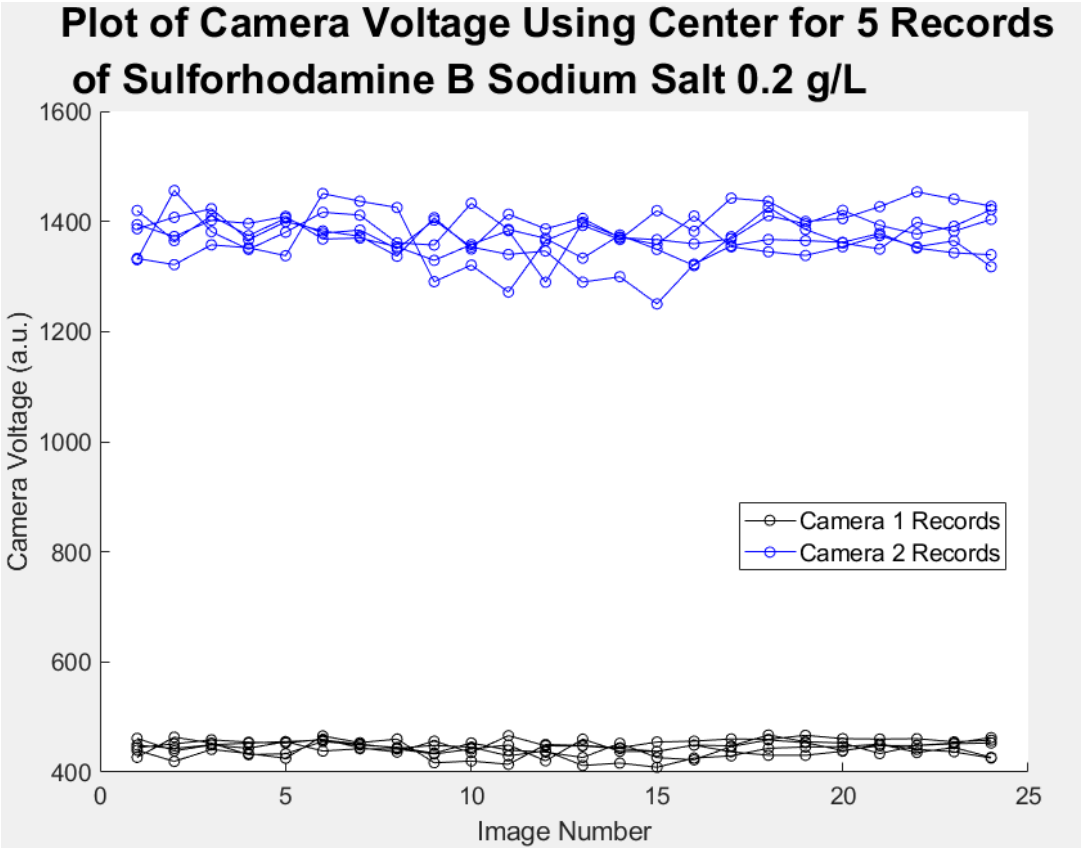

Figure 15. Camera voltage plot for all 5 records versus image number. Dye used is sulforhodamine B sodium salt 0.2 g/L in water.

The ratio of the voltage values is plotted in Figure 16 and has values ranging from 0.330 to 0.338 with an approximate mean of 0.335. The mean value of the camera voltage ratio is plotted against the number of images used in Figure 17. The mean value for most records was found to reach a fairly stable value when using at least 15 of the 24 images. Figure 18 shows the standard deviation of the camera voltage ratio versus number of images, with most records having standard deviations less than 0.0040 that does not vary significantly after 15 images. To better quantify the standard deviation, the coefficient of variance was determined which calculates the percent standard deviation relative to the mean value. The

plot of this is shown in Figure 19 and indicates that each record has a coefficient of variance less than 1.5% after 10 images.

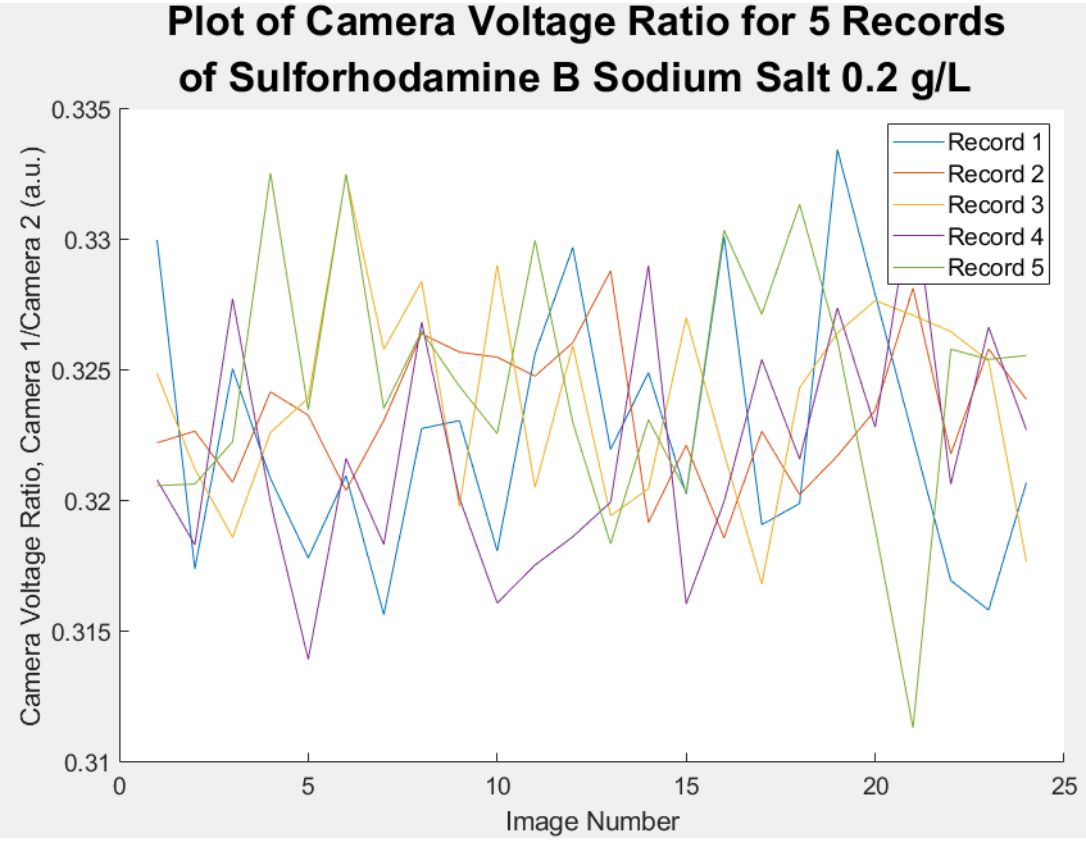

Figure 16. Camera voltage ratio plot for all 5 records versus image number. Dye used is sulforhodamine B sodium salt 0.2 g/L in water.

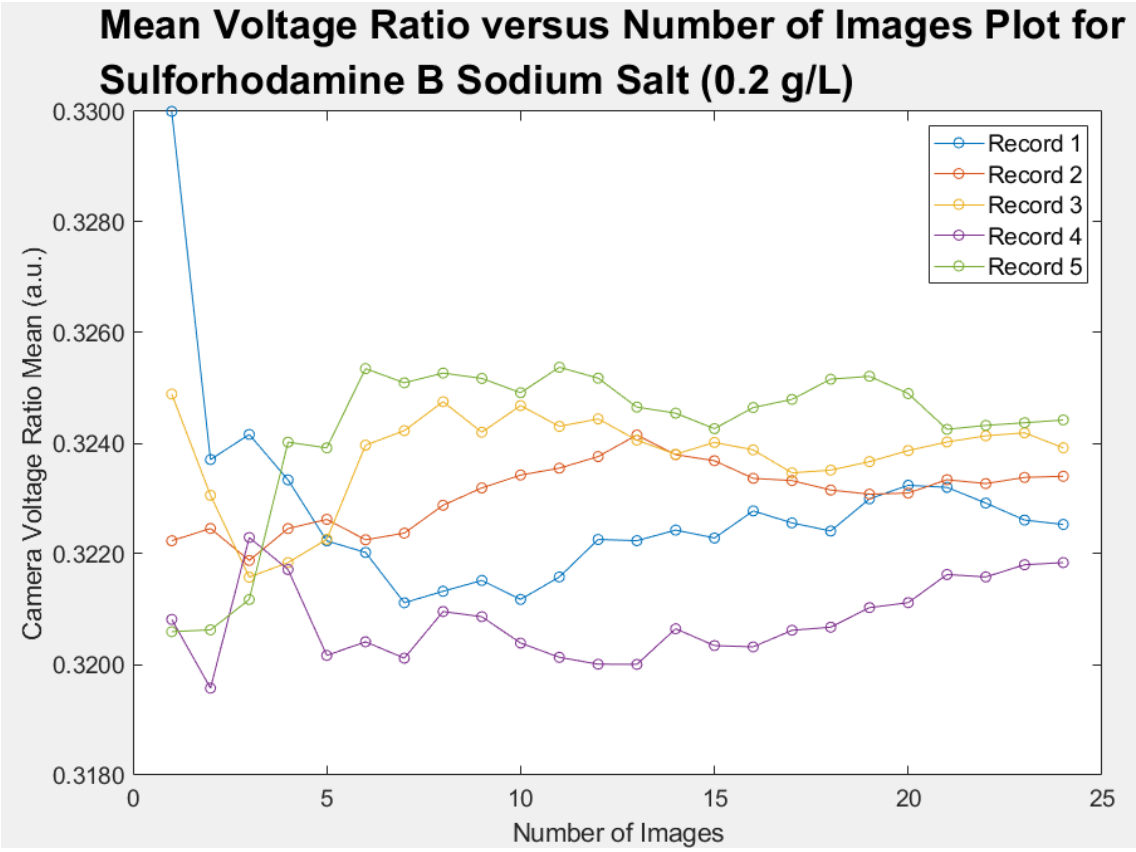

Figure 17. Camera voltage ratio mean plot for all 5 records versus number of images. Dye used is sulforhodamine B sodium salt 0.2 g/L in water.

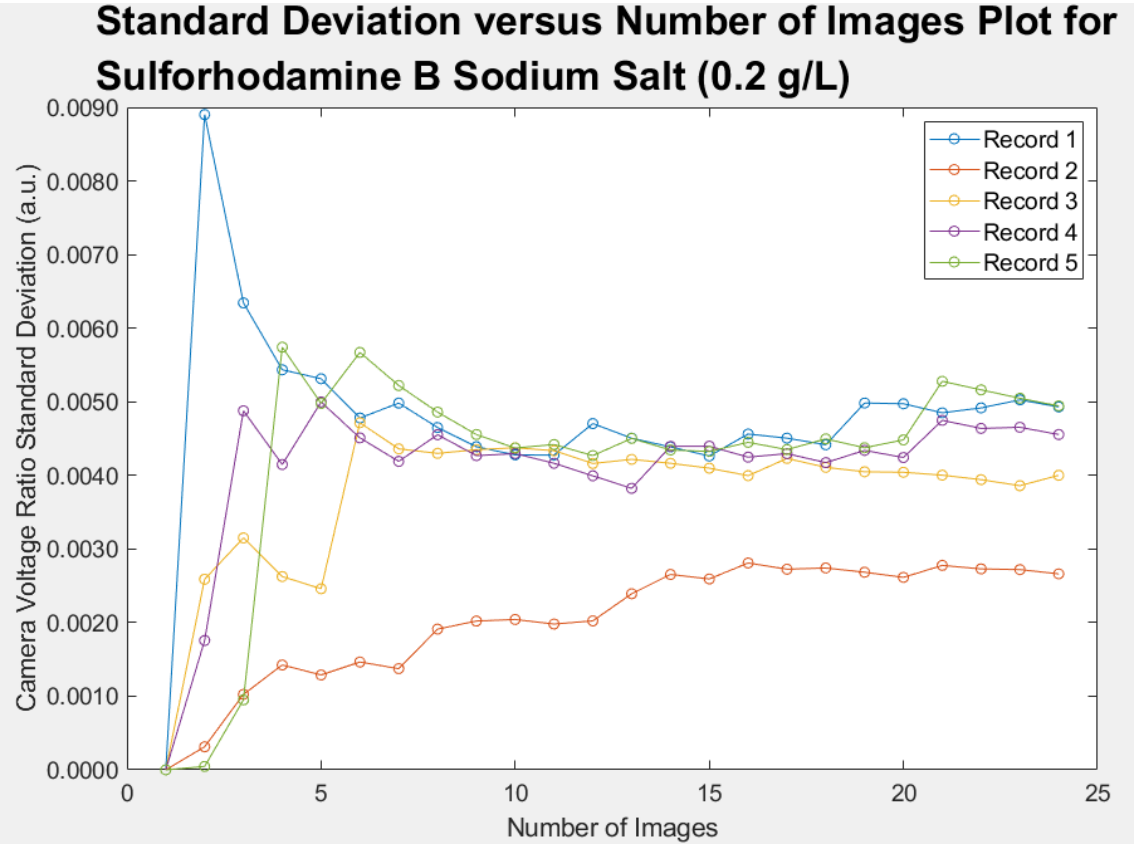

Figure 18. Camera voltage ratio standard deviation plot for all 5 records versus number of images. Dye used is sulforhodamine B sodium salt 0.2 g/L in water.

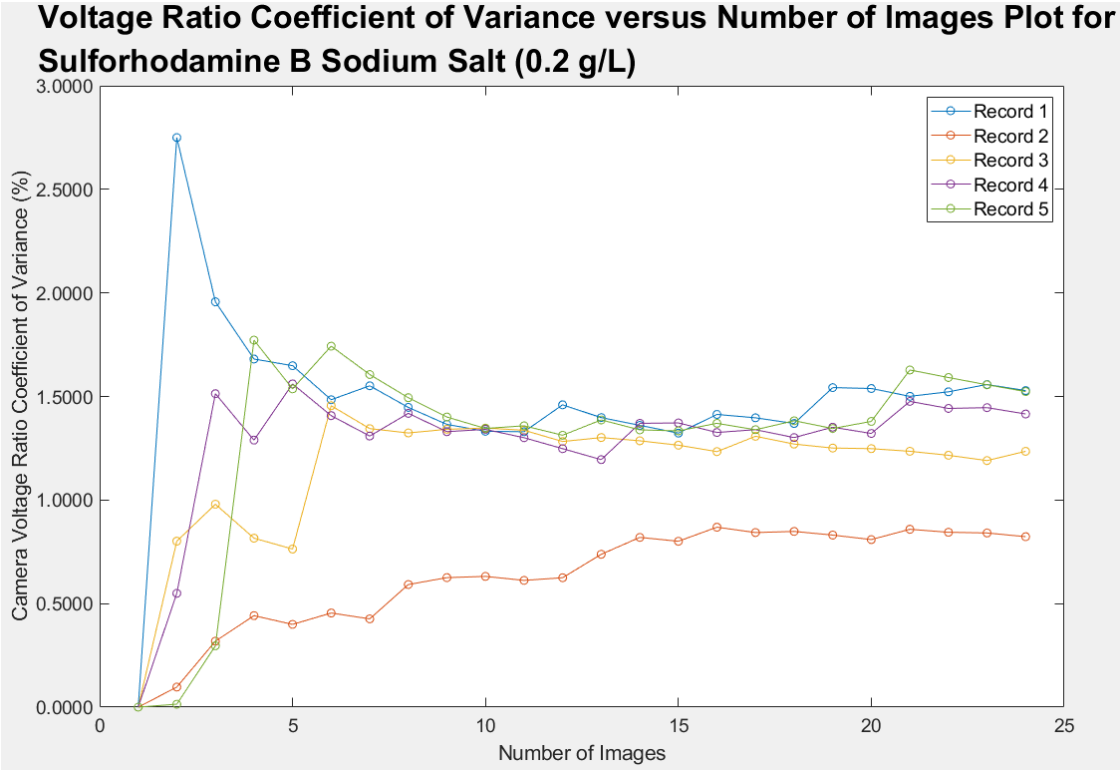

Figure 19. Camera voltage ratio coefficient of variance plot for all 5 records versus number of images. Dye used is sulforhodamine B sodium salt 0.2 g/L in water.

Additional analysis was performed to determine the number of data records that should be averaged together to produce a sufficiently low value for the coefficient of variance. The effects of averaging together between 1 and 5 records was tested. The mean ratio plot is shown in Figure 20 and indicate a similar mean voltage ratio no matter how many records are averaged together as long as at least about 15 images are used. Standard deviation, shown in Figure 21, showed a significant improvement when using more records. The standard deviation when using at least 3 records had values near 0.0020 when using at least 10 images. Finally, the camera voltage ratio coefficient of variation was compared for averaging different numbers of records together and is shown in Figure 22. Using at least 10

images and averaging at least 3 records together seems to produce a consistently low coefficient of variation of just 0.6%. In order to have a consistent result while still maintaining a high temporal resolution, 3 records of 15 images were used for all future experiments. The root-mean-square error for the voltage values of 15 records for sulforhodamine B sodium salt was found to be 16.2 and 51.6 for camera 1 and camera 2, respectively. As a percent of the mean value of the voltage of those 15 images, this is a normalized root-mean-square error of 3.68% and 3.78% for each camera, respectively.

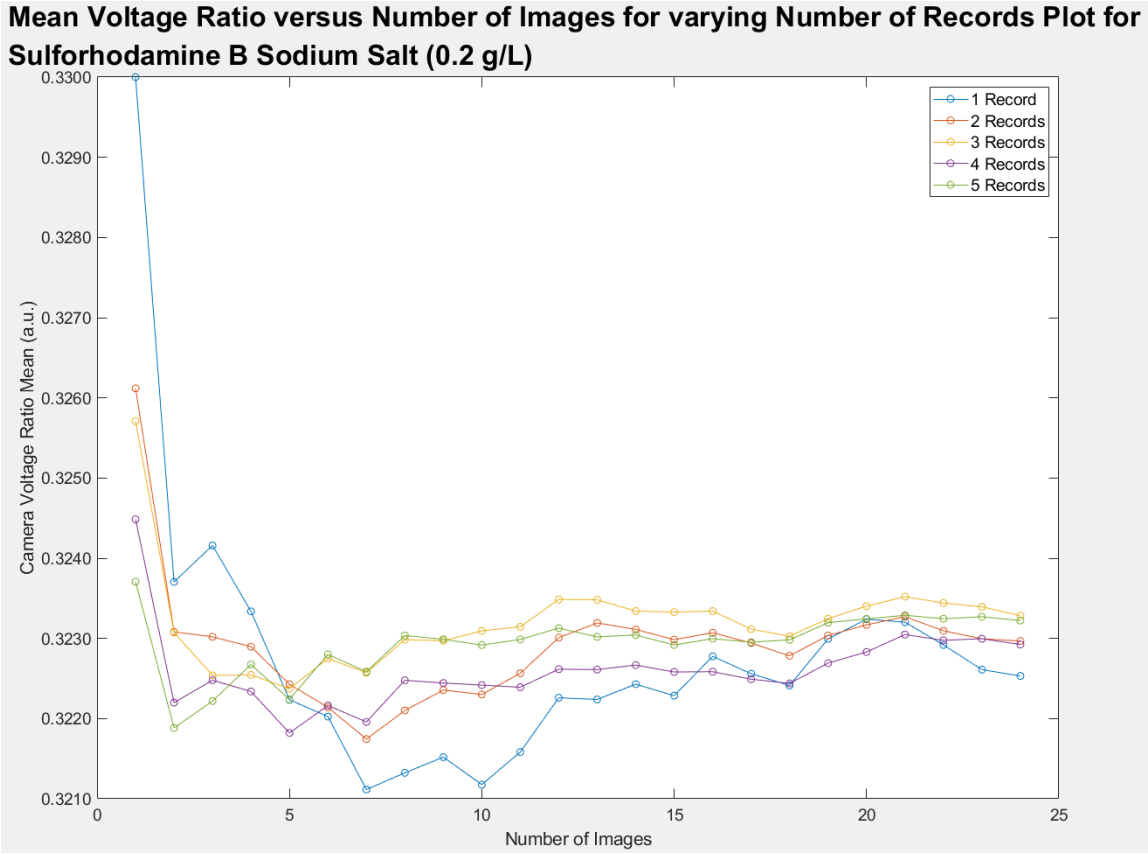

Figure 20. Mean versus number of images for sulforhodamine B sodium salt 0.2 g/L in water when averaging a varying number of records together.

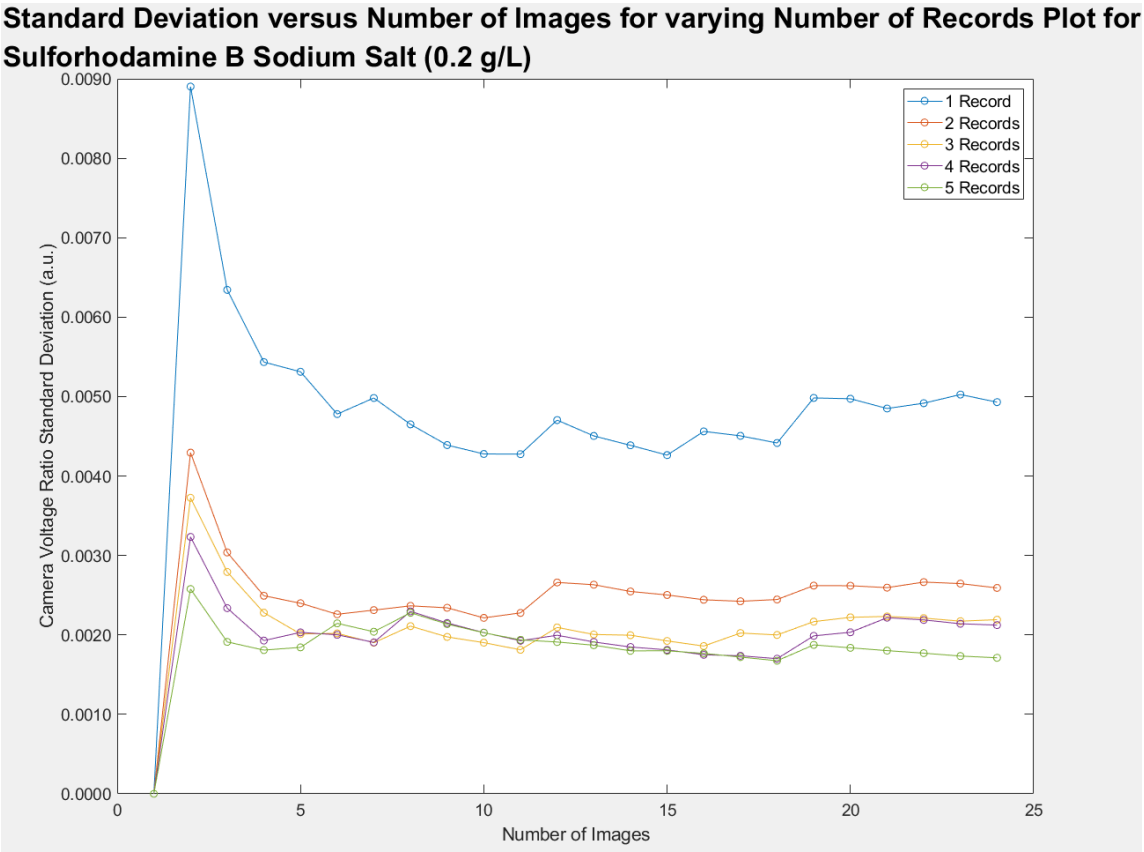

Figure 21. Standard deviation versus number of images for sulforhodamine B sodium salt 0.2 g/L in water when averaging a varying number of records together.

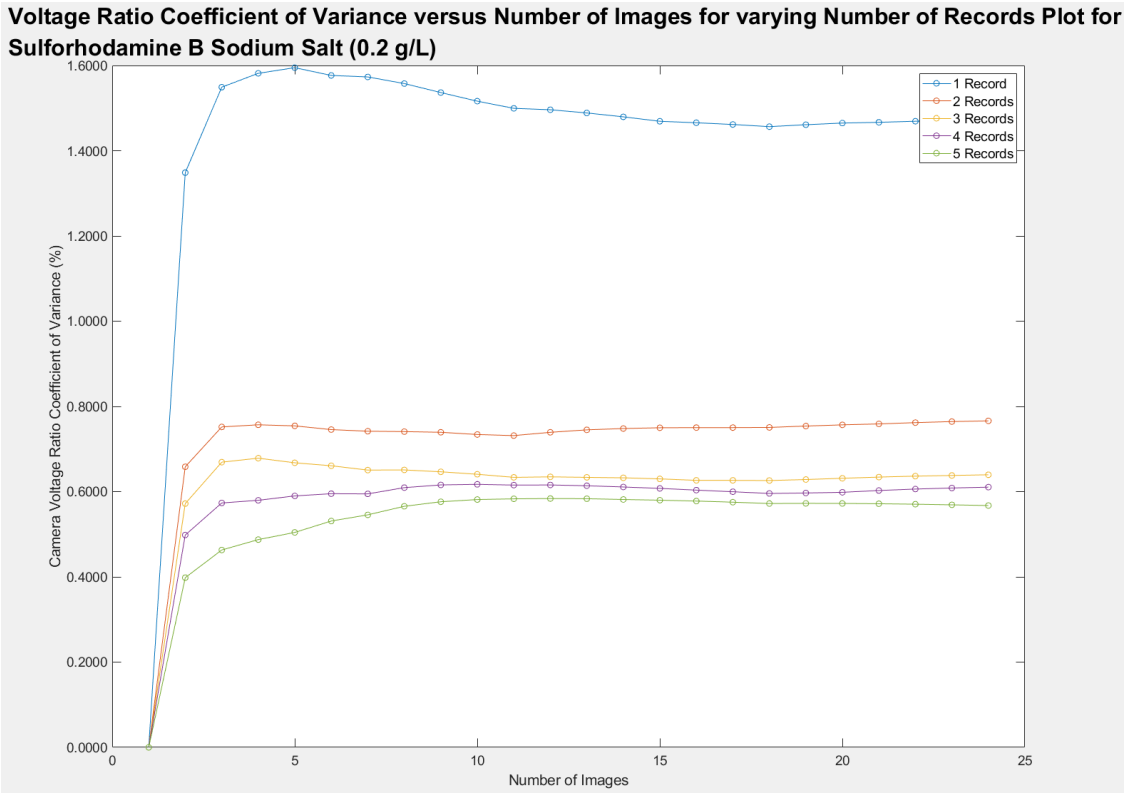

Figure 22. Coefficient of variation versus number of images for sulforhodamine B sodium salt 0.2 g/L in water when averaging a varying number of records together.

3.1.2 Spatial Resolution

A pixel averaging window was used to improve the precision of the LIF measurement technique. To determine the size of the pixel averaging window necessary to have a low coefficient of variation, the results were compared for different numbers of pixels for the square pixel averaging window. Results were compared for a 1x1 pixel averaging window (a single pixel) up to a 100x100 pixel averaging window (52 µm x 52 µm) using 3 records of 15 images averaged. The standard deviation versus square pixel averaging window size is shown below in Figure 23. The standard deviation decreases rapidly as the pixel averaging window size is increased up to about 10 pixels. Further reductions in standard deviation are

much lower as the pixel averaging window size is increased further. The coefficient of variance, plotted in Figure 24, indicates that the standard deviation is less than 1% of the mean value for the 10 x 10 pixel averaging window. As the coefficient of variation does not decrease significantly as the pixel averaging window is increased further, a 10 x 10 pixel averaging window was selected. This means that each measurement is taken in 5.2 µm by 5.2 µm regions within the device.

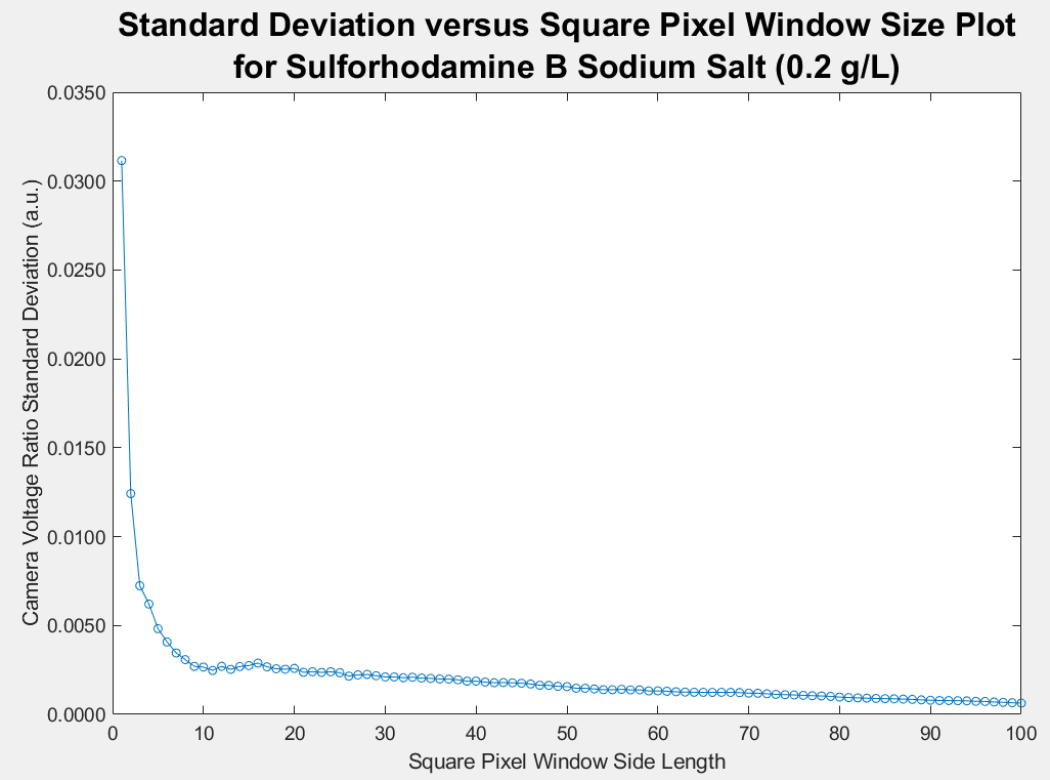

Figure 23. Standard deviation versus square pixel averaging window size for sulforhodamine B sodium salt 0.2 g/L in water when averaging 15 images.

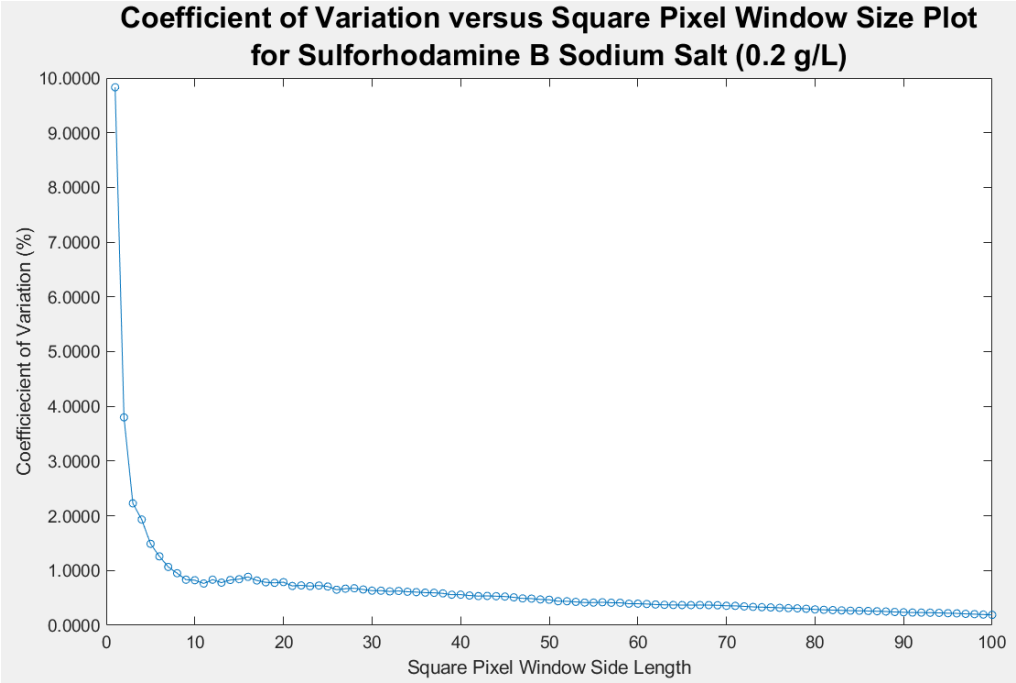

Figure 24. Coefficient of variation versus square pixel averaging window size for sulforhodamine B sodium salt 0.2 g/L in water when averaging 15 images.

3.2 Fluorescence Spectra of Fluorescent Dyes

Attempts were made to determine the fluorescence emission spectra of the fluorescent dyes experimentally to be able to determine the temperature-sensitive and temperature-insensitive wavelength ranges of each fluorescent dye but it was found that fluorescence emission spectra were too weak to measure accurately with the available equipment. Due to this, the fluorescence spectra for all fluorescent dyes tested was not available so not all results for temperature calibration could be compared with the fluorescence spectra.

3.3 Single Dye Temperature Calibration

Temperature versus camera voltage relationships were first obtained for individual dyes soluble in either water or n-decane. The PDMS device was used for the water device while a

similarly sized silicon device was used, consisting of a glass microscope slide bonded directly a PDMS wafer with the channel geometry made by photolithography. For water, the dyes tested were sulforhodamine 101, eosin Y, sulforhodamine B sodium salt, and fluorescein disodium salt hydrate due to being insoluble in oil. Eosin Y has been used in LIF thermometry with ethanol but is also soluble in water [27]. Sulforhodamine B sodium salt and sulforhodamine 101 were found to be insoluble in engine oil and also insoluble in n-decane due to similar polarity properties [28].

For n-decane, the dyes pyrromethene 567 and pyrromethene 597-8C9 were selected as n-decane-soluble dyes [28], [29]. While the papers found using these dyes indicate their solubility in hexadecane and engine oil, these two dyes were also found to be soluble in n-decane due to its similar polarity properties. Concentrations for all dyes were determined experimentally such that fluorescent dyes have camera voltages around the same order of magnitude and concentrations are close enough to each other that using two dyes together produces a noticeable change in camera voltages. It was found that even if dyes had a noticeable fluorescence response individually, combining the two dyes and having a significant difference in concentration can cause the results to resemble the results for just the single dye with the highest concentration.

The n-decane-soluble fluorescent dyes were tested using a silicon device, consisting of a silicon wafer bonded to a glass slide. This device has similar optical properties to the PDMS device in terms of the optical length. One critical issue with using PDMS for oil-based fluorescent dyes is that unmodified PDMS absorbs organic solvents and hydrophobic

molecules [30]. Due to this property, any fluorescent dyes that were soluble in n-decane would also be absorbed by the PDMS, causing the fluorescence intensity to increase as the device absorbs increasing amounts of PDMS. The resulting increase in the camera voltage would often be enough to oversaturate the cameras, causing the voltage values to exceed the maximum value for a 16-bit number and making it incredibly difficult to measure the fluorescence of the fluorescent dye itself. Attempts were made to use a Teflon coating done by other researchers, but this proved to only minorly improve the PDMS device when tested. This is possibly due to Teflon-AF 2400 being used instead of the prohibitively expensive Teflon-AF 1600 used by other researchers [30].

3.3.1 Eosin Y (Acid Red 87) in PDMS Device

Eosin Y was tested in water at a concentration of 0.8 g/L. The emission spectra at various temperatures are shown below in Figure 9. For the wavelengths captured by camera 1 (547.5-572.5 nm) and the wavelength captured by camera 2 (610-640 nm), camera 1 is expected to have a negative temperature sensitivity while camera 2 is expected to indicate a slightly positive temperature sensitivity. The results obtained experimentally are shown in Figure 25. Results for camera 1 and camera 2 agree with expectations based on Figure 9, showing a positive temperature sensitivity with a stronger response in the wavelength band captured by camera 1. This makes eosin Y a possible candidate to be used as a temperature-sensitive fluorescent dye primarily captured by camera 1 due to its low response for camera 2. Based on the plot obtained by normalizing the camera voltage by the camera voltage value at the lowest temperature, both spectral bands of the dye have approximately the

same temperature sensitivity. Camera 1's voltage has a temperature sensitivity is 1.49%/K while camera 2's voltage has a temperature sensitivity of 1.58%/K, as shown in the equations of Figure 26.

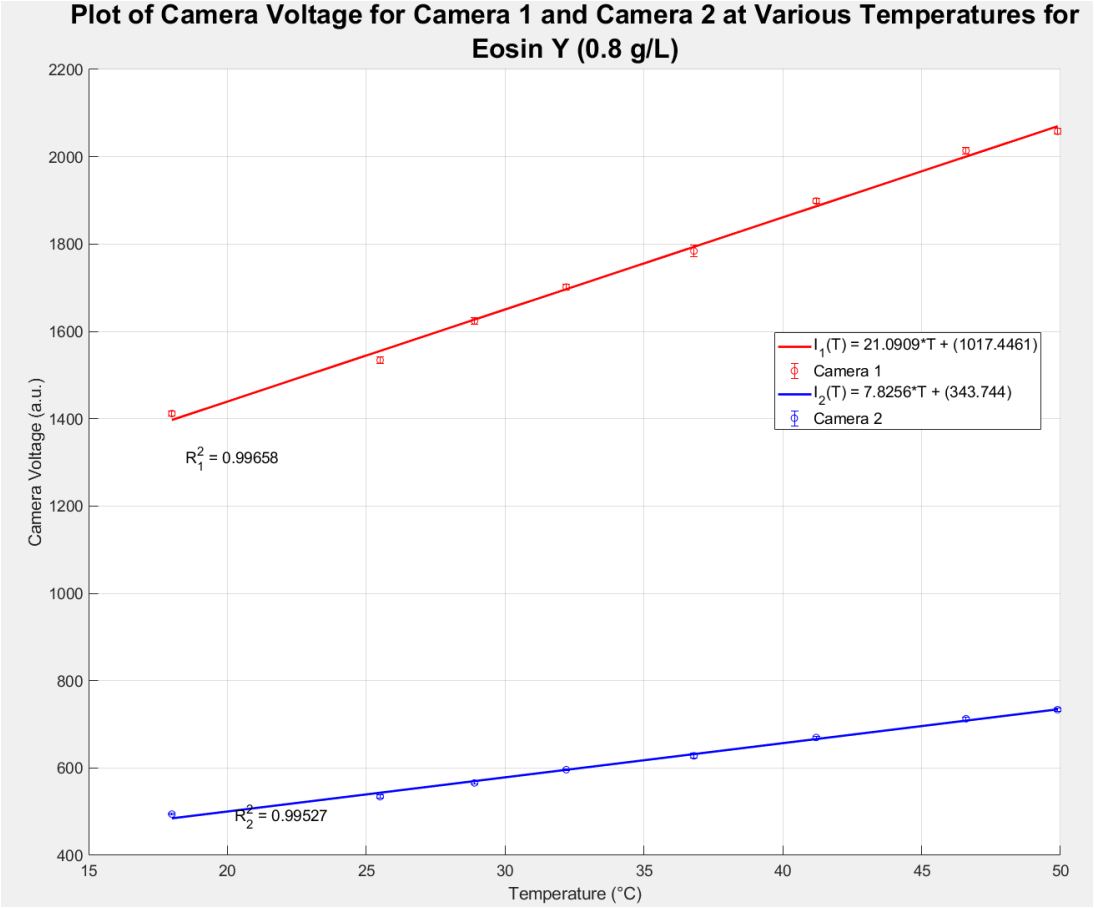

Figure 25. Camera voltage versus temperature for both cameras for eosin Y. Error bars represent a single standard deviation.

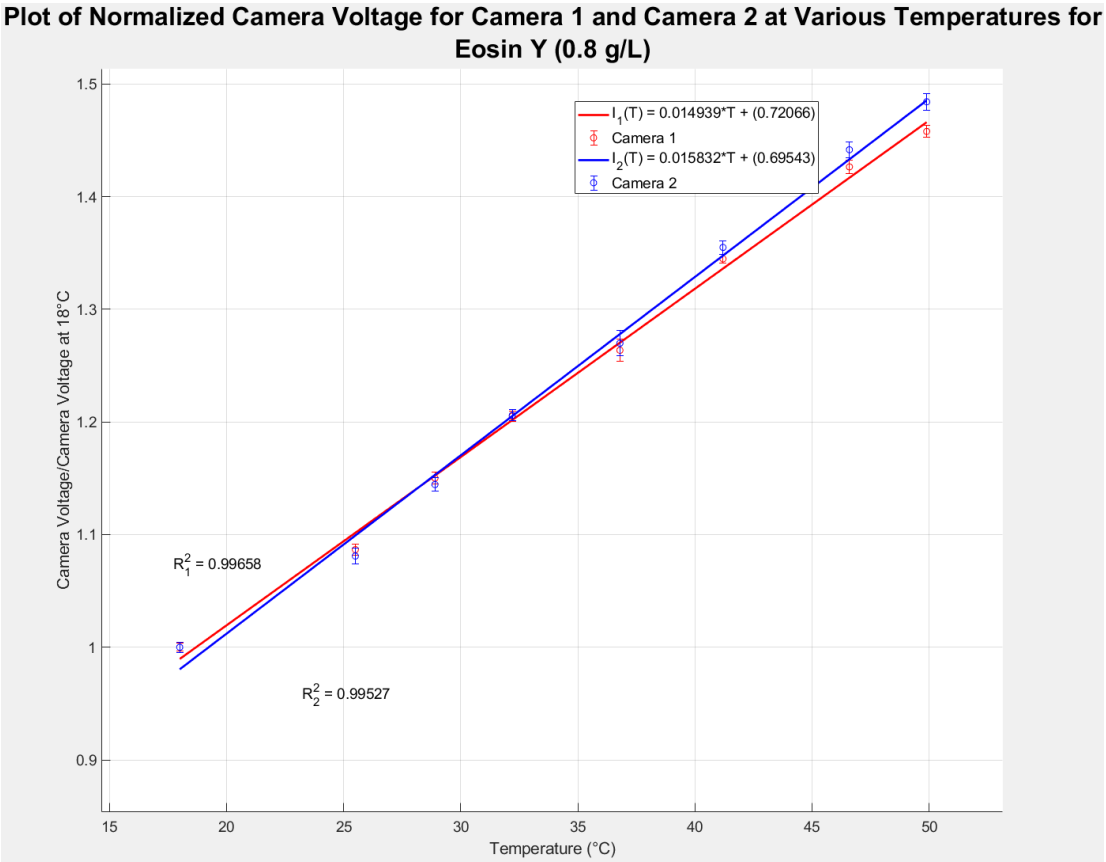

Figure 26. Normalized camera voltage versus temperature for both cameras for eosin Y. Error bars represent a single standard deviation.

Plotting the ratio of voltages between the two cameras by the ratio of the voltages of the camera at 18.0 °C produced the plot shown in Figure 27. This procedure is done when using ratiometric LIF technique to increase the temperature sensitivity of the fluorescent dyes and reduce the effects of laser light variation. However, the temperature sensitivity actually decreased to 0.0186%/K and the coefficient of variance decreased substantially to a value of 0.357. This does not match the expected results when applying the ratiometric LIF technique and the results indicate the regular LIF technique has greater temperature

sensitivity and a more consistent relationship between temperature and the fluorescence of the dye.

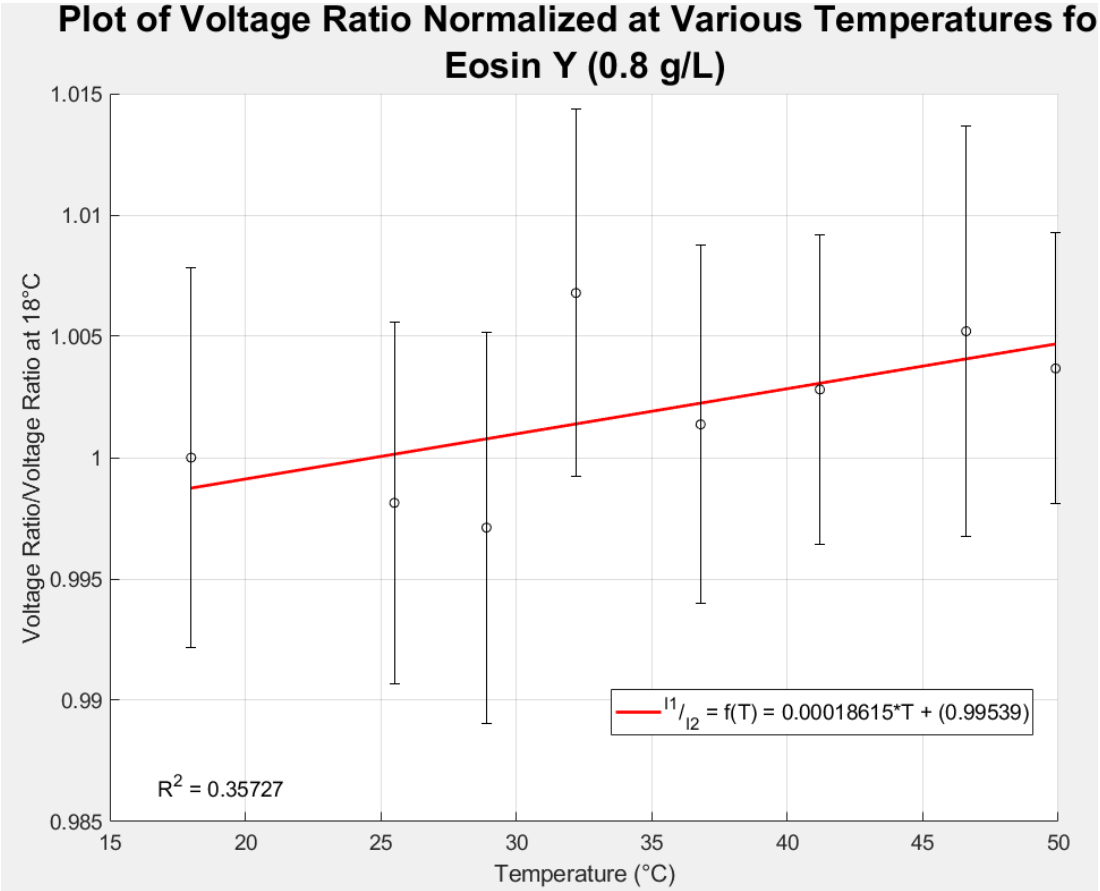

Figure 27. Normalized voltage ratio plot for eosin Y. Error bars represent a single standard deviation.

3.3.2 Sulforhodamine 101 (Sulforhodamine 640) in PDMS Device

The camera voltage plot for sulforhodamine 101 at a concentration of 0.4 g/L is shown below in Figure 28. Camera 2 showed a consistent, positively-temperature sensitive response as the temperature increased while camera 1 showed essentially no response. Normalizing the camera voltage values by dividing by the voltage values at the minimum temperature measured allows for the temperature sensitivity of the fluorescence captured

by each camera to be evaluated. The voltage of camera 1 indicates a temperature sensitivity of 0.481%/K while the voltage of camera 2 indicates a temperature sensitivity of 1.23%/K. This can be seen in Figure 29. Looking at the emission spectra versus temperature plot shown in Figure 11, this agrees with the results of other researchers. Camera 1, which captures fluorescence from 547.5-572.5 nm, shows essentially no fluorescence being captured in this wavelength band. The emission spectra shows that there should be essentially no fluorescence response in this wavelength band, matching the experimental results. In the wavelength band of 610-640 nm captured by camera 2, the fluorescence response increased with temperature, rather than decreasing. This is likely due to the solvent being used in that paper being ethanol instead of water [23].

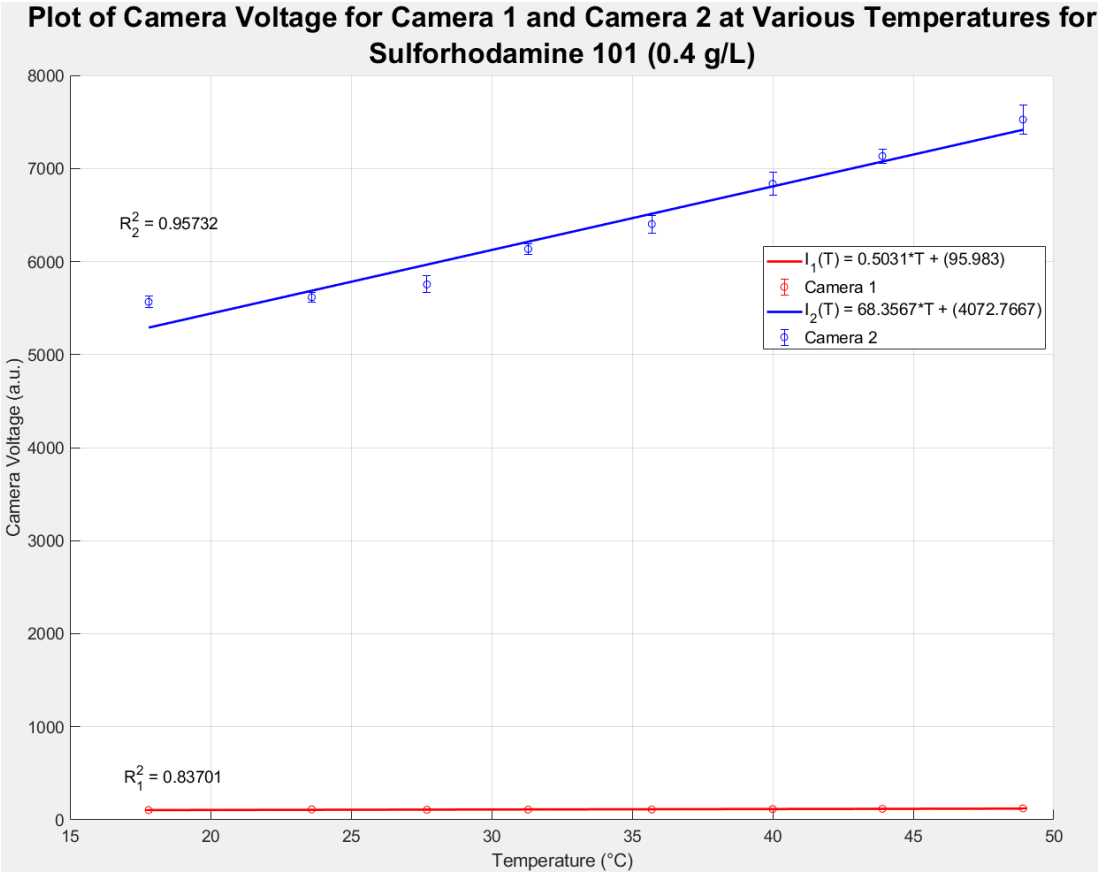

Figure 28. Camera voltage versus temperature for both cameras for sulforhodamine 101. Error bars represent a single standard deviation.

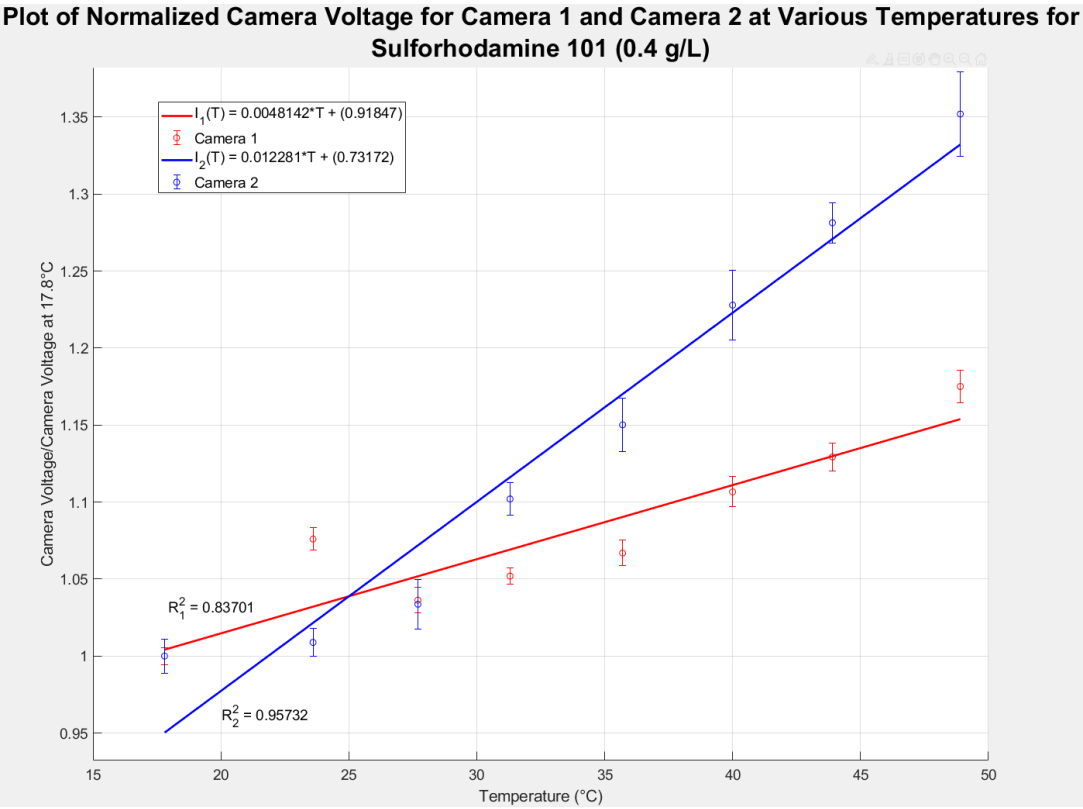

Figure 29. Normalized camera voltage versus temperature for both cameras for sulforhodamine 101. Error bars represent a single standard deviation.

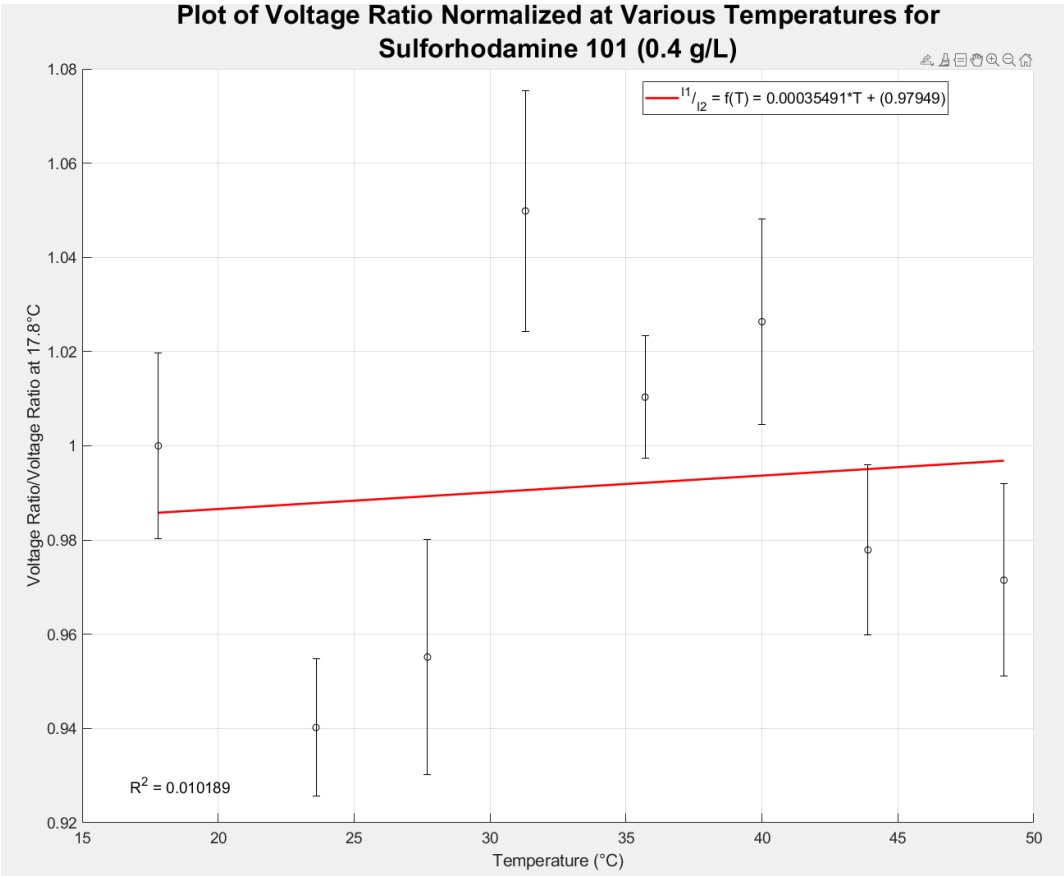

Figure 30. Normalized voltage ratio plot for sulforhodamine 101. Error bars represent a single standard deviation.

3.3.3 Sulforhodamine B Sodium Salt (Sulforhodamine 620 or Kiton Red 620) in PDMS Device

The results for sulforhodamine B sodium salt are shown below in Figure 31, indicating that dye is negatively temperature-sensitive in both of the wavelength bands captured by each camera. Sulforhodamine B sodium salt shows a larger fluorescence response in the wavelength range of 610-640 nm captured by camera 2. The camera voltage captured by camera 2 also decreases more rapidly than for camera 1, indicating a larger negative temperature sensitivity. The normalized camera voltage plot, shown in Figure 32, indicates a temperature sensitivity of -1.25%/K and -1.22%/K for camera 1 and camera 2,

respectively. The normalized voltage ratio, plotted in Figure 33, indicates a slightly positive increase in voltage ratio with temperature but also has a poor fit.

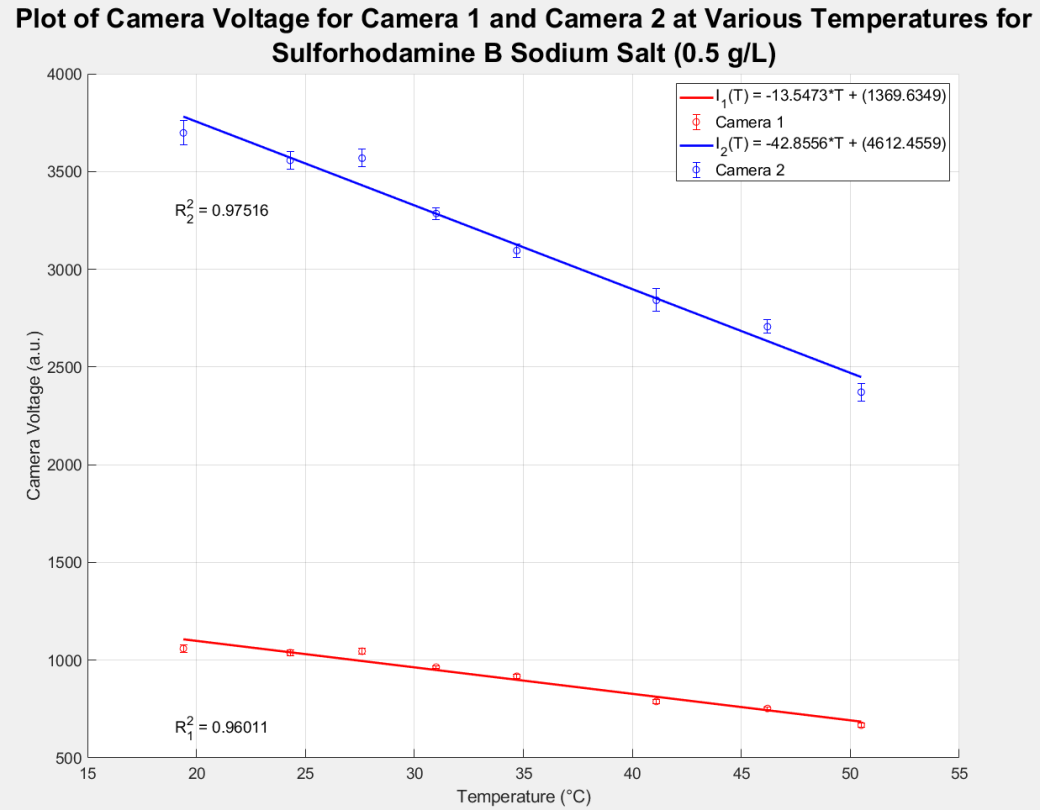

Figure 31. Camera voltage versus temperature for both cameras for sulforhodamine B sodium salt. Error bars represent a single standard deviation.

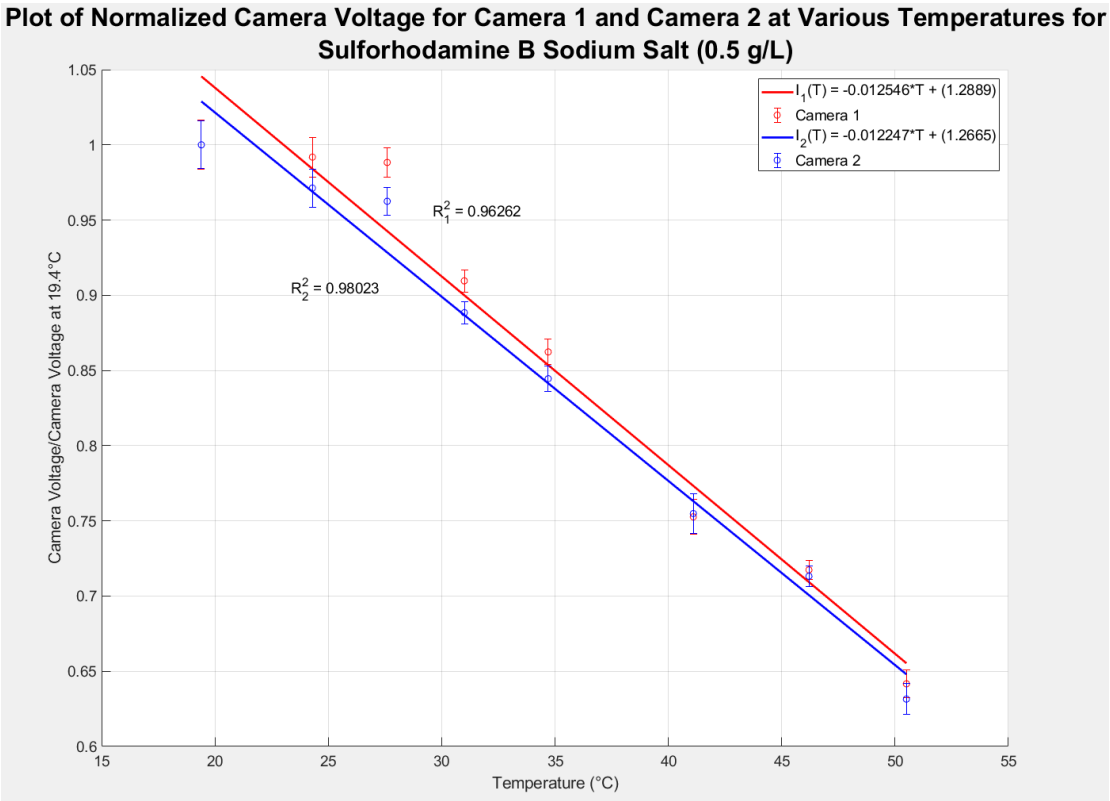

Figure 32. Normalized camera voltage versus temperature for both cameras for sulforhodamine B sodium salt. Error bars represent a single standard deviation.

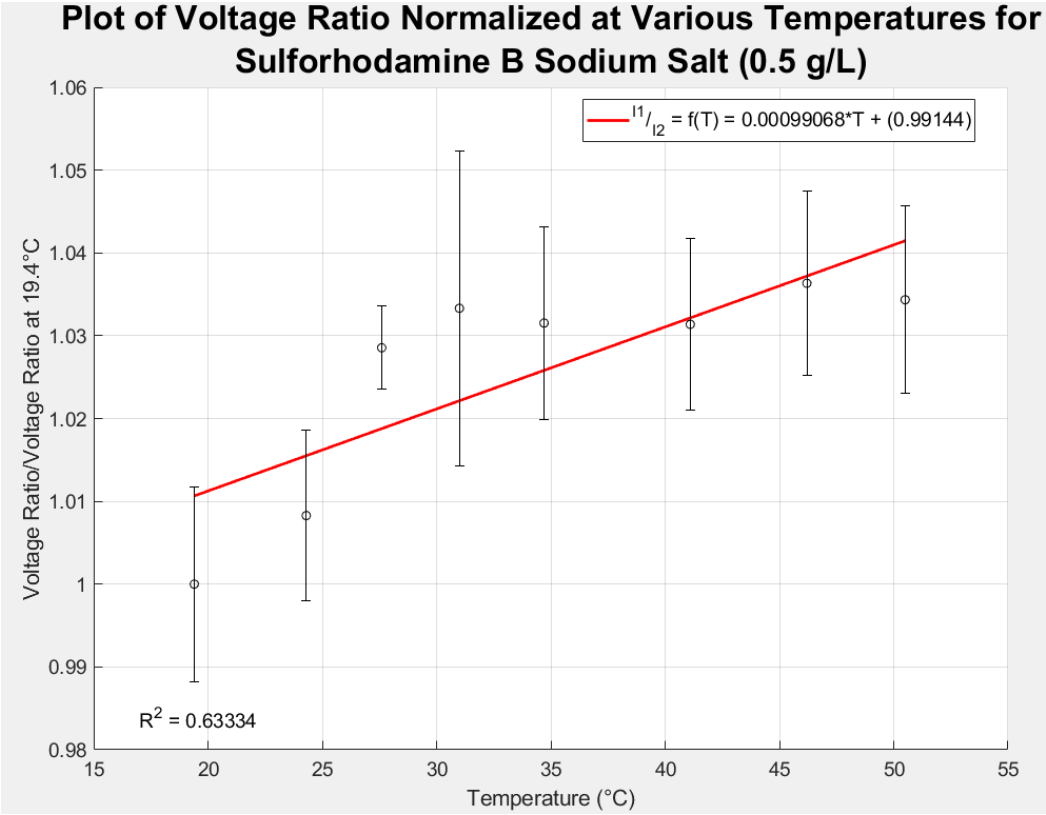

Figure 33. Normalized voltage ratio plot for sulforhodamine B sodium salt. Error bars represent a single standard deviation.

3.3.4 Fluorescein Disodium Salt Hydrate (Fluorescein Sodium Salt) in PDMS Device

Fluorescein disodium salt hydrate at a concentration of 2.0 g/L was selected as another water-soluble dye. This fluorescent dye has a noticeably weaker fluorescence intensity than other fluorescent dyes but was not increased to a concentration higher than 2.0 g/L to avoid issues with the concentration of it relative to another fluorescent dye being too high when used in the ratiometric technique. The camera voltage plot is shown below in Figure 34 and indicates a positive temperature sensitivity response from the portion of fluorescence captured by each camera. The data was normalized by dividing by the camera voltage of each camera at the minimum temperature measure (23.5 °C). The results of this are shown

in Figure 35 and indicate that the camera voltage of camera 1 increased by 4.53%/K while the camera voltage of 2 increases by 2.31%/K. These results agree with the fluorescence spectra plot shown in Figure 12, which indicate a strong positive temperature sensitive and a higher fluorescence intensity within the wavelength band captured by camera 1.

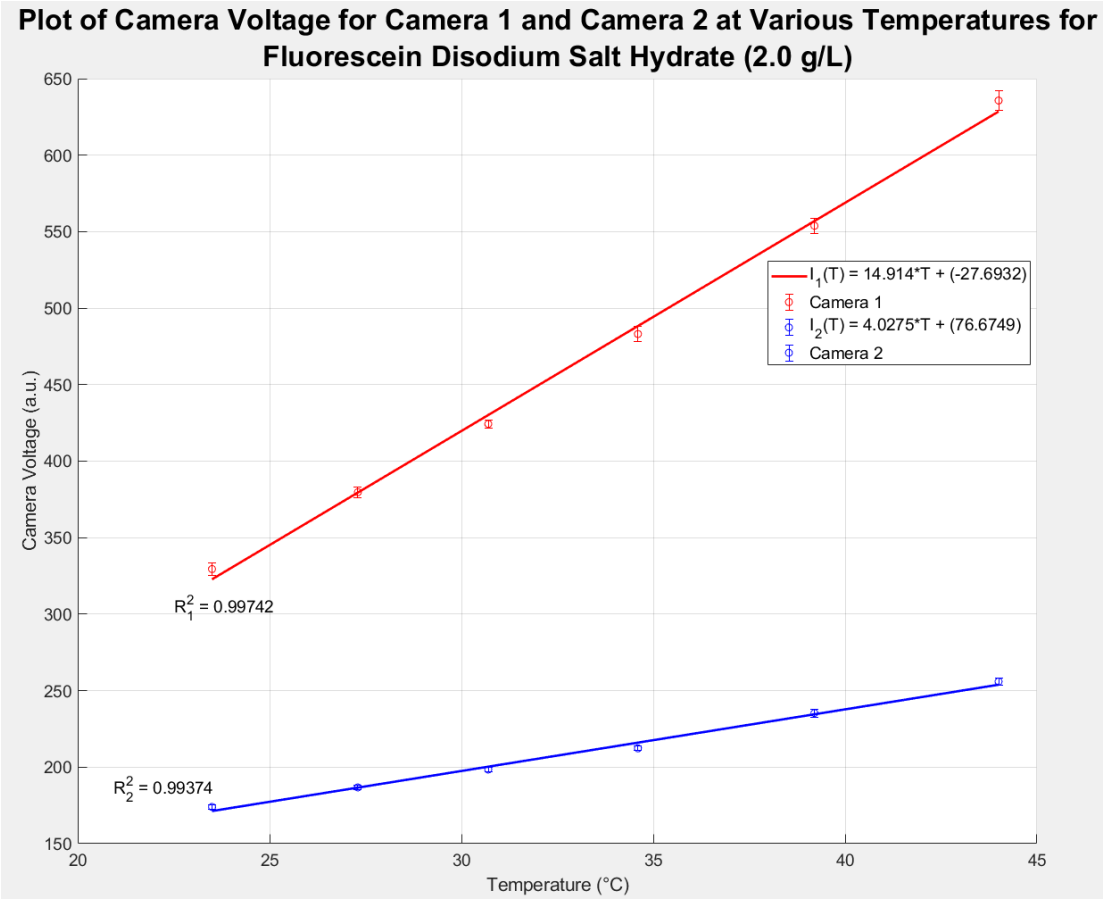

Figure 34. Camera voltage versus temperature for both cameras for fluorescein disodium salt hydrate. Error bars represent a single standard deviation.

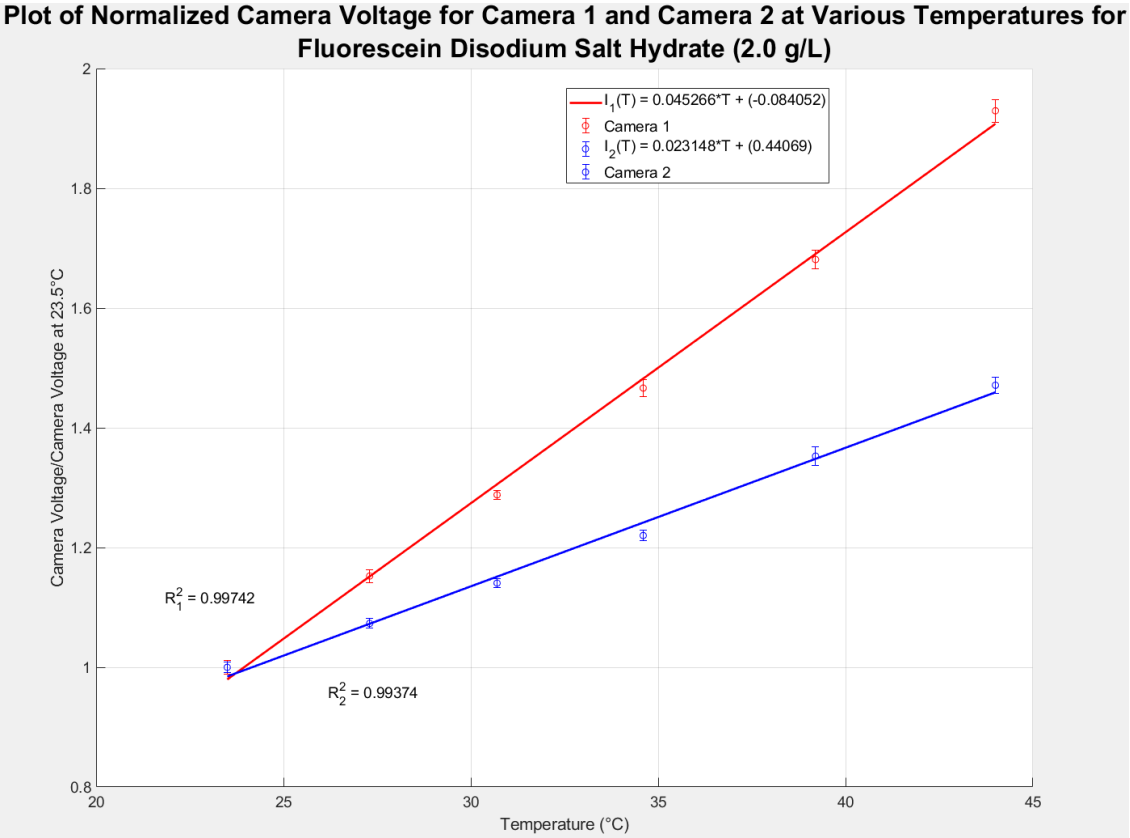

Figure 35. Normalized camera voltage versus temperature for both cameras for fluorescein disodium salt hydrate. Error bars represent a single standard deviation.

3.3.5 Pyrromethene 567 in Silicon Device

In regards to the emission spectra of pyrromethene 567 and pyrromethene 597-8C9, plots of the emission spectra of each of these dyes separately was not found. The plot shown in Figure 10, showing the emission spectra at various temperatures for a mixture of both dyes. The fluorescent emission value for this dye combination appears to decrease with temperature for nearly all wavelengths. This fluorescent dye was noted specifically to have a lower temperature sensitivity than pyrromethene 597-8C9 [22].

In terms of the results for pyrromethene 567 specifically, the camera voltage plots for a concentration of 0.050 g/L are shown below in Figure 36 and Figure 37. The results for this dye indicate a stronger response within the wavelength band captured by camera 1 with very little temperature sensitivity. The temperature sensitivity for this fluorescent dye is 0.174%/K and 0.196%/K for camera 1 and camera 2, respectively. This temperature sensitivity is far too low to be useful by itself. Similarly to the other fluorescent dyes, the normalized voltage ratio plot, shown in Figure 38, shows a very poor fit. The low temperature sensitivity was expected, although no temperature sensitivity value was noted in literature.

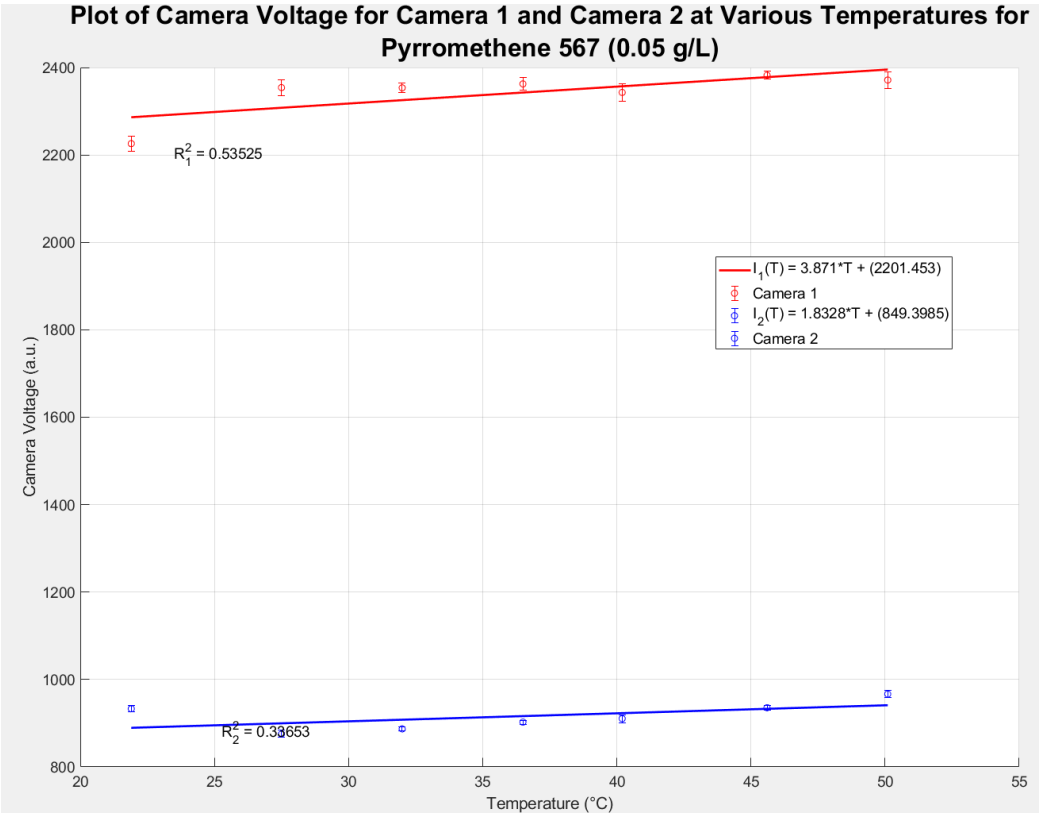

Figure 36. Camera voltage versus temperature for both cameras for pyrromethene 567. Error bars represent a single standard deviation.

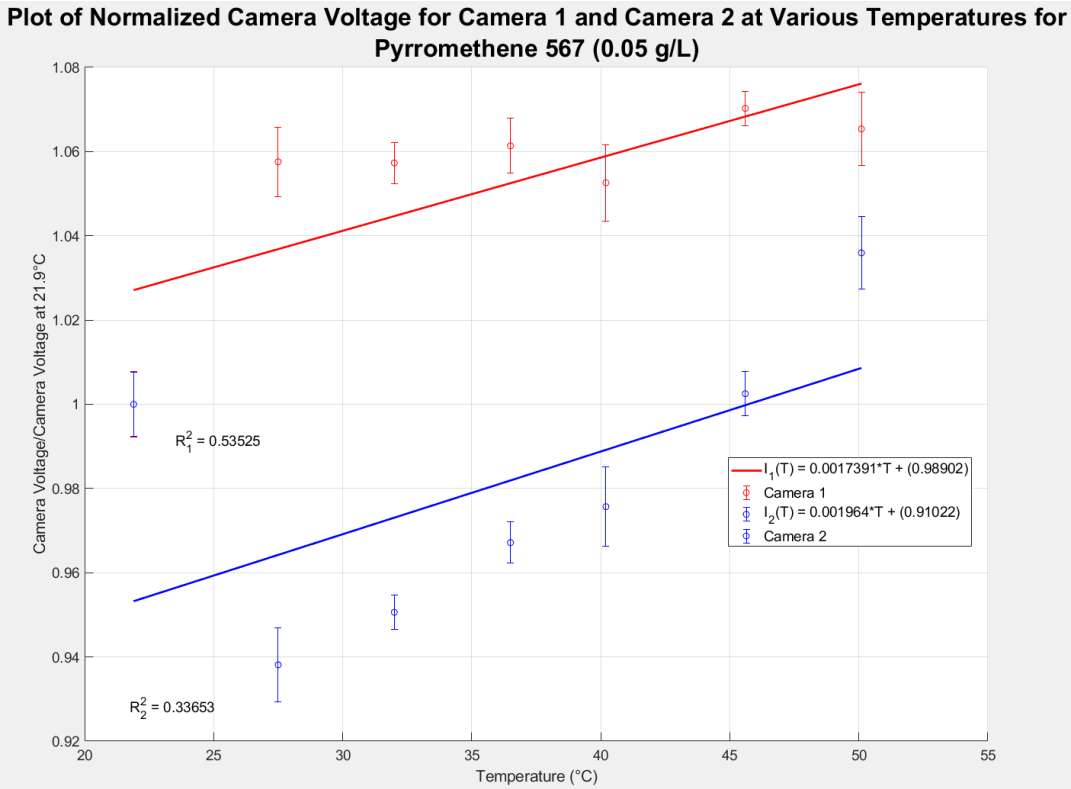

Figure 37. Normalized camera voltage versus temperature for both cameras for pyrromethene 567. Error bars represent a single standard deviation.

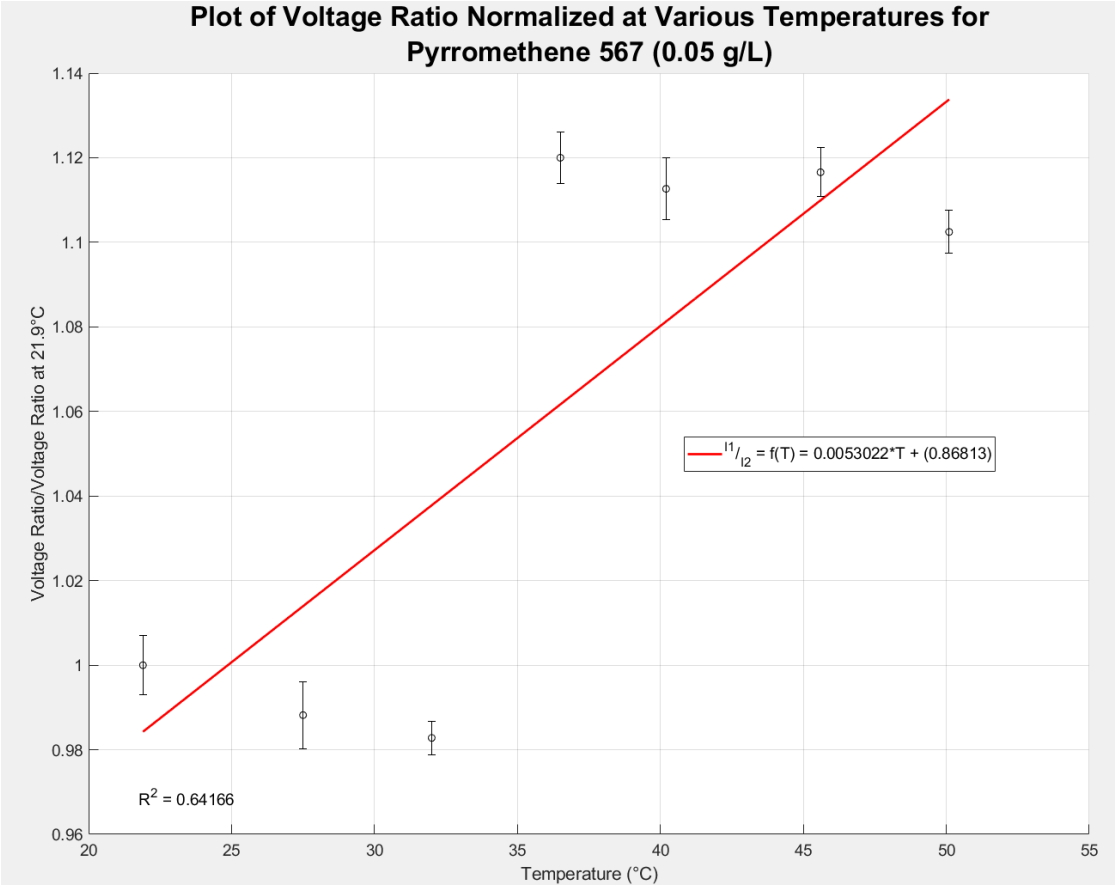

Figure 38. Normalized voltage ratio plot for pyrromethene 567. Error bars represent a single standard deviation.

3.3.6 Pyrromethene 597-8C9 in Silicon Device

Pyrromethene 597-8C9 was expected to be much more temperature sensitive than pyrromethene 567 [22]. Based on the camera voltage plot shown in Figure 39 and the normalized camera voltage plot shown in Figure 40, these expectations were met. The results for camera 1 and camera 2 indicate a temperature sensitivity of -1.25%/K and -1.49%/K for each camera, respectively. As camera 2 has a considerably higher camera voltage than camera 1 with this dye, the camera voltages determined for camera 2 will be used when determining the temperature of the n-decane solution.

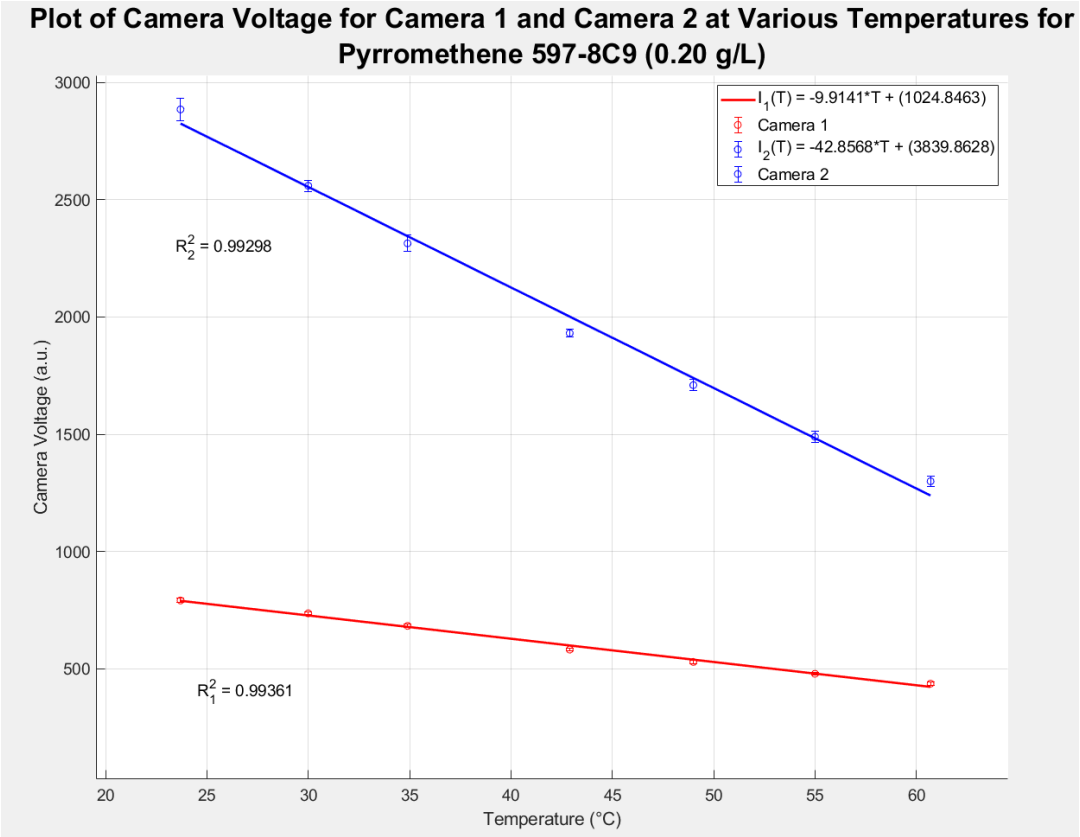

Figure 39. Camera voltage versus temperature for both cameras for pyrromethene 597-8C9. Error bars represent a single standard deviation.

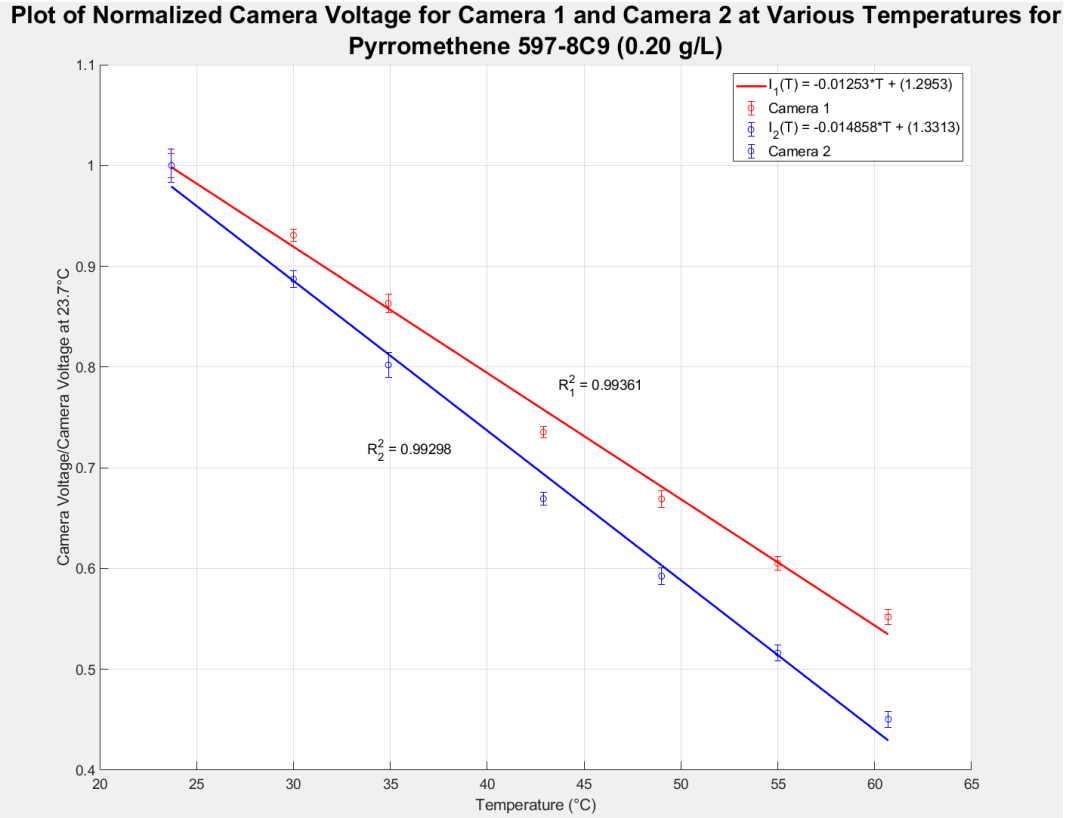

Figure 40. Normalized camera voltage versus temperature for both cameras for pyrromethene 597-8C9. Error bars represent a single standard deviation.

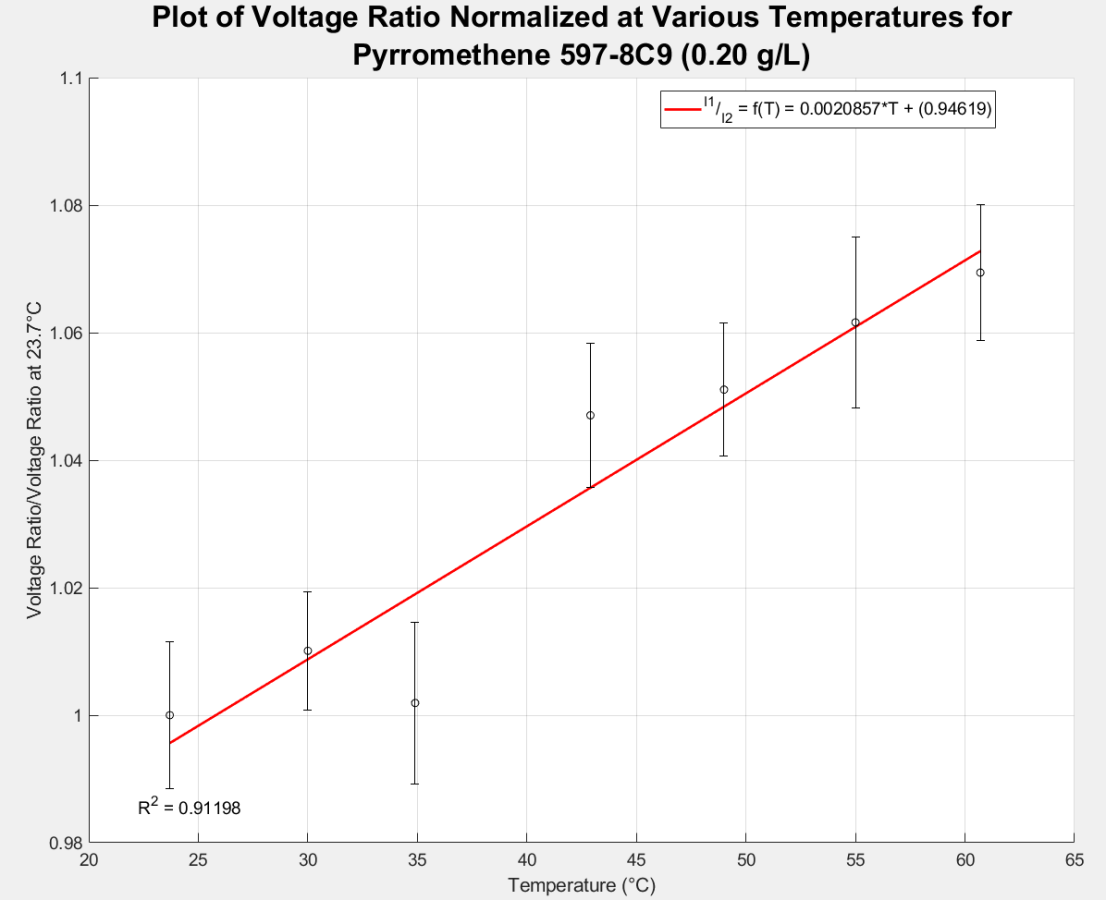

Figure 41. Normalized voltage ratio plot for pyrromethene 597-8C9. Error bars represent a single standard deviation.

3.4 Ratiometric LIF

As the ratiometric technique was found to give poor results, as seen in all of the normalized voltage ratio plots in section **3.3 Single Dye Temperature Calibration**, the ratiometric technique was not used for further experiments. Different filters, devices, dye concentrations, laser parameters (pulse rate, Q-switch delay), camera parameters (exposure time, capturing frequency), ways of determining the ratio between the two cameras, and many different lines of MATLAB code were all tried in hopes of getting the

ratiometric technique working well. Ultimately, the ratiometric proved to not be feasible with the current experimental setup and even just mixing two dyes together was found to give poor results due overlaps in their emission and absorption spectra. While LIF thermometry is most commonly done now using a ratiometric technique to reduce the effects of laser intensity variation, results for individual cameras proved far better than the ratiometric technique. It is not completely known why the ratiometric proved to give such poor results with this experimental setup.

3.6 Single-Phase LIF Thermometry Testing of Temperature Gradient

3.6.1 Pyrromethene 597-8C9 in Silicon Device

To test the results of the single-dye temperature calibration performed earlier, a temperature gradient was setup using the two water baths at a constant temperature. The hot bath was set to 65 °C and cold bath was set to 17 °C, resulting in the 66.8 °C and 21.8 °C temperatures used in the simulation. The silicon device was used rather than the PDMS device due to the issue of PDMS adsorption of pyrromethene 597-8C9. Pyrromethene 597-8C9 was selected as the only temperature-sensitive n-decane-soluble fluorescent dye of those tested. Pyrromethene 597-8C9 at a concentration of 0.20 g/L was found to have an approximate temperature sensitivity of $-1.33\%/\mathrm{K}$ in ***3.3.6 Pyrromethene 597-8C9 in Silicon Device*** for the wavelength band captured by camera 2. Camera 2 was chosen for this fluorescent dye due to having a more intense fluorescence response than in the wavelength band captured by camera 1.

Three sets of 15 images were taken throughout the silicon device, moving in a straight line from the cold bath side of the device to the hot bath side of the device. Thermal paste was placed between the silicon device and the walls of the aluminum block to assist with conduction heat transfer. The silicon device's top surface was insulated using a layer of polyurethane foam. After the device reached steady-state, the measured temperatures between where the silicon device contacts each side of the aluminum block were 21.8 °C and 66.8 °C for the cold bath and hot bath side, respectively. The camera voltage plot versus distance from the cold side of the PDMS device is shown below in Figure 42. Camera voltage decreased as the distance from the cold side of the PDMS device increased. The temperature determined from the camera voltage is plotted in Figure 43, indicating that the temperature measured increased moving towards the hot side of the PDMS. Agreement between the temperature values measured and the results of the simulation shown in ***A.1.2 Temperature Gradient Applied to PDMS Device*** are reasonably close.

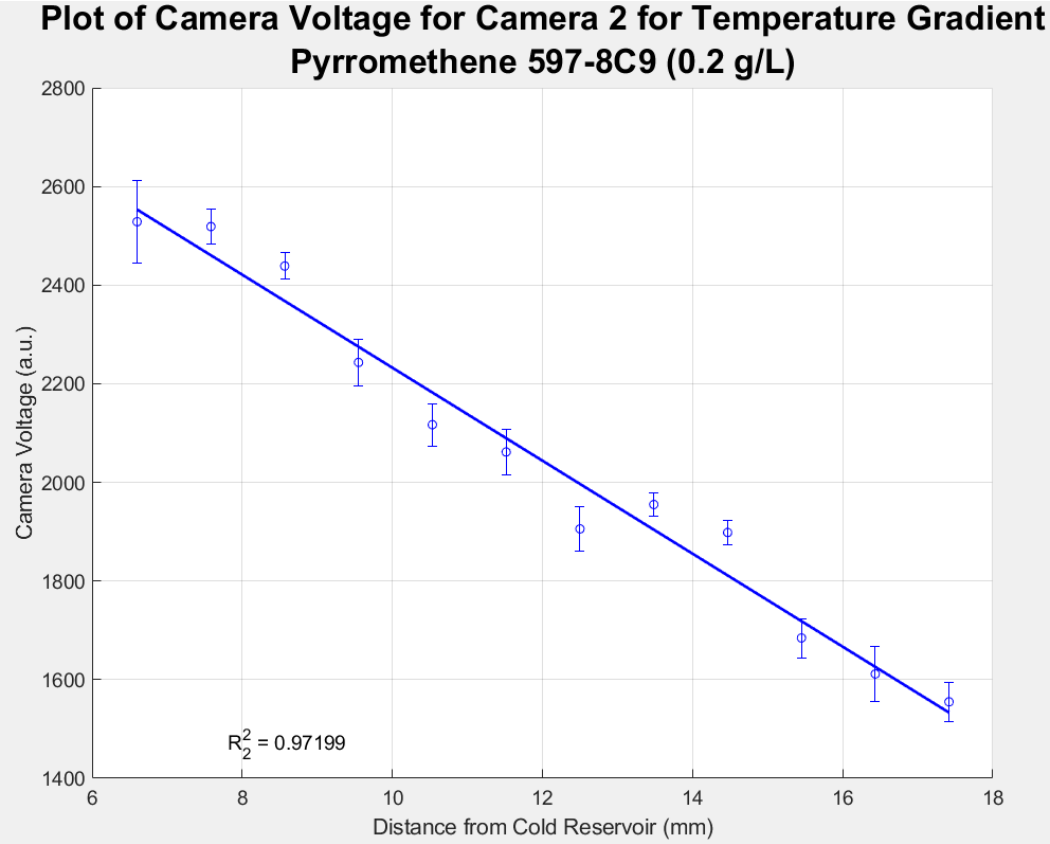

Figure 42. Camera voltage versus distance from the cold side of the PDMS device with an applied temperature gradient for pyrromethene 597-8C9 at a concentration of 0.20 g/L.

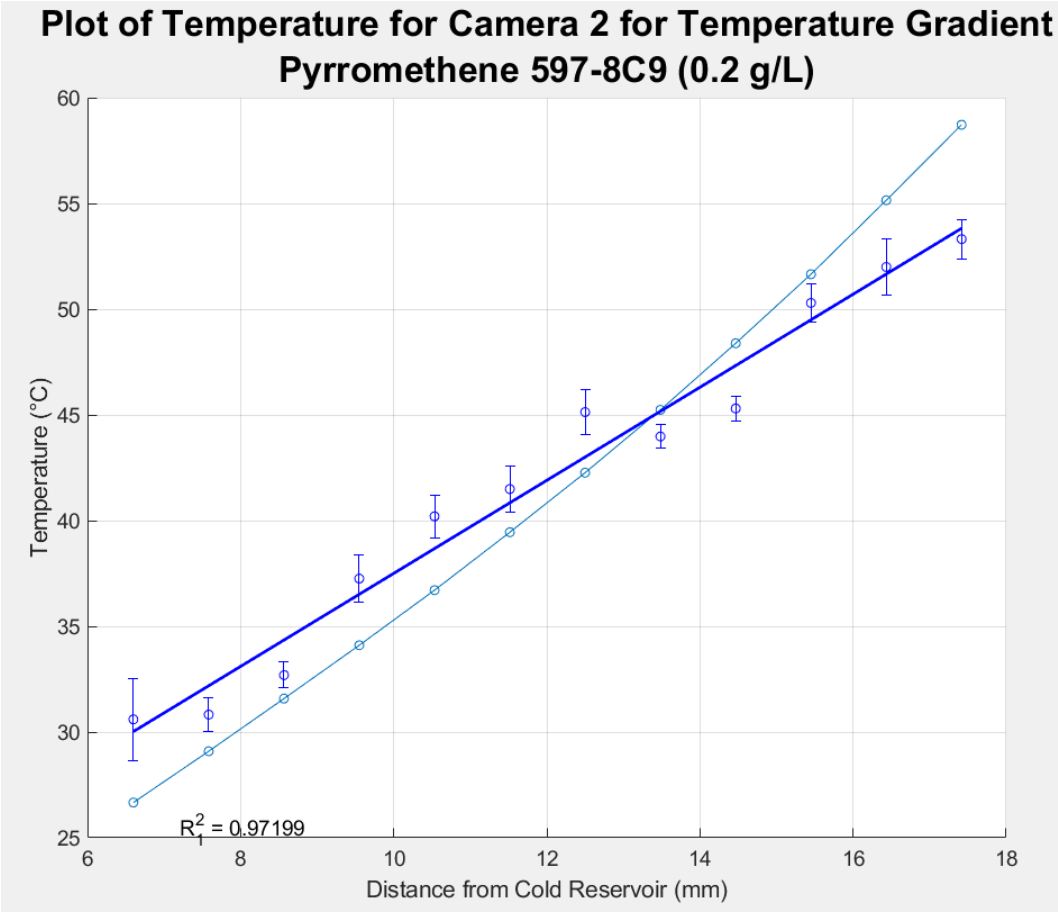

Figure 43. Temperature versus distance from the cold side of the PDMS device with an applied temperature gradient for pyrromethene 597-8C9 at a concentration of 0.20 g/L.

3.6.2 Fluorescein Disodium Salt Hydrate in PDMS Device

Fluorescein disodium salt hydrate at a concentration of 2.0 g/L was tested using an applied temperature gradient. Thermocouples were used to verify that the temperature on the walls of the PDMS device matched the 66.8 °C and 21.8 °C temperature values used in the simulation boundary conditions. As fluorescein disodium salt hydrate's fluorescence response is primarily captured by camera 1 (547.5-572.5 nm band), only camera 1 was used to measure the temperature within the device. The camera voltage values are plotted

below in Figure 44, and show that camera voltage increases moving away from the cold side of the PDMS device. This agrees with the expected result, as fluorescein disodium salt hydrate's fluorescence intensity increases with temperature and temperatures should increase as measurements move away from the cold side of the device. The corresponding measured temperature, shown below in Figure 45, shows some agreement with the simulation results. In general, the temperature measurements agree with measurements but are as much as about 4 °C above the simulated data.

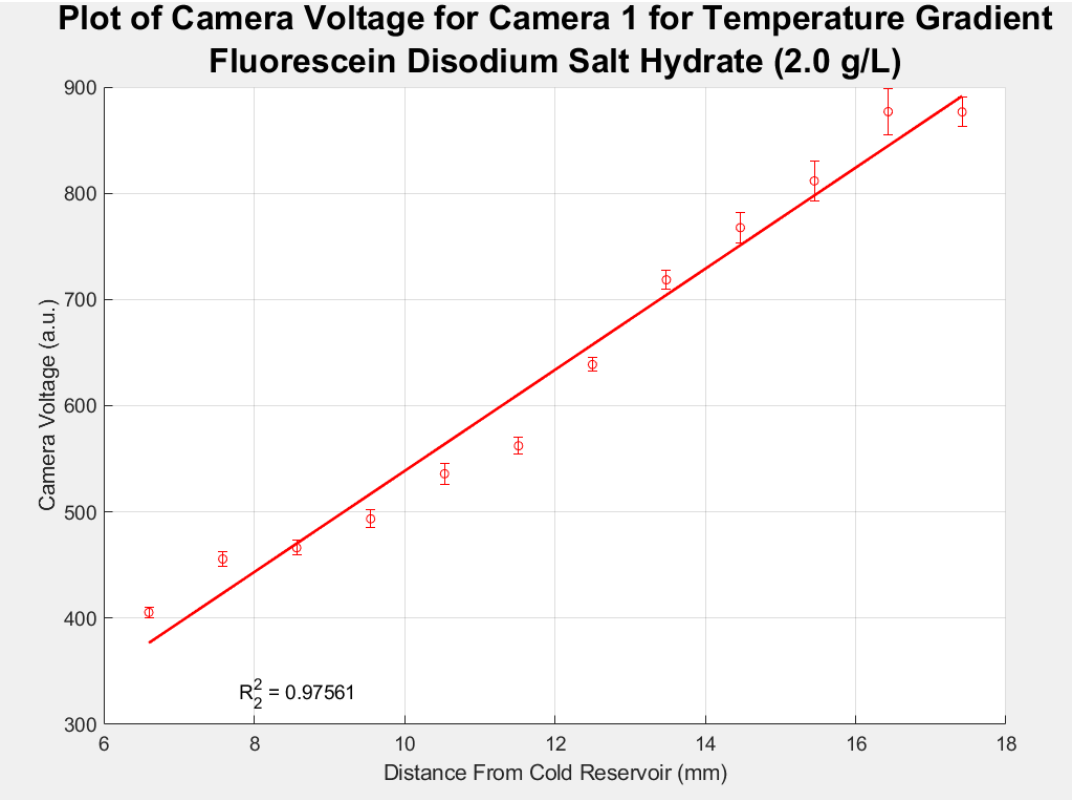

Figure 44. Camera 1 voltage versus measurement location for fluorescein disodium salt hydrate at a various locations within the PDMS device exposed to a temperature gradient.

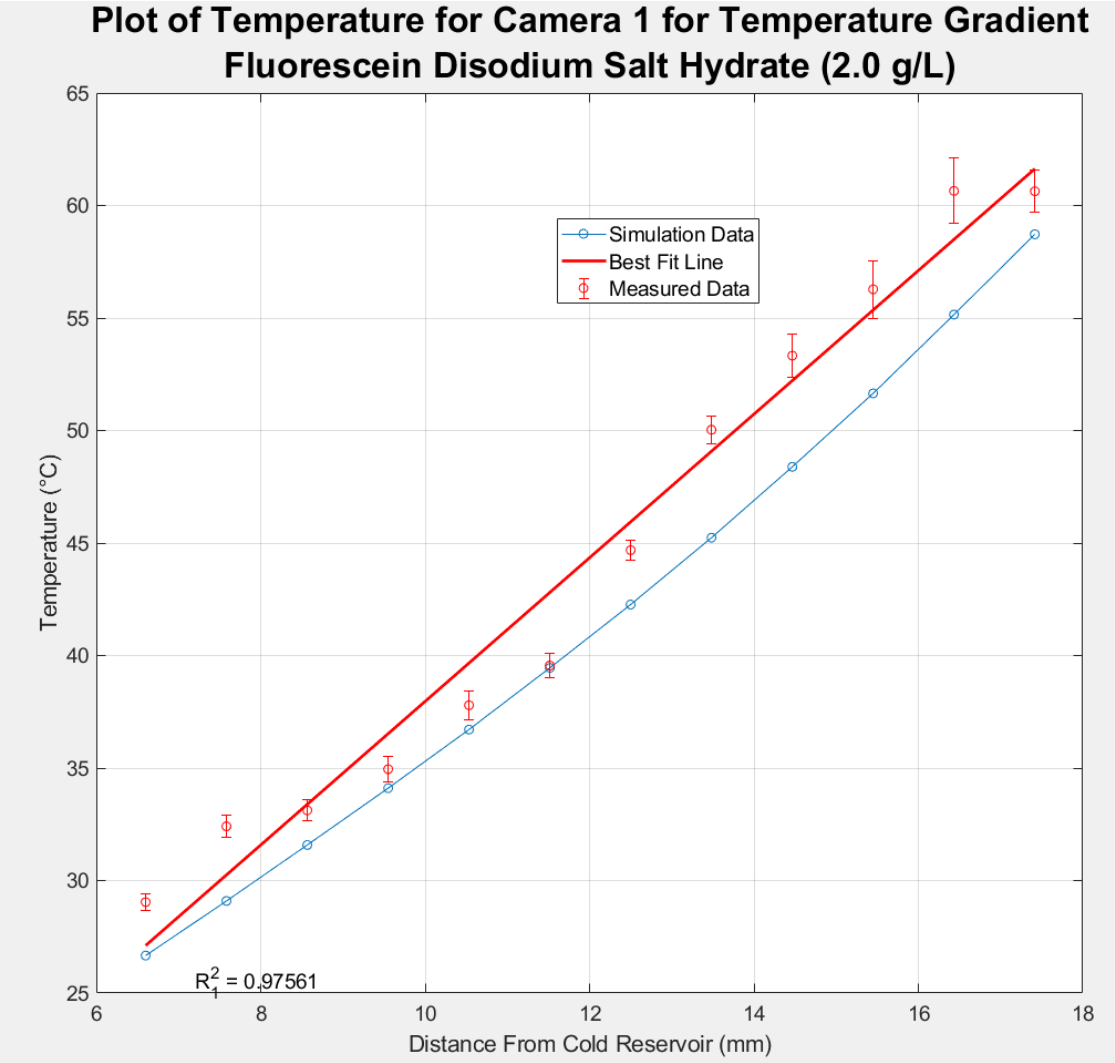

Figure 45. Temperature versus measurement location for fluorescein disodium salt hydrate at a various locations within the PDMS device exposed to a temperature gradient.

To test the effects of the location of the measurement, the measured temperature was determined in 100 different 10 x 10 pixel windows at each location where measurements were taken in PDMS device. Results were averaged for 3 sets of 15 images all taken at the same locations. The plot shown in Figure 46 shows the results in the location closest to the cold reservoir and closest to the hot reservoir used to establish the temperature gradient

within the PDMS device. Temperatures near the cold reservoir were all generally higher than the simulated temperature value of 26.667 °C and are about ±3 °C from an average measured temperature of roughly 29 °C. For the contour plot near the hot reservoir, temperatures varied from 52 °C to 66 °C within a $(100\ \mu m)^2$ region of the device. Both contour plots showed a similar trend of having the average measured temperature occur roughly at a distance 50-60 µm in both directions, the location where the temperature has been measured for all experiments.

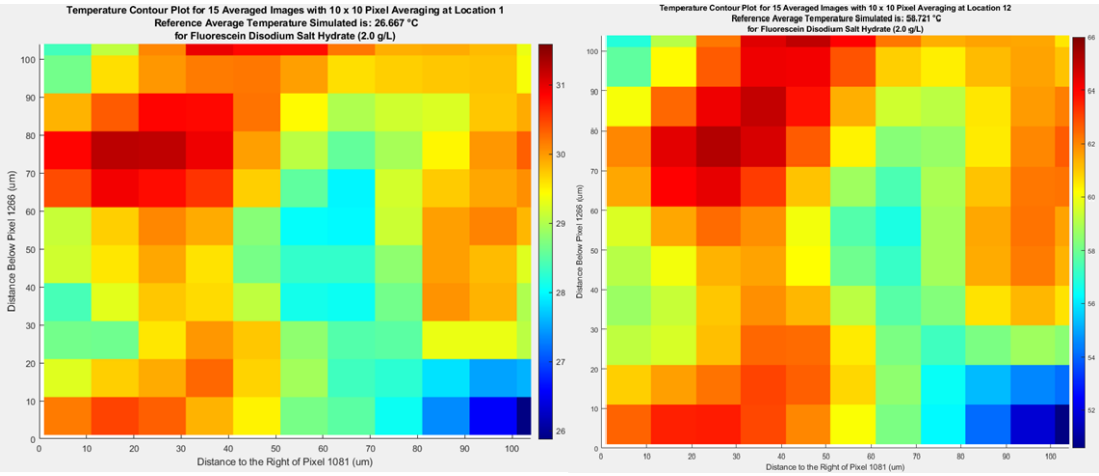

Figure 46. Temperature contour plot closest to the coldest part of the PDMS device, plot is approximately 50 µm in all directions from the location measured for the measured temperature plot shown earlier. Left plot is the location in the device closest to the cold reservoir, right plot is the location in the device closest to the hot reservoir. Scale on right is temperature (°C).

Location within the image was found to greatly influence the results for the measured temperature and proved difficult to correct for. The variation of laser light within the PDMS device is dependent on the location within the image, as the region located at an x distance of 20-30 µm and a y distance of 70-80 µm has been the hottest for all of the temperature

contour plots. The location dependence appears to be due to issues with laser intensity uniformity, which should be correctable using a ratiometric LIF technique instead. The ratiometric LIF technique would account for this by taking the ratio of the fluorescence intensities from both cameras which should cause small fluctuations in fluorescence intensity to be negligible, as each camera should be roughly equally affected.

3.7 Multi-Phase LIF Thermometry Testing of Temperature Gradient

To test LIF thermometry applied to multiphase flow, the silicon porous micromodel was first filled with pyrromethene 597-8C9 at a concentration of 0.2 g/L before fluorescein disodium salt hydrate at a concentration of 2.0 g/L was injected into the device, producing regions containing either fluid. Since pyrromethene 597-8C9 has a significantly higher fluorescence within the wavelength band captured by camera 2, which fluid the camera is looking at can be easily distinguished visually or by measuring the fluorescence intensity for camera 2. Responses for both dyes are similar for camera 1, so the intensity value measured by camera 2 was used to distinguish the different fluid phases and apply the correct temperature correlation. An example image taken by both cameras is shown below in Figure 47, with the brighter spots in the image taken by camera 2 being regions containing pyrromethene 597-8C9. In images captured by camera 1, the two fluorescent dyes are indistinguishable, as the fluorescence intensity of both dyes is about the same within the wavelength band captured by camera 1.

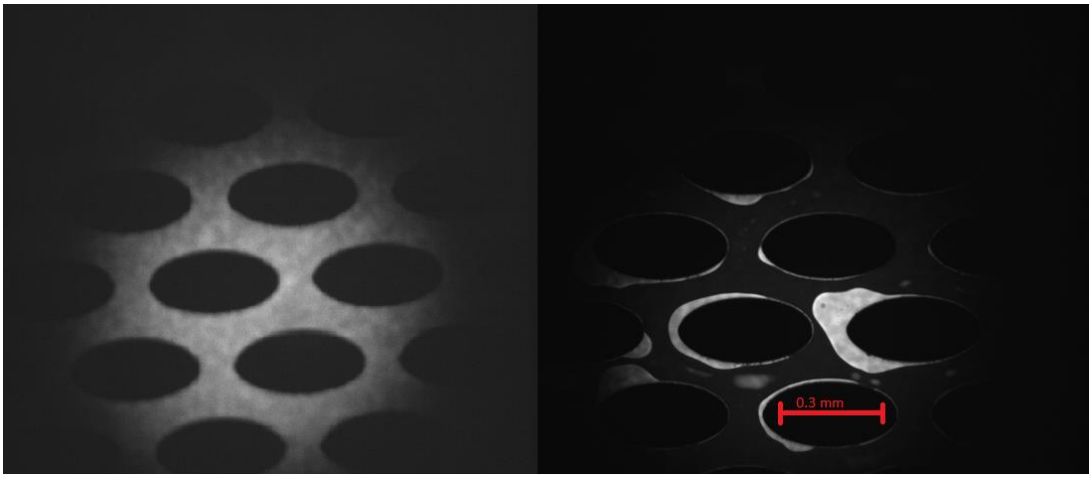

Figure 47. Image taken of silicon device containing pyrromethene 597-8C9 (0.2 g/L) and fluorescein disodium salt hydrate (2.0 g/L). Image of left is from camera 1 and the image on the right is from camera 2. Bright spots on the right side are locations containing pyrromethene 597-8C9.

Measurements were taken of both fluids at approximately the same locations used for the single-phase experiments. The camera was panned to the left or right within the channel to find regions where each fluid could be found at the same vertical distance from the cold and hot reservoir, meaning each measurement taken at roughly the same location should have approximately the same temperature. The voltages for each dye are plotted below in Figure 48 and Figure 49, and show similar trends to those found in 3.6 Single-Phase LIF Thermometry Testing of Temperature Gradient. Some points were omitted from the pyrromethene 597-8C9 plot due to the voltages being clear outliers, likely due to their not being a sufficient large amount of the n-decane to get a good measurement in that location. The temperature measurement data is shown in Figure 50 and Figure 51 for sulforhdamine disodium salt hydrate and pyrromethene 597-8C9, respectively. Measured temperatures were generally within 4 °C of the simulated temperature but still showed low standard

deviations. Measurements for temperatures appear to be fairly precise but also have poorer accuracy than would be desirable for a temperature measurement technique.

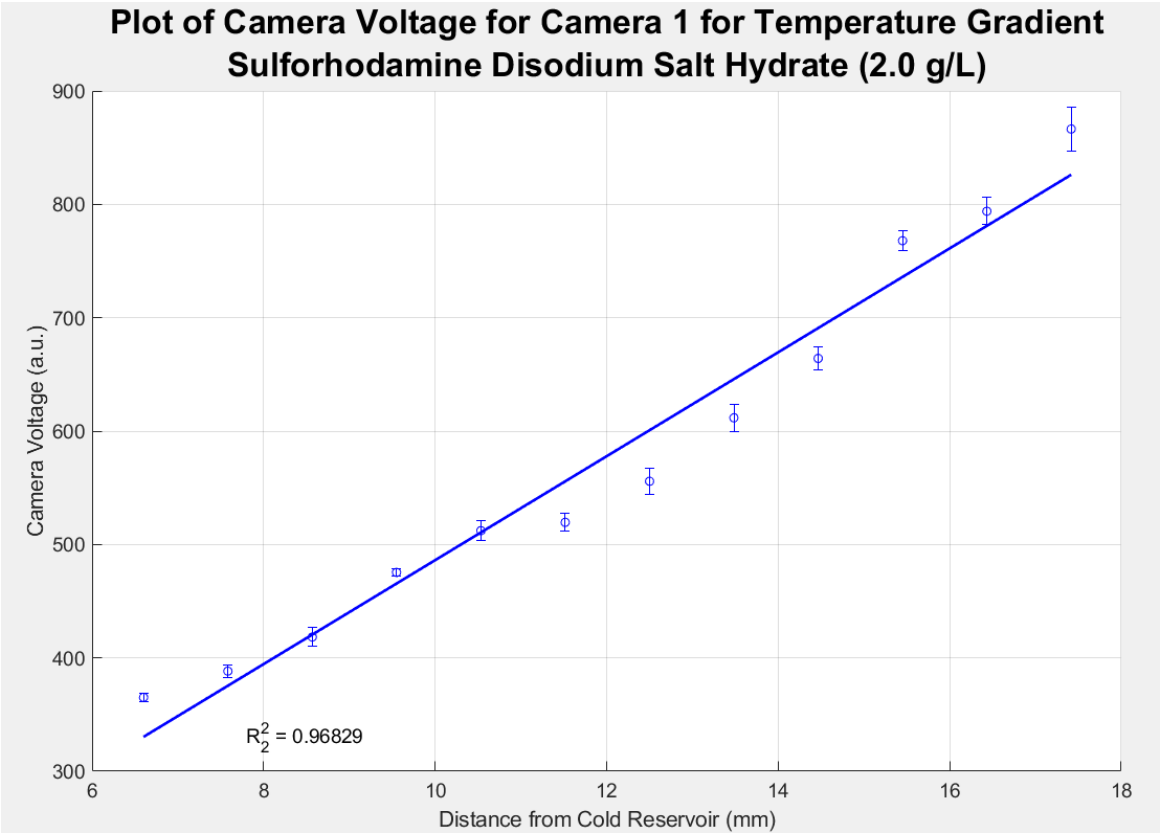

Figure 48. Camera 1 voltage versus distance from the cold reservoir plot for fluoroscein disodium salt hydrate (2.0 g/L) during the multiphase flow experiment.

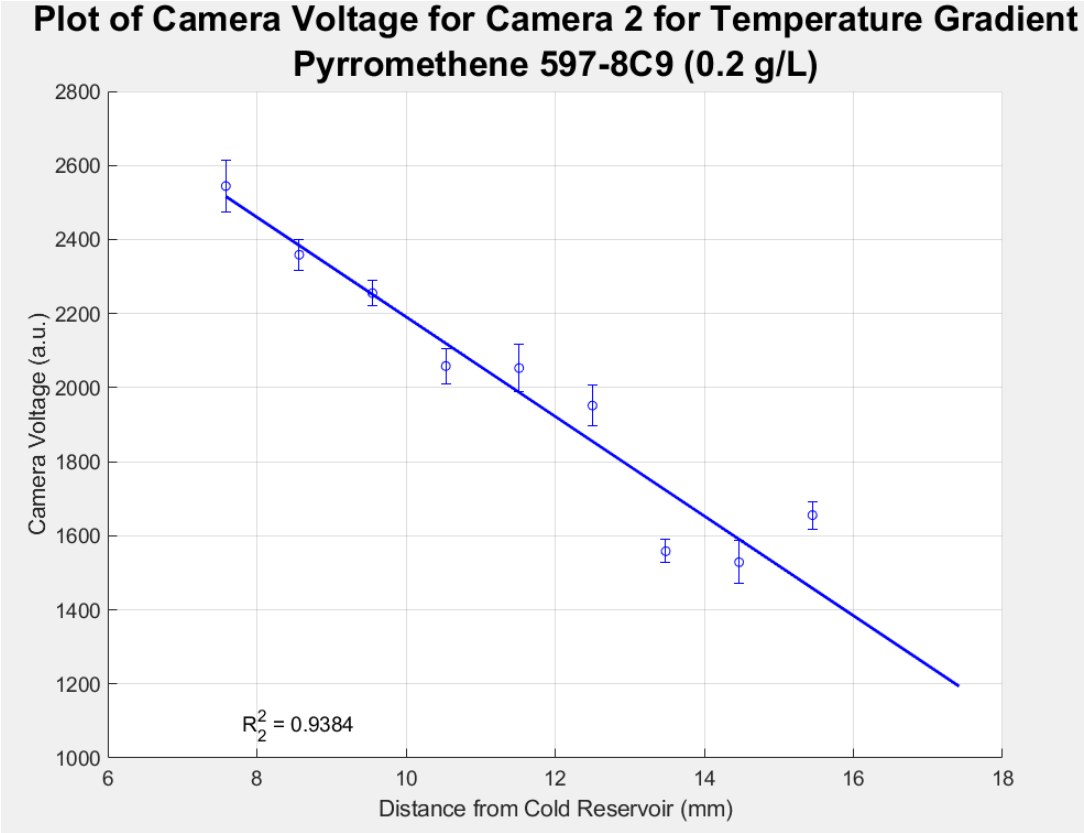

Figure 49. Camera 2 voltage versus distance from the cold reservoir plot for pyrromethene 597-8C9 (0.2 g/L) during the multiphase flow experiment.

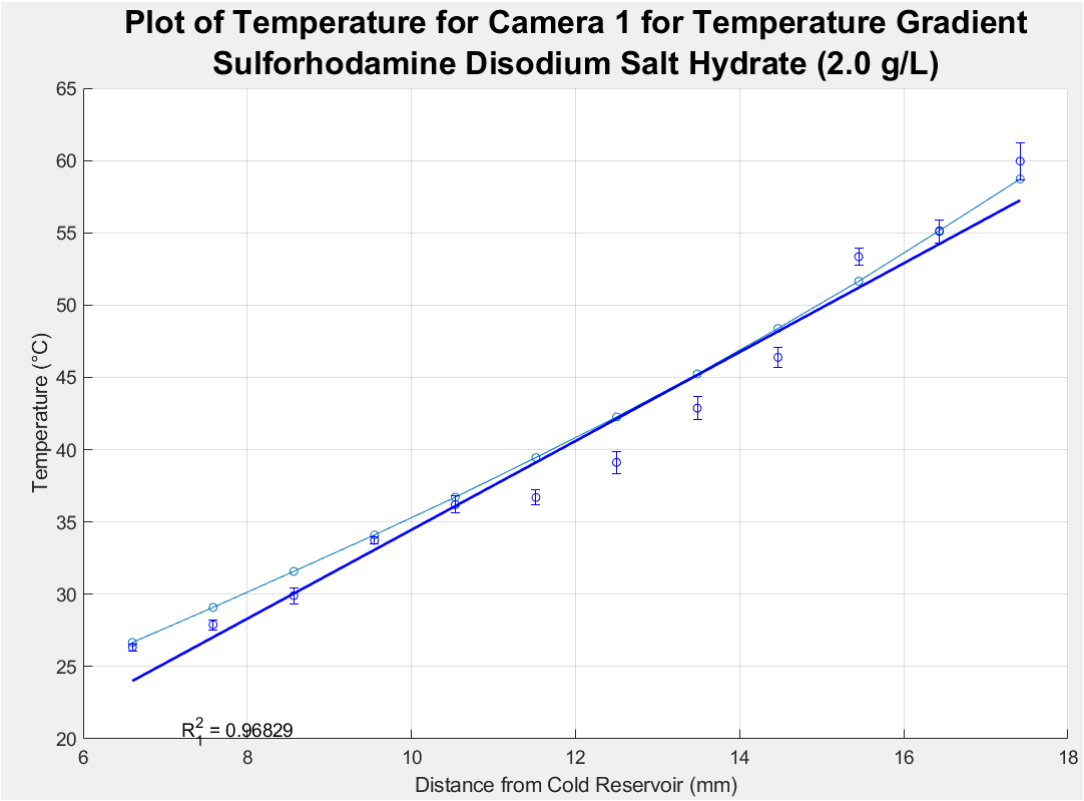

Figure 50. Temperature versus distance from the cold reservoir plot for fluorescein disodium salt hydrate (2.0 g/L) during the multiphase flow experiment. Blue points and line are the measured data and best fit line, light blue line is the simulation data.

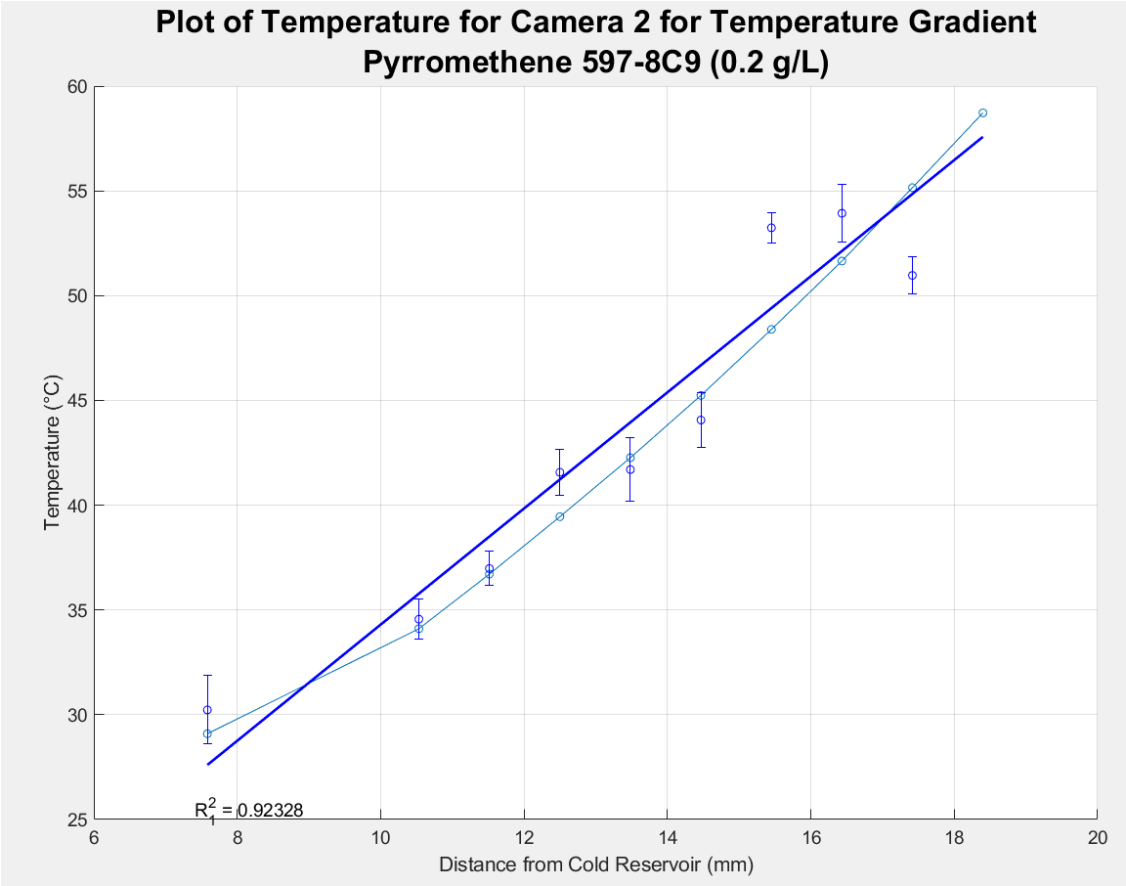

Figure 51. Temperature versus distance from the cold reservoir plot for pyrromethene 597-8C9 (0.2 g/L) during the multiphase flow experiment. Blue points and line are the measured data and best fit line, light blue line is the simulation data.

The root-mean-square error between the simulated data and the measured data are shown in Table 2 and Table 3 for fluorescein disodium salt hydrate and pyrromethene 597-8C9, respectively. The root-mean-square error for fluorescein disodium salt hydrate and pyrromethene 597-8C9 are 1.62 °C and 4.30 °C, respectively. Results for pyrromethene 597-8c9 were as much as 5 °C from the simulated temperature value while results for fluorescein disodium salt hydrate were as much as 2.5 °C from the simulated temperature value. Assuming the results for the simulation are the correct temperature, this technique

has a considerably lower accuracy than a thermocouple is capable of. The result for fluorescein disodium salt hydrate is reasonably close to expectations for single-dye LIF, which has a typical accuracy of ±1.5 K [11].

Table 2
Root mean squared error calculations for fluorescein disodium salt hydrate (2.0 g/L) for the multiphase flow experiment.

Fluorescein Disodium Salt Hydrate 2.0 g/L (Multiphase)		
Simulation Data (°C)	Experimental Dye (°C)	Squared Error ($°C^2$)
26.67	26.34	0.11
29.09	27.89	1.44
31.58	29.91	2.81
34.11	34.10	0.00
36.71	34.68	4.11
39.45	37.09	5.58
42.27	40.75	2.29
45.24	44.20	1.08
48.39	50.93	6.49
51.65	52.71	1.11
55.15	57.47	5.35
58.72	59.84	1.26
Root Mean Squared Error (°C)		1.62

Table 3
Percent difference calculations for pyrromethene 597-8C9 (0.2 g/L) for the multiphase flow experiment.

Pyrromethene 597-8C9 0.2 g/L (Multiphase)		
Simulation Data (°C)	Experimental Dye (°C)	Squared Error ($°C^2$)
26.67	30.23	12.70
29.09		
31.58	34.56	8.87
34.11	36.98	8.24
36.71	41.57	23.63
39.45	41.70	5.06
42.27	44.06	3.23
45.24	53.24	64.00
48.39	53.93	30.71
51.65	50.97	0.47
55.15	60.48	28.37
58.72		
Root Mean Squared Error (°C)		4.30

4 CONCLUSION

Table 4
Summary table for temperature calibration and multiphase flow testing of each fluorescent dye tested.

Fluorescent Dye	Fluid	Concentration (g/L)	Primary Camera	Temperature Sensitivity (%/K)	Multiphase Root Mean Square Error
Eosin Y	Water	0.8	Camera 1	1.49	Not Tested
Sulforhodamine 101	Water	0.4	Camera 2	1.23	Not Tested
Sulforhodamine B Sodium Salt	Water	0.5	Camera 2	-1.22	Not Tested
Fluorescein Disodium Salt Hydrate	Water	2.0	Camera 1	4.53	1.62
Pyrromethene 567	n-decane	0.05	Camera 1	0.17	Not Tested
Pyrromethene 597-8C9	n-decane	0.2	Camera 2	-1.49	4.30

Through this research it was found that LIF thermometry is a viable technique for measuring temperature distributions within porous media micromodels in principle. Results for the spatial resolution of this technique compared favorably to thermocouples probes which are at minimum 13 micrometers in size [7]. With this experimental setup, temperature measurements for applied temperature gradients in multiphase flow achieved a mean root-mean-square-error of at most 4.30 °C from the simulated data in 5.2 µm by 5.2 µm square regions within the fluid domain but accuracy could be greatly improved. Results for fluorescein disodium salt hydrate produced a much lower root-mean-square error of 1.62 °C. Measurement accuracy is considerably lower than that of a Type T thermocouple, which has an accuracy of ±1 °C or 0.75% of the measurement [9]. Using 3 sets of 15 images

for a temporal resolution of one measurement every 4.5 seconds with a measurement window of $(5.2\ \mu m)^2$ was found to produce a coefficient of variance of less than 1%. This makes the temporal resolution considerably worse than a Type T thermocouple (response time typically <1 second), as the response time was at least 4 times slower [8]. Measurements were found to vary significantly based on the location the measurement was taken within the device, despite the coefficient of variance among repeated measurements taken at any single location being low.

The fluorescent dyes, eosin Y, sulforhodamine 101, sulforhodamine B sodium salt, fluorescein disodium salt hydrate, pyrromethene 567, and pyrromethene 597-8C9 were all tested as potential candidates for temperature measurement. Two cameras were used, with wavelength filters that cause each camera to capture 547.5 to 572.5 nm and 610 to 640 nm, respectively. For this set of filters, fluorescein disodium salt hydrate and pyrromethene 597-8C9 were found to have the highest temperature sensitivities, at 4.53%/K and -1.33%/K for camera 1 and camera 2, respectively.

Fluorescein disodium salt hydrate and pyrromethene 597-8C9 were found to be the best option for temperature measurement for the optical setup used in this research, as these dyes had highly temperature-sensitive emission spectra for the wavelengths captured by the filters. Additionally, these regions did not significantly overlap, allowing for the correct temperature correlations to be applied based on the fluorescence intensity measured by each camera. The ratiometric LIF thermometry technique used by many other researchers was unable to be successfully utilized because the temperature sensitivity and the goodness

of fit between temperature and the fluorescence response were found to be significantly worse than expected. Instead, non-ratiometric LIF thermometry technique had to be used, where the fluorescent response of only one camera is used for each dye and the effect of laser light intensity variation is not accounted for. Variations in laser intensity can lead to false conclusions that the temperature being measured changed significantly.

While the LIF thermometry technique was shown to work in principle for multiphase flow in porous media, the accuracy of the technique must be greatly improved for this to be a practical technique. One factor hindering the accuracy of this technique is the variation in the fluorescence intensity and captured camera voltage depending on the location the temperature is measured within the device. Most research papers on LIF thermometry use a ratiometric technique to account for the variation in laser light intensity, helping to reduce this type of error. However, efforts to utilize the ratiometric technique proved unsuccessful with the current experimental setup for an unknown reason. For all experimental data used in the preceding sections, at least 3 trials of the same experiment were done and a similar degree of accuracy was found in all 3 trials. Possibly the accuracy could be improved by obtaining a background image prior to every measurement in each location and dividing the intensity values measured with the dye by the background image, similar to the procedure followed by the ratiometric technique. However, this approach is very impractical compared to how the ratiometric technique is normally performed, which uses the fluorescence captured by the other camera to account for intensity variation in the device.

The best approach to improve LIF thermometry would be the ratiometric technique. For multiphase flow in porous media, LIF thermometry was found to be a viable option for temperature measurement but the accuracy of the technique was found to be too coarse for practical temperature measurements inside microscale porous multiphase flow configurations. A spatial resolution of 5.2 by 5.2 micrometers and a temporal resolution of one measurement every 4.5 seconds seems achievable in terms of the coefficient of variance, but accuracy of the technique must be further improved to be less susceptible to sources of error.

REFERENCES

[1] J. P. Davim and R. K. Singh, Eds. "Advances in MEMS and microfluidic systems" in *Advances in Mechatronics and Mechanical Engineering (AMME)*, IGI Global, 2023.

[2] M. Wu, F. Xiao, R. M. Johnson-Paben, S. T. Retterer, X. Yin, and K. B. Neeves, "Single- and two-phase flow in microfluidic porous media analogs based on Voronoi tessellation," *Lab Chip*, vol. 12, no. 2, pp. 253–261, 2012, doi: 10.1039/C1LC20838A.

[3] H. Shi, N. Di Miceli Raimondi, E. Cid, M. Cabassud, and C. Gourdon, "Temperature field acquisition by planar laser induced fluorescence using the two-color/two-dye technique for liquid flows in a millimetric zigzag channel," *Chemical Engineering Journal*, vol. 426, p. 131460, Dec. 2021, doi: 10.1016/j.cej.2021.131460.

[4] A. Faghri and Y. Zhang, *Fundamentals of Multiphase Heat Transfer and Flow*. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-22137-9.

[5] T.-C. Hung and W.-M. Yan, "Thermal performance enhancement of microchannel heat sinks with sintered porous media," *Numerical Heat Transfer, Part A: Applications*, vol. 63, no. 9, pp. 666–686, May 2013, doi: 10.1080/10407782.2013.751778.

[6] H. J. Kim, K. D. Kihm, and J. S. Allen, "Examination of ratiometric laser induced fluorescence thermometry for microscale spatial measurement resolution," *International Journal of Heat and Mass Transfer*, vol. 46, no. 21, pp. 3967–3974, Oct. 2003, doi: 10.1016/S0017-9310(03)00243-6.

[7] Y. Suh, "Characterization of microscopic thermofluidic transport for development of porous media used in phase-change devices," M.S. Thesis, Department of Mechanical and Aerospace Engineering, University of California, Irvine, Irvine, CA, 2020.

[8] "RTDs & thermocouples frequently asked questions," Omega Engineering, Inc. Accessed: Nov. 13, 2023. [Online]. Available: https://www.te.com/usa-en/faqs/rtds-thermocouples-faqs.html

[9] "Thermocouple types," Omega Engineering, Inc. Accessed: Nov. 13, 2023. [Online]. Available: https://www.omega.com/en-us/resources/thermocouple-types

[10] "Thermocouple response time," Omega Engineering, Inc. Accessed: Nov. 13, 2023. [Online]. Available: https://www.omega.com/en-us/resources/thermocouples-response-time

[11] J. Sakakibara and R. J. Adrian, “Whole field measurement of temperature in water using two-color laser induced fluorescence,” *Experiments in Fluids*, vol. 26, no. 1–2, pp. 7–15, Jan. 1999, doi: 10.1007/s003480050260.

[12] B. Peterson, E. Baum, B. Böhm, V. Sick, and A. Dreizler, “Evaluation of toluene LIF thermometry detection strategies applied in an internal combustion engine,” *Appl. Phys. B*, vol. 117, no. 1, pp. 151–175, Oct. 2014, doi: 10.1007/s00340-014-5815-0.

[13] D. Ross, M. Gaitan, and L. E. Locascio, “Temperature measurement in microfluidic systems using a temperature-dependent fluorescent dye,” *Anal. Chem.*, vol. 73, no. 17, pp. 4117–4123, Sep. 2001, doi: 10.1021/ac010370l.

[14] P. Chamarthy, S. V. Garimella, and S. T. Wereley, “Measurement of the temperature non-uniformity in a microchannel heat sink using microscale laser-induced fluorescence,” *International Journal of Heat and Mass Transfer*, vol. 53, no. 15–16, pp. 3275–3283, Jul. 2010, doi: 10.1016/j.ijheatmasstransfer.2010.02.052.

[15] J. A. Sutton, B. T. Fisher, and J. W. Fleming, “A laser-induced fluorescence measurement for aqueous fluid flows with improved temperature sensitivity,” *Exp. Fluids*, vol. 45, no. 5, pp. 869–881, Nov. 2008, doi: 10.1007/s00348-008-0506-4.

[16] M. Koegl, C. Weiß, and L. Zigan, “Fluorescence spectroscopy for studying evaporating droplets using the dye eosin-Y,” *Sensors*, vol. 20, no. 21, p. 5985, Oct. 2020, doi: 10.3390/s20215985.

[17] V. K. Natrajan and K. T. Christensen, “Two-color laser-induced fluorescent thermometry for microfluidic systems,” *Meas. Sci. Technol.*, vol. 20, no. 1, p. 015401, Jan. 2009, doi: 10.1088/0957-0233/20/1/015401.

[18] A. Labergue, A. Delconte, and F. Lemoine, “Study of the thermal mixing between two non-isothermal sprays using combined three-color LIF thermometry and phase Doppler analyzer,” *Exp. Fluids*, vol. 54, no. 6, p. 1527, Jun. 2013, doi: 10.1007/s00348-013-1527-1.

[19] J. J. L. Higdon, “Multiphase flow in porous media,” *J. Fluid Mech.*, vol. 730, pp. 1–4, Sep. 2013, doi: 10.1017/jfm.2013.296.

[20] “Fluorescence SpectraViewer,” ThermoFisher Scientific. Accessed: Dec. 06, 2022. Accessed: Nov. 13, 2023. [Online]. Available: https://www.thermofisher.com/order/fluorescence-spectraviewer#!/

[21] G. Blume, G. Mielke, J. Kohnert, R. Pörtner, and K. H. Trieu, “Development of a process for the manufacturing of SU-8 100 for the use in cell culture,” *Journal of Bioactive and Compatible Polymers*, vol. 33, no. 4, pp. 349–360, Jul. 2018, doi: 10.1177/0883911518765216.

[22] J. Vogt and P. Stephan, “Using microencapsulated fluorescent dyes for simultaneous measurement of temperature and velocity fields,” *Meas. Sci. Technol.*, vol. 23, no. 10, p. 105306, Oct. 2012, doi: 10.1088/0957-0233/23/10/105306.

[23] Y. N. Mishra, A. Yoganantham, M. Koegl, and L. Zigan, “Investigation of five organic dyes in ethanol and butanol for two-color laser-induced fluorescence ratio thermometry,” *Optics*, vol. 1, no. 1, pp. 1–17, Dec. 2019, doi: 10.3390/opt1010001.

[24] A. Banerjee, Y. Shuai, R. Dixit, I. Papautsky, and D. Klotzkin, “Concentration dependence of fluorescence signal in a microfluidic fluorescence detector,” *Journal of Luminescence*, vol. 130, no. 6, pp. 1095–1100, Jun. 2010, doi: 10.1016/j.jlumin.2010.02.002.

[25] P. Baj, P. J. K. Bruce, and O. R. H. Buxton, “On a PLIF quantification methodology in a nonlinear dye response regime,” *Exp. Fluids*, vol. 57, no. 6, p. 106, Jun. 2016, doi: 10.1007/s00348-016-2190-0.

[26] W. Chaze, O. Caballina, G. Castanet, and F. Lemoine, “The saturation of the fluorescence and its consequences for laser-induced fluorescence thermometry in liquid flows,” *Exp. Fluids*, vol. 57, no. 4, p. 58, Apr. 2016, doi: 10.1007/s00348-016-2142-8.

[27] L. Biasiori-Poulanges, S. Jarny, and H. El-Rabii, “Data on eosin Y solutions for laser-induced fluorescence in water flows,” *Data in Brief*, vol. 29, p. 105350, Apr. 2020, doi: 10.1016/j.dib.2020.105350.

[28] B. Thirouard, “Characterization and modeling of the fundamental aspects of oil transport in the piston ring pack of internal combustion engines,” Ph.D. thesis, Dept. of Mech. Eng., Massachusetts Inst. of Technol., Cambridge, MA, USA, 2001.

[29] A. Labergue, V. Deprédurand, A. Delconte, G. Castanet, and F. Lemoine, “New insight into two-color LIF thermometry applied to temperature measurements of droplets,” *Exp Fluids*, vol. 49, no. 2, pp. 547–556, Aug. 2010, doi: 10.1007/s00348-010-0828-x.

[30] J. W. Park, S. Na, M. Kang, S. J. Sim, and N. L. Jeon, “PDMS microchannel surface modification with teflon for algal lipid research,” *BioChip J.*, vol. 11, no. 3, pp. 180–186, Sep. 2017, doi: 10.1007/s13206-017-1302-0.

[31] Y. A. Çengel and A. J. Ghajar, *Heat and Mass Transfer: Fundamentals & Applications*, Fifth edition. New York, NY: McGraw Hill Education, 2015.

[32] H. Vasudevan, V. K. N. Kottur, and A. A. Raina, Eds., *Proceedings of International Conference on Intelligent Manufacturing and Automation: ICIMA 2020*. in Lecture Notes in Mechanical Engineering. Singapore: Springer Singapore, 2020. doi: 10.1007/978-981-15-4485-9.

APPENDIX A

A.1 Finite Element Analysis Simulation of Temperature Distribution within Device

A.1.1 Fixed Temperature Applied to PDMS Device

In order to evaluate the ability of LIF thermometry to be used to measure temperatures accurately, an experimental setup consisting of a rectangular aluminum block with a cutout rectangular portion large enough to hold the PDMS device and two separate flow channels was used. The side walls of the PDMS device nearly contact the insides of the aluminum device, with thermal paste added to improve the conduction heat transfer. Two water temperature baths are used to maintain the temperature of each the two longer sides of the aluminum block by pumping water at a specified temperature. Finite Element Analysis was done with Ansys to determine the expected temperature distributions of the fluid within the PDMS device in this setup. The temperatures on each side of the PDMS device where it contacts the aluminum walls were measured by thermocouples for two different cases. Case 1 where the temperatures on both sides of the device were measured to be 50 °C and case 2 where the hot side of the device was measured to be 63.8 °C and the cold side was measured to be 21.8 °C.

For the simulation of the device, a simplified geometry was used with the aluminum block removed and the temperature boundary conditions applied to directly to the sides of the glass and PDMS surfaces of the device. The convection coefficient used were based on the following equation for the Nusselt number $Nu = 0.27Ra_L^{1/4}$(meant for Rayleigh

numbers between 100,000 and 100,000,000,000) for the bottom surface [31]. The natural convection values resulting from these equations is shown below in Table 5

Table of calculations to determine the natural convection heat transfer coefficient for each surface of the PDMS device.. For the surface temperature used in these calculations, the value was updated to match the approximate average surface temperature on that face determined from the results of the FEA simulation. Only the glass bottom surface was assumed to be subject to natural convection, as the top surface of the PDMS device has been insulated with polyurethane foam and other surfaces are either in contact with the aluminum block holding the device or are small and have a very minor effect on the temperature of the PDMS device.

Table 5
Table of calculations to determine the natural convection heat transfer coefficient for each surface of the PDMS device.

Variable	Value	Unit
β	0.003411	K^{-1}
g	9.81	ms^{-2}
$T_{surface}$	40	°C
$T_{ambient}$	20	°C
L_c	0.006266	m
ν	1.51E-05	m^2s^{-1}
Pr	0.712	a.u.
Ra_L	517.0152	a.u.
Nu	1.287477	a.u.
k	0.02587	$Wm^{-1}K^{-1}$
h	5.315139	$Wm^{-2}K^{-1}$

A 0.5 mm mesh was used for most of the simulation, with a 0.05 mm mesh used for the water inside the device. The overall mesh is shown below in Figure 52 and the mesh used for the water inside the device is shown below in Figure 53. A summary of the mesh quality statistics is shown in Table 6. The mesh consists of 4,164,451 nodes and 2,544,068 elements, with the majority of these used for meshing the fine details of the water within the flow channel. The average aspect ratio of this mesh is 1.1615, with a maximum value of 39.394. Ideally, the aspect ratio should be in the range of 0-5 with a value >10 being considered bad [32]. The skewness has a maximum value of 0.9976 and an average value of 0.27787. Skewness is considered good between 0.25-0.75 and bad for values above 0.75 [32]. Element quality has an average value of 0.79609 and a minimum value of 0.055103. Element quality is considered excellent for values of 0.95-1.00 and good for values of 0.20-0.69, meaning that on average the element quality is between good and excellent [32]. While all three of these mesh quality statistics have some elements that would be considered bad quality, the vast majority of the elements would be considered of good quality. For example, the element quality plot shown in Figure 54 indicates that nearly all elements have an element quality of at least 0.2, which is the minimum element quality still considered to be good. Based on this, it is expected that the small number of poor quality element did not significantly impact the results of the simulation.

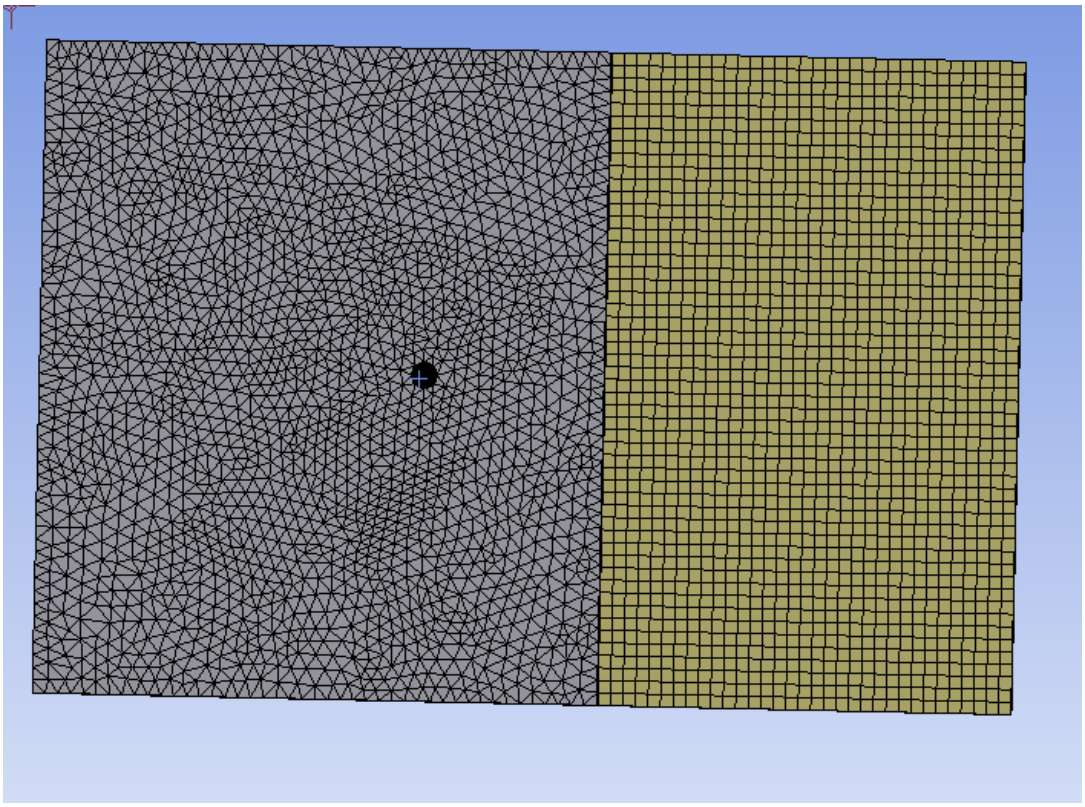

Figure 52. Top view of the overall mesh used for the PDMS device, symmetry used on the left face to reduce number of mesh elements needed. The gray meshed portion is the mesh for the PDMS device and yellow part of the device is the mesh for the glass microscope slide.

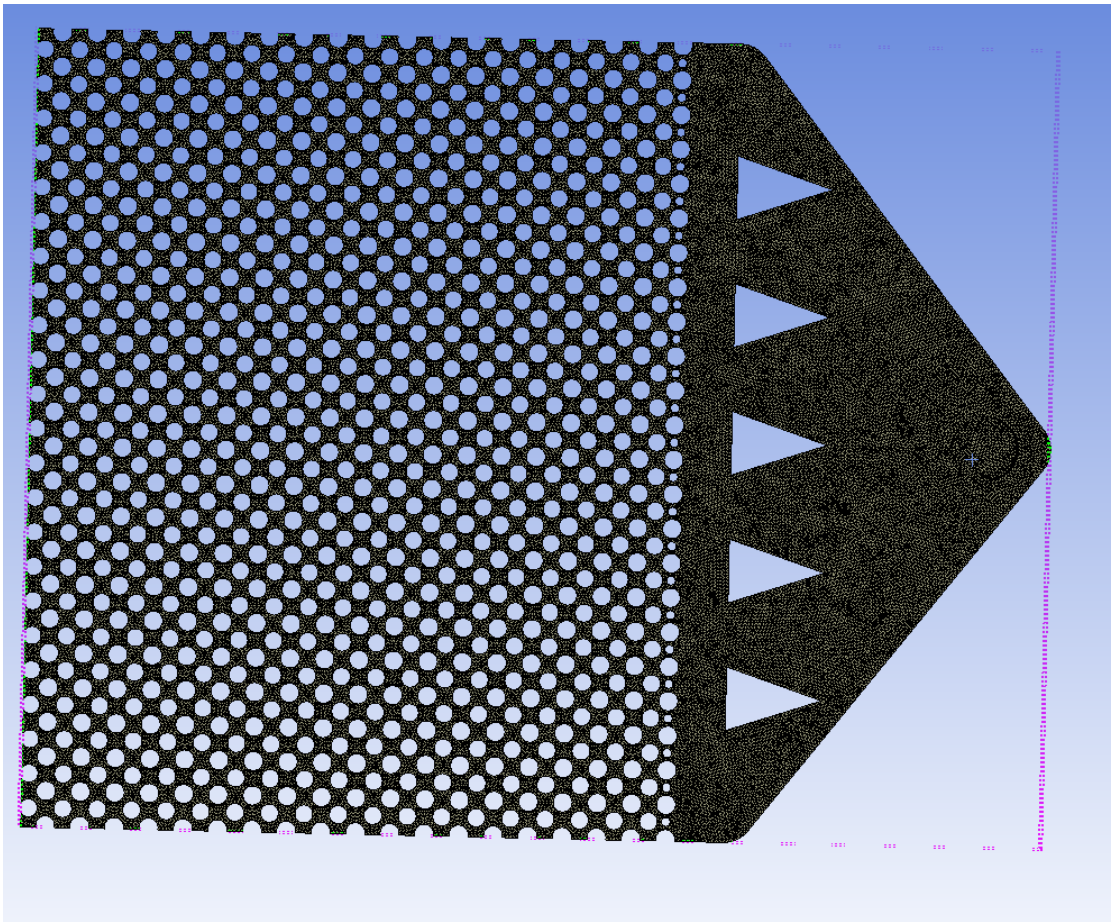

Figure 53. Mesh used for the water inside the PDMS device.

Table 6
Summary of mesh quality statistics for FEA simulation.

Mesh Quality Statistics	
Overall Element Size (mm)	0.05
Water Element Size (mm)	0.005
Nodes	4164451
Elements	2544068
Average Aspect Ratio	1.1615
Maximum Aspect Ratio	39.394
Average Skewness	0.27787
Maximum Skewness	0.9976
Average Element Quality	0.79609
Minimum Average Quality	0.0551

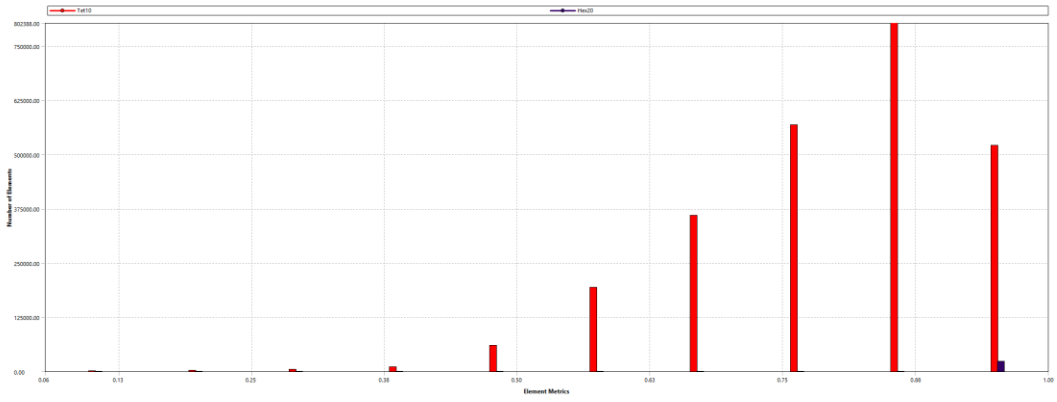

Figure 54. Element quality plot for mesh used in simulation, elements closer to an element quality of 1.0 are of higher quality.

The overall temperature contour plot for the top surface of the PDMS device is shown below in Figure 55. The water temperature is shown in Figure 56, with a temperature of 44.724 °C in the center of the water within the device. The resulting temperature

distribution within the PDMS device is given below in Figure 57, which plots the temperature determined in the simulation against the distance from the cold reservoir. Due to the symmetry of the device, only half of the device was simulated. The results of the simulation indicate the temperature of the device is more than 5 °C lower than the temperature of the reservoirs. The temperature near the outlet port of the device is similar to the temperature of the center of the device, so temperature measurements for the temperature calibration were taken in the center of the device and a thermocouple inserted into the outlet of the device was used to measure the temperature.

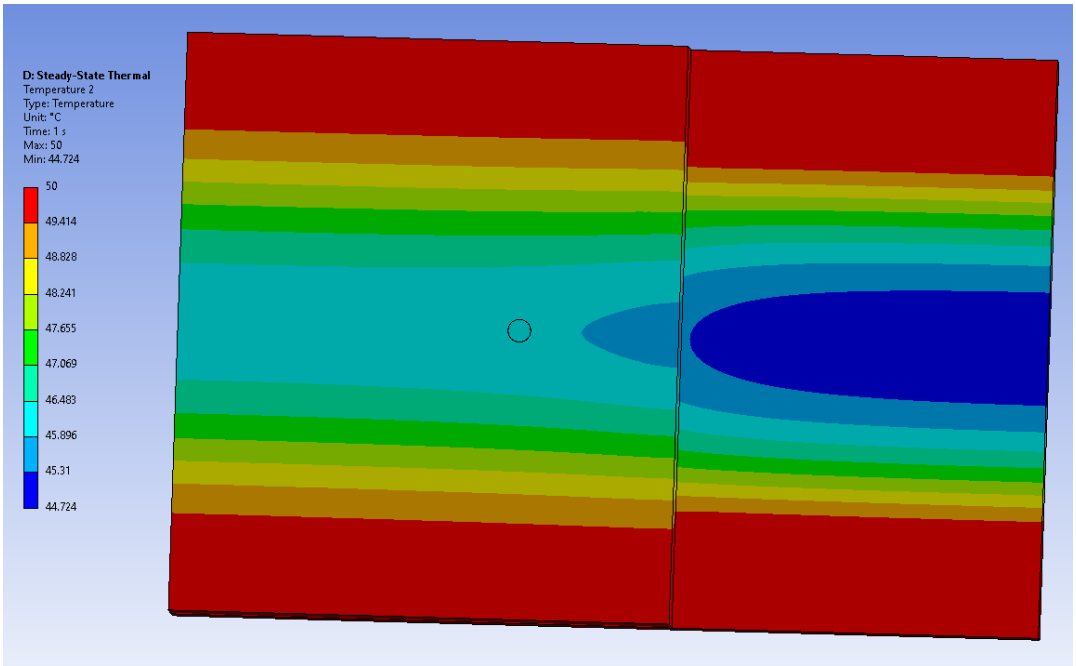

Figure 55. Overall temperature results for fixed 50 °C walls with natural convection.

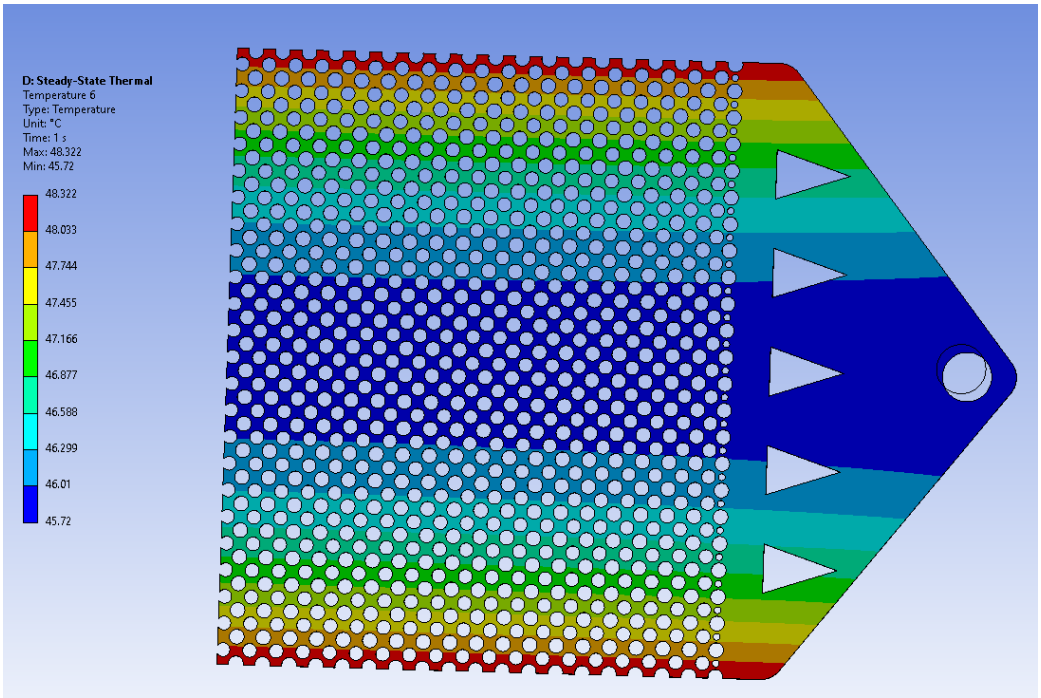

Figure 56. Water temperature contour plot for fixed 50 °C walls with natural convection.

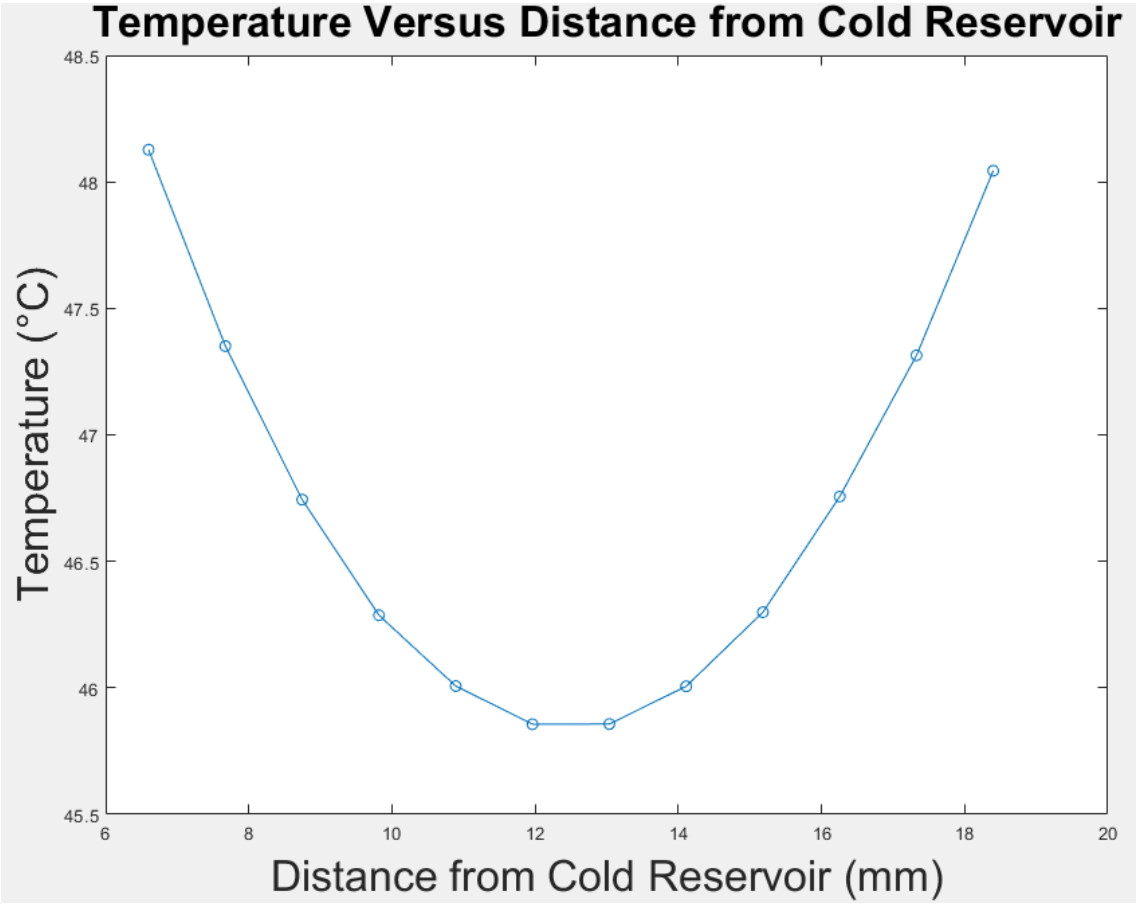

Figure 57. Simulated water temperature results for fixed 50 °C walls with natural convection. Temperature determined along the left edge of the device, corresponding to the approximate center of the PDMS device.

A.1.2 Temperature Gradient Applied to PDMS Device

A second case was simulated in which one side of the PDMS device was given a fixed temperature of 66.8 °C and the other was given a fixed temperature of 21.8 °C. The same natural convection parameter was used as before, as the average surface temperature were found to be similar. The overall contour plot is shown below in Figure 58. The water temperature contour plot is shown below in Figure 59. To test how much of an effect the natural convection had, a comparison with and without natural convection is shown below

in Figure 60. Temperature of the water is observed to increase more rapidly as the water moves further from the cold reservoir. Water near the bottom of the PDMS device was at a temperature of 25.978 °C while water neat the top of the device is at 59.869 °C. This is expected as near the cold reservoir the temperature of the water is closer to the room temperature and the heat transfer rate would be lower between the water and the ambient air. Temperatures are up to 3 °C colder in the simulation with natural convection when compared to the simulation without natural convection. Due to the Rayleigh number for the bottom glass surface not being within the range required for the natural convection, there is some inaccuracy in the calculated natural convection heat transfer coefficient but this value is reasonable for natural convection. The change in temperature throughout the flow channel is significant enough that the LIF measurement technique can be adequately tested. While the silicon device used for n-decane-soluble fluorescent dyes is a bit different from that of the PDMS, the geometry does not vary significantly enough from the PDMS device where the temperature distribution within the flow channel would be significantly different.

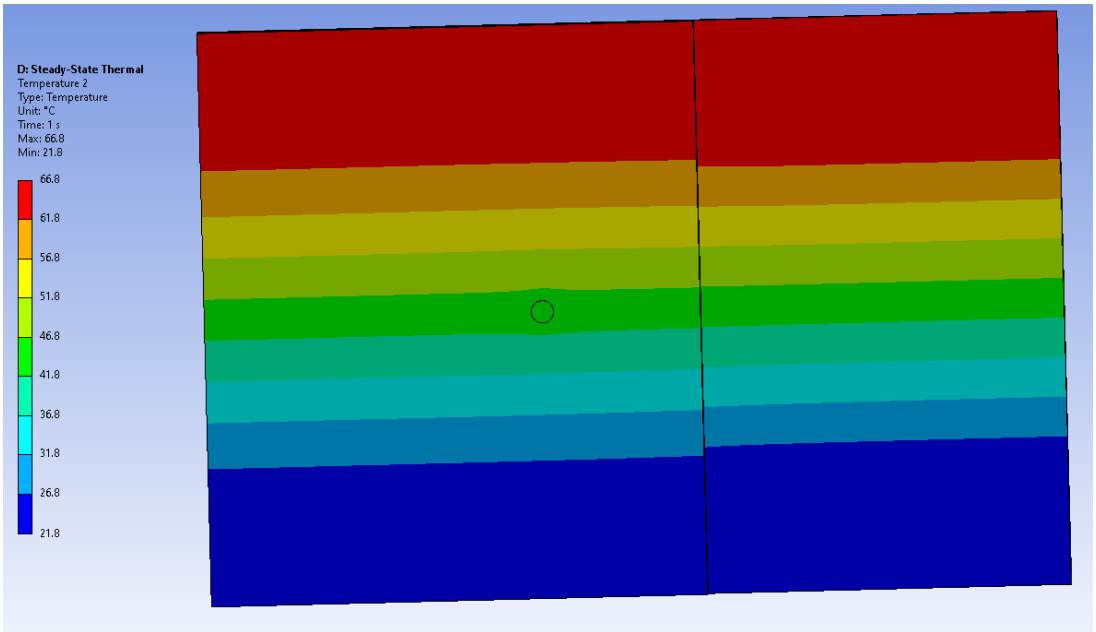

Figure 58. Overall temperature results for fixed 66.8 °C and 21.8 °C walls with natural convection.

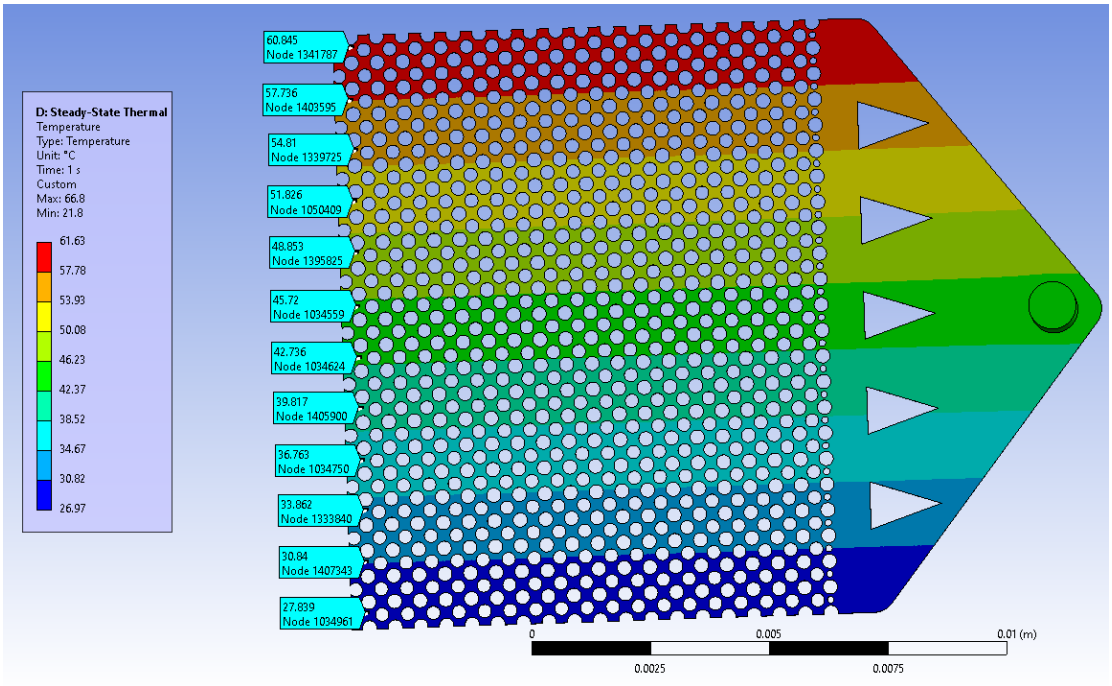

Figure 59. Water temperature contour plot for fixed 66.8 °C and 21.8 °C walls with natural convection.

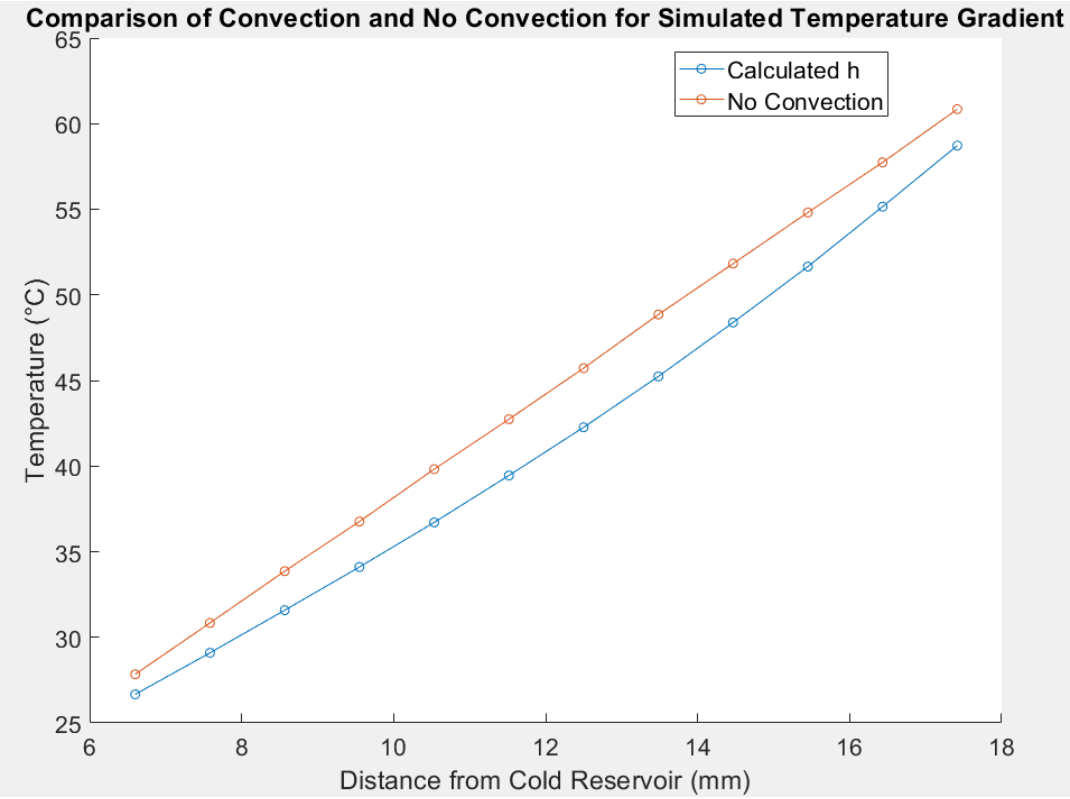

Figure 60. Comparison of water temperature results between simulation with no natural convection and the simulation using the calculated natural convection heat transfer coefficient.

APPENDIX B

B.1 Data Processing MATLAB Code

B.1.1 Temporal Resolution Code

```
clear; close all;
%dye 1
testNumbers = [103,104,105,106,107];
tempCases = ["17.2°C","17.2°C","17.2°C","17.2°C","17.2°C"];
tempNumbers = [17.2,17.2,17.2,17.2,17.2];
dye = "Sulforhodamine B Sodium Salt (0.2 g/L)";


Date = "2023-06-02";
Date2 = 20230602;
Date3 = 20230602;
%use average of center m x n pixels of image
numImages = 24;
m=10;
n=10;
x_loc  = 1122;
y_loc = 2160-856;
%whether or not to save plots and tables in process_dir
saveData = 1;
imagesUsed = 1;
startingImage = 1;
finalImage = imagesUsed+startingImage-1;

%data paths
images_dir = "G:\Shared drives\TEFL - LIF Thermometry #2\Keep These\";
process_dir = "G:\Shared drives\TFEL - LIF Thermometry\Processed Data and
Figures\Temporal_Resolution_" +Date+"\";
mkdir(process_dir)
%plot markers
markers = ["-o", "-*", "-^", "-v", "-x", "-d", "-+", "-s", "->", "-<", "-p", "-
h", "-."];
testNumbers = string(num2cell(testNumbers));
imageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_'+testNumbers+'_cam1_'+tempNumbers+'C_24')
filePattern1 = fullfile(imageTestFolder1, '*.tif');
tifFiles1 = [];
for i = 1:length(filePattern1)
    tifFiles1 = [tifFiles1; dir(filePattern1(i))];
end
tifFilesSorted1 = natsortfiles(tifFiles1);
pixelValue1 = zeros(length(tifFilesSorted1),1);
intensityValue1 = zeros(length(tifFilesSorted1),1);

imageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_'+testNumbers+'_cam2_'+tempNumbers+'C_24')
filePattern2 = fullfile(imageTestFolder2, '*.tif');
```

```
tifFiles2 = [];
for i = 1:length(filePattern2)
    tifFiles2 = [tifFiles2; dir(filePattern2(i))];
end
tifFilesSorted2 = natsortfiles(tifFiles2);
pixelValue2 = zeros(length(tifFilesSorted2),1);
intensityValue2 = zeros(length(tifFilesSorted2),1);
imageCalibFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_101_cam1_calib\'+Date3+'_101_cam1_calib_X1.tif');
imageCalibFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_101_cam2_calib\'+Date3+'_101_cam2_calib_X1.tif');
imageCalib1 = imread(imageCalibFile1);
imageCalib2 = imread(imageCalibFile2);
imageCalib1 = imresize(imageCalib1,[2160 2560]);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
imshow(imageCalib2,[0 300])
imageTranformFromCam2to1 = imregcorr(imageCalib2,imageCalib1);
%Rfixed = imref2d(size(imageCalib1));
%imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1,'OutputView',Rfixed);
imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
figure()
imshow(imageCalib1,[0 300])
figure()
imshow(imageCalib2,[0 100])
bgNumber = 102;
bgimageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_'+bgNumber+'_cam1_water_las');
bgfilePattern1 = fullfile(bgimageTestFolder1, '*.tif');
bgtifFiles1 = [];
for i = 1:length(bgfilePattern1)
    bgtifFiles1 = [bgtifFiles1; dir(bgfilePattern1(i))];
end
bgtifFilesSorted1 = natsortfiles(bgtifFiles1);
bgpixelValue1 = zeros(length(bgtifFilesSorted1),1);
bgintensityValue1 = zeros(length(bgtifFilesSorted1),1);

bgimageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_'+bgNumber+'_cam2_water_las');
bgfilePattern2 = fullfile(bgimageTestFolder2, '*.tif');
bgtifFiles2 = [];
for i = 1:length(bgfilePattern2)
    bgtifFiles2 = [bgtifFiles2; dir(bgfilePattern2(i))];
end
bgtifFilesSorted2 = natsortfiles(bgtifFiles2);
bgpixelValue2 = zeros(length(bgtifFilesSorted2),1);
bgintensityValue2 = zeros(length(bgtifFilesSorted2),1);
%EY:
imageBackgroundFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_102_cam1_water_las\'+Date2+'_102_cam1_water_las_X1.
tif');
imageBackgroundFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_102_cam2_water_las\'+Date2+'_102_cam2_water_las_X1.
tif');
```

```
imageBackground1 = zeros([2160 2560]);
imageBackground2 = zeros([2160 2560]);
for k = 1:24
    bgbaseFileName1 = bgtifFilesSorted1(k).name;
    bgbaseFileName2 = bgtifFilesSorted2(k).name;
    bgfullFileName1 = fullfile(bgimageTestFolder1(floor((k+numImages-
1)/numImages)),bgbaseFileName1);
    bgfullFileName2 = fullfile(bgimageTestFolder2(floor((k+numImages-
1)/numImages)),bgbaseFileName2);
    fprintf(1, 'Now reading %s\n', bgfullFileName1);
    fprintf(1, 'Now reading %s\n', bgfullFileName2);
    imageBackground1 = imageBackground1 +
double(imread(imageBackgroundFile1));
    imageBackground2 = imageBackground2 +
double(imread(imageBackgroundFile2));
end
imageBackground1 = imageBackground1/24;
imageBackground2 = imageBackground2/24;
imageBackground2 = imwarp(imageBackground2,imageTranformFromCam2to1);
imageBackground2 = imresize(imageBackground2,[2160 2560]);

%SRB Sodium Salt
%imageBackgroundFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_115_cam1_water_las\'+Date2+'_115_cam1_water_las_X1.ti
f');
%imageBackgroundFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_115_cam2_water_las\'+Date2+'_115_cam2_water_las_X1.ti
f');
%
% %to test background with no laser light:
% %imageBackgroundFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_102_cam1_bg\'+Date2+'_102_cam1_bg_X1.tif');
% %imageBackgroundFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_102_cam2_bg\'+Date2+'_102_cam2_bg_X1.tif');



%to test no background:
%imageBackground1 = zeros([2160 2560]);
%imageBackground2 = zeros([2160 2560]);
for k = 1:length(tifFilesSorted1)
    baseFileName1 = tifFilesSorted1(k).name;
    baseFileName2 = tifFilesSorted2(k).name;
    fullFileName1 = fullfile(imageTestFolder1(floor((k+numImages-
1)/numImages)),baseFileName1);
    fullFileName2 = fullfile(imageTestFolder2(floor((k+numImages-
1)/numImages)),baseFileName2);
    fprintf(1, 'Now reading %s\n', fullFileName1);
    fprintf(1, 'Now reading %s\n', fullFileName2);
    %if image is 2160 x 2560 pixels, center roughly here:
    %imageValue = imread(fullFileName,'PixelRegion',{[1072,1087],[1272,1287]});
    imageValue1 = double(imread(fullFileName1));
    imageValue2 = double(imread(fullFileName2));
    Rfixed=imref2d(size(imageValue1));
    imageValue2 = imwarp(imageValue2,imageTranformFromCam2to1);
```

```
    imageValue2 = imresize(imageValue2,[2160 2560]);
    %imageValue2 = imwarp(imageValue2, imageTranformFromCam2to1);
     if mod(m,2) == 0 && mod(n,2) == 0 %%if m and n are even, center m x n
pixels
        pixelValue1(k) = mean(imageValue1(y_loc-
floor(m/2):y_loc+floor(m/2)-1,x_loc-floor(n/2):x_loc+floor(n/2)-1) -
imageBackground1(y_loc-floor(m/2):y_loc+floor(m/2)-1,x_loc-
floor(n/2):x_loc+floor(n/2)-1),'all');
        pixelValue2(k) = mean(imageValue2(y_loc-
floor(m/2):y_loc+floor(m/2)-1,x_loc-floor(n/2):x_loc+floor(n/2)-1) -
imageBackground2(y_loc-floor(m/2):y_loc+floor(m/2)-1,x_loc-
floor(n/2):x_loc+floor(n/2)-1),'all');
    elseif mod(m,2) == 1 && mod(n,2) == 1 %%if m and n are even, center m x n
pixels
        pixelValue1(k) = mean(imageValue1(y_loc-
floor(m/2):y_loc+floor(m/2),x_loc-floor(n/2):x_loc+floor(n/2)) -
imageBackground1(y_loc-floor(m/2):y_loc+floor(m/2),x_loc-
floor(n/2):x_loc+floor(n/2)),'all');
        pixelValue2(k) = mean(imageValue2(y_loc-
floor(m/2):y_loc+floor(m/2),x_loc-floor(n/2):x_loc+floor(n/2)) -
imageBackground2(y_loc-floor(m/2):y_loc+floor(m/2),x_loc-
floor(n/2):x_loc+floor(n/2)),'all');
    elseif m == 1 && n == 1 %center 1x1 pixel
        pixelValue1(k) = mean(imageValue1(y_loc,x_loc) -
imageBackground1(y_loc,x_loc),'all');
        pixelValue2(k) = mean(imageValue2(y_loc,x_loc) -
imageBackground1(y_loc,x_loc),'all');
    end
end
intensity1 = [];
intensity2 = [];
%extract intensity values, with each column corresponding to a different
temperature
for i = 1:length(testNumbers)
    intensity1 = [intensity1, pixelValue1(1+numImages*(i-1):numImages*i)];
    intensity2 = [intensity2, pixelValue2(1+numImages*(i-1):numImages*i)];
end
figure()
hold on;
legendText = [];
for i = 1:length(testNumbers)
    plot(intensity1(:,i), "-ko")
    plot(intensity2(:,i), "-bo")
end
ylabel("Camera Voltage (a.u.)","FontSize",18);
xlabel("Image Number","FontSize",18);
legend("Camera 1 Records","","","","","Camera 2
Records","Location","best","FontSize",14);
titleLabel = append("Plot of Camera Voltage Using Center for 5 Records \newline
of Sulforhodamine B Sodium Salt 0.2 g/L");
title(titleLabel,"FontSize",24);
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
```

```
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'imreadCamera1Plot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'imreadCamera1Plot'), 'png')
end
hold off
intensityRatio = [];
%extract intensity values, with each column corresponding to a different
temperature
for i = 1:length(testNumbers)
    %intensityRatio = [intensityRatio, (pixelValue1(1+numImages*(i-1):numImages*i)
- Pi_SR101(1+numImages*(i-1):numImages*i).*pixelValue2(1+numImages*(i-
1):numImages*i))./ pixelValue2(1+numImages*(i-1):numImages*i)];
    intensityRatio = [intensityRatio, pixelValue1(1+numImages*(i-
1):numImages*i)./ pixelValue2(1+numImages*(i-1):numImages*i)];
end
figure()
hold on;
legendText = [];
for i = 1:5
    plot(intensityRatio(:,i))
    legendText = [legendText, tempCases(i)];
end
ylabel("Camera Voltage Ratio, Camera 1/Camera 2 (a.u.)","FontSize",18);
xlabel("Image Number","FontSize",18);
%legend(legendText,"Location","best");
legend("Record 1","Record 2", "Record 3", "Record 4","Record 5","FontSize",14)
titleLabel = append("Plot of Camera Voltage Ratio for 5 Records \newline of
Sulforhodamine B Sodium Salt 0.2 g/L");
title(titleLabel,"FontSize",24);
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'intensityRatioPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'intensityRatioPlot'), 'png')
end
hold off
intensityRatio_v1 = intensityRatio(:,1)
intensityRatio_v2 = intensityRatio(:,2)
intensityRatio_v3 = intensityRatio(:,3)
intensityRatio_v4 = intensityRatio(:,4)
intensityRatio_v5 = intensityRatio(:,5)
intensityRatioSTD_v1 = zeros([1 24])
intensityRatioSTD_v2 = zeros([1 24])
intensityRatioSTD_v3 = zeros([1 24])
intensityRatioSTD_v4 = zeros([1 24])
intensityRatioSTD_v5 = zeros([1 24])
for i = 1:24
    intensityRatioSTD_v1(i) = std(intensityRatio_v1(1:i))
    intensityRatioSTD_v2(i) = std(intensityRatio_v2(1:i))
    intensityRatioSTD_v3(i) = std(intensityRatio_v3(1:i))
    intensityRatioSTD_v4(i) = std(intensityRatio_v4(1:i))
    intensityRatioSTD_v5(i) = std(intensityRatio_v5(1:i))
end
figure()
```

```
h = plot(linspace(1,24,24),intensityRatioSTD_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioSTD_v2(1,1:24),"-o")
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
hold on
plot(linspace(1,24,24),intensityRatioSTD_v2(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v3(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v4(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v5(1,1:24),"-o")
legend("Record 1","Record 2", "Record 3", "Record 4","Record 5","FontSize",14)
xlabel("Number of Images","FontSize",18)
ylabel("Camera Voltage Ratio Standard Deviation (a.u.)","FontSize",18)
title("Standard Deviation versus Number of Images Plot for \newlineSulforhodamine
B Sodium Salt (0.2 g/L)","FontSize",24)
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'STDPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'STDPlot'), 'png')
end
hold off



figure()
intensityRatioMean_v1 = zeros([1 24])
intensityRatioMean_v2 = zeros([1 24])
intensityRatioMean_v3 = zeros([1 24])
intensityRatioMean_v4 = zeros([1 24])
intensityRatioMean_v5 = zeros([1 24])
for i = 1:24
    intensityRatioMean_v1(i) = mean(intensityRatio_v1(1:i))
    intensityRatioMean_v2(i) = mean(intensityRatio_v2(1:i))
    intensityRatioMean_v3(i) = mean(intensityRatio_v3(1:i))
    intensityRatioMean_v4(i) = mean(intensityRatio_v4(1:i))
    intensityRatioMean_v5(i) = mean(intensityRatio_v5(1:i))
end
h = plot(linspace(1,24,24),intensityRatioMean_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
hold on
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v3(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v4(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v5(1,1:24),"-o")
xlabel("Number of Images","FontSize",18)
ylabel("Camera Voltage Ratio Mean (a.u.)","FontSize",18)
legend("Record 1","Record 2", "Record 3", "Record 4","Record 5","FontSize",14)
title("Mean Voltage Ratio versus Number of Images Plot for \newlineSulforhodamine
B Sodium Salt (0.2 g/L)","FontSize",24)
```

```
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'MeanPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'MeanPlot'), 'png')
end
hold off

figure()
intensityRatioCV_v1 = zeros([1 24])
intensityRatioCV_v2 = zeros([1 24])
intensityRatioCV_v3 = zeros([1 24])
intensityRatioCV_v4 = zeros([1 24])
intensityRatioCV_v5 = zeros([1 24])
for i = 1:24
    intensityRatioCV_v1(i) =
100*std(intensityRatio_v1(1:i))/mean(intensityRatio_v1(1:i))
    intensityRatioCV_v2(i) =
100*std(intensityRatio_v2(1:i))/mean(intensityRatio_v2(1:i))
    intensityRatioCV_v3(i) =
100*std(intensityRatio_v3(1:i))/mean(intensityRatio_v3(1:i))
    intensityRatioCV_v4(i) =
100*std(intensityRatio_v4(1:i))/mean(intensityRatio_v4(1:i))
    intensityRatioCV_v5(i) =
100*std(intensityRatio_v5(1:i))/mean(intensityRatio_v5(1:i))
end
h = plot(linspace(1,24,24),intensityRatioCV_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
hold on
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
plot(linspace(1,24,24),intensityRatioCV_v2(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioCV_v3(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioCV_v4(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioCV_v5(1,1:24),"-o")
xlabel("Number of Images","FontSize",18)
ylabel("Camera Voltage Ratio Coefficient of Variance (%)","FontSize",18)
legend("Record 1","Record 2", "Record 3", "Record 4","Record 5","FontSize",14)
title("Voltage Ratio Coefficient of Variance versus Number of Images Plot for
\newlineSulforhodamine B Sodium Salt (0.2 g/L)","FontSize",24)
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'CoefficientofVariancePlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'CoefficientofVariancePlot'), 'png')
end
hold off
figure()
intensityRatioMean_v1 = zeros([1 24])
intensityRatioMean_v12 = zeros([1 24])
intensityRatioMean_v123 = zeros([1 24])
```

```
intensityRatioMean_v1234 = zeros([1 24])
intensityRatioMean_v12345 = zeros([1 24])
for i = 1:24
    intensityRatioMean_v1(i) = mean(intensityRatio_v1(1:i))
    intensityRatioMean_v12(i)
=mean((intensityRatio_v1(1:i)+intensityRatio_v2(1:i))/2);
    intensityRatioMean_v123(i)
=mean((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i
))/3);
    intensityRatioMean_v1234(i)
=mean((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i
)+intensityRatio_v4(1:i))/4);
    intensityRatioMean_v12345(i)
=mean((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i
)+intensityRatio_v4(1:i)+intensityRatio_v5(1:i))/5);
%     intensityRatioMean_v2(i) = mean(intensityRatio_v2(1:i));
%     intensityRatioMean_v12(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i)))/2;
%     intensityRatioMean_v123(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i)))/3;
%     intensityRatioMean_v1234(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i))+mean(intensityRatio_v5(1:i)))/4;
end
h = plot(linspace(1,24,24),intensityRatioMean_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
hold on
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
plot(linspace(1,24,24),intensityRatioMean_v12(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v123(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v1234(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v12345(1,1:24),"-o")
xlabel("Number of Images","FontSize",18)
ylabel("Camera Voltage Ratio Mean (a.u.)","FontSize",18)
legend("1 Record","2 Records", "3 Records", "4 Records","5
Records","FontSize",14)
title("Mean Voltage Ratio versus Number of Images for varying Number of Records
Plot for \newlineSulforhodamine B Sodium Salt (0.2 g/L)","FontSize",24)
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'MeanNumImagePlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'MeanNumImagePlot'), 'png')
end
hold off
figure()
intensityRatioSTD_v1 = zeros([1 24])
intensityRatioSTD_v12 = zeros([1 24])
intensityRatioSTD_v123 = zeros([1 24])
intensityRatioSTD_v1234 = zeros([1 24])
intensityRatioSTD_v12345 = zeros([1 24])
```

```
for i = 1:24
    intensityRatioSTD_v1(i) = std(intensityRatio_v1(1:i))
    intensityRatioSTD_v12(i) =
std((intensityRatio_v1(1:i)+intensityRatio_v2(1:i))/2);
    intensityRatioSTD_v123(i) =
std((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i))
/3);
    intensityRatioSTD_v1234(i) =
std((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i)+
intensityRatio_v4(1:i))/4);
    intensityRatioSTD_v12345(i) =
std((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i)+
intensityRatio_v4(1:i)+intensityRatio_v5(1:i))/5);
%     intensityRatioMean_v2(i) = mean(intensityRatio_v2(1:i));
%     intensityRatioMean_v12(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i)))/2;
%     intensityRatioMean_v123(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i)))/3;
%     intensityRatioMean_v1234(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i))+mean(intensityRatio_v5(1:i)))/4;
end
h = plot(linspace(1,24,24),intensityRatioSTD_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
hold on
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
plot(linspace(1,24,24),intensityRatioSTD_v12(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v123(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v1234(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v12345(1,1:24),"-o")
xlabel("Number of Images","FontSize",18)
ylabel("Camera Voltage Ratio Standard Deviation (a.u.)","FontSize",18)
legend("1 Record","2 Records", "3 Records", "4 Records", "5
Records","FontSize",14)
title("Standard Deviation versus Number of Images for varying Number of Records
Plot for \newlineSulforhodamine B Sodium Salt (0.2 g/L)","FontSize",24)
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'STDRecordPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'STDRecordPlot'), 'png')
end
hold off
figure()
intensityRatioCOV_v1 = zeros([1 24])
intensityRatioCOV_v12 = zeros([1 24])
intensityRatioCOV_v123 = zeros([1 24])
intensityRatioCOV_v1234 = zeros([1 24])
intensityRatioCOV_v12345 = zeros([1 24])
for i = 1:24;
```

```
    intensityRatioCOV_v1(i) =
intensityRatioSTD_v1(1:i)/intensityRatioMean_v1(1:i)*100
    intensityRatioCOV_v12(i) =
intensityRatioSTD_v12(1:i)/intensityRatioMean_v12(1:i)*100;
    intensityRatioCOV_v123(i) =
intensityRatioSTD_v123(1:i)/intensityRatioMean_v123(1:i)*100;
    intensityRatioCOV_v1234(i) =
intensityRatioSTD_v1234(1:i)/intensityRatioMean_v1234(1:i)*100;
    intensityRatioCOV_v12345(i) =
intensityRatioSTD_v12345(1:i)/intensityRatioMean_v12345(1:i)*100;
%     intensityRatioMean_v2(i) = mean(intensityRatio_v2(1:i));
%     intensityRatioMean_v12(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i)))/2;
%     intensityRatioMean_v123(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i)))/3;
%     intensityRatioMean_v1234(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i))+mean(intensityRatio_v5(1:i)))/4;
end
h = plot(linspace(1,24,24),intensityRatioCOV_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
hold on
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
plot(linspace(1,24,24),intensityRatioCOV_v12(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioCOV_v123(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioCOV_v1234(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioCOV_v12345(1,1:24),"-o")
xlabel("Number of Images","FontSize",18)
ylabel("Camera Voltage Ratio Coefficient of Variance (%)","FontSize",18)
legend("1 Record","2 Records", "3 Records", "4 Records", "5
Records","FontSize",14)
title("Voltage Ratio Coefficient of Variance versus Number of Images for varying
Number of Records Plot for \newlineSulforhodamine B Sodium Salt (0.2
g/L)","FontSize",24)
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'CoVNumImagePlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'CoVNumImagePlot'), 'png')
end
hold off
figure()
hold on
[f1,xi1] = ksdensity(intensityRatio_v1);
plot(xi1,f1)
[f2,xi2] = ksdensity(intensityRatio_v2);
plot(xi2,f2)
[f3,xi3] = ksdensity(intensityRatio_v3);
plot(xi3,f3)
[f4,xi4] = ksdensity(intensityRatio_v4);
plot(xi4,f4)
```

```
xlabel("Fluorescnce Intensity Ratio")
ylabel("Probability Density Function")
legend("Record 1","Record 2","Record 3","Record 4","Record 5")
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'IntensityRatioPDFPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'IntensityRatioPDFPlot'), 'png')
end
hold off

intensityratio_mean = mean(intensityRatio)
intensityratio_var = var(intensityRatio)
%30 images

% y =
[intensityRatio_v1;intensityRatio_v2;intensityRatio_v3;intensityRatio_v4;intensityR
atio_v5];
% group = repelem(1:5, 1, [numel(intensityRatio_v1),
numel(intensityRatio_v2),numel(intensityRatio_v3),numel(intensityRatio_v4),numel(in
tensityRatio_v5) ])
% p = anova1(y,group)
% xlabel("Record Number")
% ylabel("Fluorescence Intensity Ratio")
% title("ANOVA Test for Two Records at Same Temperature, 24 Images")
y =
[intensityRatio_v1;intensityRatio_v2;intensityRatio_v3;intensityRatio_v4;i
ntensityRatio_v5];
group = repelem(1:5, 1,
[numel(intensityRatio_v1),numel(intensityRatio_v2),numel(intensityRatio_v3
),numel(intensityRatio_v4),numel(intensityRatio_v5) ]);
p = anova1(y,group);
xlabel("Record Number","FontSize",18)
ylabel("Fluorescence Intensity Ratio","FontSize",18)
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
title("ANOVA Test for Four Records at Same Temperature, 24 Images","FontSize",24)
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'ANOVAPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'ANOVAPlot'), 'png')
end
recordNum = ["Record 1"; "Record 2"; "Record 3"; "Record 4"; "Record 5"; "First
2"; "First 3"; "First 4"; "All 5" ]
meanval = [intensityRatioMean_v1(24); intensityRatioMean_v2(24);
intensityRatioMean_v3(24); intensityRatioMean_v4(24);
intensityRatioMean_v5(24);intensityRatioMean_v12(24) ;
intensityRatioMean_v123(24) ; intensityRatioMean_v1234(24);
intensityRatioMean_v12345(24) ]
stdval = [intensityRatioSTD_v1(24); intensityRatioSTD_v2(24);
intensityRatioSTD_v3(24);
intensityRatioSTD_v4(24);intensityRatioSTD_v5(24);
intensityRatioSTD_v12(24); intensityRatioSTD_v123(24);
intensityRatioSTD_v1234(24); intensityRatioSTD_v12345(24)]
relative_STD = stdval./meanval
percent_STD = 100*relative_STD
```

```
percent_diff = abs(meanval -
intensityRatioMean_v12345(24))./(meanval+intensityRatioMean_v12345(24))/2*
100
results = table(recordNum, meanval, stdval, percent_STD,percent_diff)

if saveData == 1
    writetable(results,fullfile(process_dir,
'TemporalResolutionSummary.xlsx'))
end
```

Published with MATLAB® R2021b

B.1.2 Spatial Resolution Code

```
clear; close all;
%dye 1
testNumbers = [103,104,105,106,107];
tempCases = ["17.2°C","17.2°C","17.2°C","17.2°C","17.2°C"];
tempNumbers = [17.2,17.2,17.2,17.2,17.2];
dye = "Sulforhodamine B Sodium Salt (0.2 g/L)";


Date = "2023-06-02";
Date2 = 20230602;
Date3 = 20230602;
%use average of center m x n pixels of image
numImages = 24;
m=8;
n=8;
x_loc  = 1122;
y_loc = 2160-856;
%whether or not to save plots and tables in process_dir
saveData = 1;
imagesUsed = 1;
startingImage = 1;
finalImage = imagesUsed+startingImage-1;

%data paths
images_dir = "G:\Shared drives\TEFL - LIF Thermometry #2\Keep These\";
process_dir = "G:\Shared drives\TFEL - LIF Thermometry\Processed Data and
Figures\Spatial_Resolution_" +Date+"\";
mkdir(process_dir)
%plot markers
markers = ["-o", "-*", "-^", "-v", "-x", "-d", "-+", "-s", "->", "-<", "-p", "-
h", "-."];
testNumbers = string(num2cell(testNumbers));
imageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_'+testNumbers+'_cam1_'+tempNumbers+'C_24')
filePattern1 = fullfile(imageTestFolder1, '*.tif');
tifFiles1 = [];
for i = 1:length(filePattern1)
    tifFiles1 = [tifFiles1; dir(filePattern1(i))];
end
tifFilesSorted1 = natsortfiles(tifFiles1);
```

```
pixelValue1 = zeros(length(tifFilesSorted1),1);
intensityValue1 = zeros(length(tifFilesSorted1),1);

imageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_'+testNumbers+'_cam2_'+tempNumbers+'C_24')
filePattern2 = fullfile(imageTestFolder2, '*.tif');
tifFiles2 = [];
for i = 1:length(filePattern2)
    tifFiles2 = [tifFiles2; dir(filePattern2(i))];
end
tifFilesSorted2 = natsortfiles(tifFiles2);
pixelValue2 = zeros(length(tifFilesSorted2),1);
intensityValue2 = zeros(length(tifFilesSorted2),1);
imageCalibFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_101_cam1_calib\'+Date3+'_101_cam1_calib_X1.tif');
imageCalibFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_101_cam2_calib\'+Date3+'_101_cam2_calib_X1.tif');
imageCalib1 = imread(imageCalibFile1);
imageCalib2 = imread(imageCalibFile2);
imageCalib1 = imresize(imageCalib1,[2160 2560]);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
imshow(imageCalib2,[0 300])
imageTranformFromCam2to1 = imregcorr(imageCalib2,imageCalib1);
%Rfixed = imref2d(size(imageCalib1));
%imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1,'OutputView',Rfixed);
imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
figure()
imshow(imageCalib1,[0 300])
figure()
imshow(imageCalib2,[0 100])
bgNumber = 102;
bgimageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_'+bgNumber+'_cam1_water_las');
bgfilePattern1 = fullfile(bgimageTestFolder1, '*.tif');
bgtifFiles1 = [];
for i = 1:length(bgfilePattern1)
    bgtifFiles1 = [bgtifFiles1; dir(bgfilePattern1(i))];
end
bgtifFilesSorted1 = natsortfiles(bgtifFiles1);
bgpixelValue1 = zeros(length(bgtifFilesSorted1),1);
bgintensityValue1 = zeros(length(bgtifFilesSorted1),1);

bgimageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_'+bgNumber+'_cam2_water_las');
bgfilePattern2 = fullfile(bgimageTestFolder2, '*.tif');
bgtifFiles2 = [];
for i = 1:length(bgfilePattern2)
    bgtifFiles2 = [bgtifFiles2; dir(bgfilePattern2(i))];
end
bgtifFilesSorted2 = natsortfiles(bgtifFiles2);
bgpixelValue2 = zeros(length(bgtifFilesSorted2),1);
bgintensityValue2 = zeros(length(bgtifFilesSorted2),1);
%EY:
```

```
imageBackgroundFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_102_cam1_water_las\'+Date2+'_102_cam1_water_las_X1.
tif');
imageBackgroundFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_102_cam2_water_las\'+Date2+'_102_cam2_water_las_X1.
tif');
imageBackground1 = zeros([2160 2560]);
imageBackground2 = zeros([2160 2560]);
for k = 1:24
    bgbaseFileName1 = bgtifFilesSorted1(k).name;
    bgbaseFileName2 = bgtifFilesSorted2(k).name;
    bgfullFileName1 = fullfile(bgimageTestFolder1(floor((k+numImages-
1)/numImages)),bgbaseFileName1);
    bgfullFileName2 = fullfile(bgimageTestFolder2(floor((k+numImages-
1)/numImages)),bgbaseFileName2);
    fprintf(1, 'Now reading %s\n', bgfullFileName1);
    fprintf(1, 'Now reading %s\n', bgfullFileName2);
    imageBackground1 = imageBackground1 +
double(imread(imageBackgroundFile1));
    imageBackground2 = imageBackground2 +
double(imread(imageBackgroundFile2));
end
imageBackground1 = imageBackground1/24;
imageBackground2 = imageBackground2/24;
imageBackground2 = imwarp(imageBackground2,imageTranformFromCam2to1);
imageBackground2 = imresize(imageBackground2,[2160 2560]);

%SRB Sodium Salt
%imageBackgroundFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_115_cam1_water_las\'+Date2+'_115_cam1_water_las_X1.ti
f');
%imageBackgroundFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_115_cam2_water_las\'+Date2+'_115_cam2_water_las_X1.ti
f');
%
% %to test background with no laser light:
% %imageBackgroundFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_102_cam1_bg\'+Date2+'_102_cam1_bg_X1.tif');
% %imageBackgroundFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_102_cam2_bg\'+Date2+'_102_cam2_bg_X1.tif');



%to test no background:
%imageBackground1 = zeros([2160 2560]);
%imageBackground2 = zeros([2160 2560]);
for k = 1:length(tifFilesSorted1)
    baseFileName1 = tifFilesSorted1(k).name;
    baseFileName2 = tifFilesSorted2(k).name;
    fullFileName1 = fullfile(imageTestFolder1(floor((k+numImages-
1)/numImages)),baseFileName1);
    fullFileName2 = fullfile(imageTestFolder2(floor((k+numImages-
1)/numImages)),baseFileName2);
    fprintf(1, 'Now reading %s\n', fullFileName1);
    fprintf(1, 'Now reading %s\n', fullFileName2);
```

```
    %if image is 2160 x 2560 pixels, center roughly here:
    %imageValue = imread(fullFileName,'PixelRegion',{[1072,1087],[1272,1287]});
    imageValue1 = double(imread(fullFileName1));
    imageValue2 = double(imread(fullFileName2));
    Rfixed=imref2d(size(imageValue1));
    imageValue2 = imwarp(imageValue2,imageTranformFromCam2to1);
    imageValue2 = imresize(imageValue2,[2160 2560]);
    %imageValue2 = imwarp(imageValue2, imageTranformFromCam2to1);
     if mod(m,2) == 0 && mod(n,2) == 0 %%if m and n are even, center m x n
pixels
        pixelValue1(k) = mean(imageValue1(y_loc-
floor(m/2):y_loc+floor(m/2)-1,x_loc-floor(n/2):x_loc+floor(n/2)-1) -
imageBackground1(y_loc-floor(m/2):y_loc+floor(m/2)-1,x_loc-
floor(n/2):x_loc+floor(n/2)-1),'all');
        pixelValue2(k) = mean(imageValue2(y_loc-
floor(m/2):y_loc+floor(m/2)-1,x_loc-floor(n/2):x_loc+floor(n/2)-1) -
imageBackground2(y_loc-floor(m/2):y_loc+floor(m/2)-1,x_loc-
floor(n/2):x_loc+floor(n/2)-1),'all');
    elseif mod(m,2) == 1 && mod(n,2) == 1 %%if m and n are even, center m x n
pixels
        pixelValue1(k) = mean(imageValue1(y_loc-
floor(m/2):y_loc+floor(m/2),x_loc-floor(n/2):x_loc+floor(n/2)) -
imageBackground1(y_loc-floor(m/2):y_loc+floor(m/2),x_loc-
floor(n/2):x_loc+floor(n/2)),'all');
        pixelValue2(k) = mean(imageValue2(y_loc-
floor(m/2):y_loc+floor(m/2),x_loc-floor(n/2):x_loc+floor(n/2)) -
imageBackground2(y_loc-floor(m/2):y_loc+floor(m/2),x_loc-
floor(n/2):x_loc+floor(n/2)),'all');
    elseif m == 1 && n == 1 %center 1x1 pixel
        pixelValue1(k) = mean(imageValue1(y_loc,x_loc) -
imageBackground1(y_loc,x_loc),'all');
        pixelValue2(k) = mean(imageValue2(y_loc,x_loc) -
imageBackground1(y_loc,x_loc),'all');
    end
end
intensity1 = [];
intensity2 = [];
%extract intensity values, with each column corresponding to a different
temperature
for i = 1:length(testNumbers)
    intensity1 = [intensity1, pixelValue1(1+numImages*(i-1):numImages*i)];
    intensity2 = [intensity2, pixelValue2(1+numImages*(i-1):numImages*i)];
end
figure()
hold on;
legendText = [];
for i = 1:length(testNumbers)
    plot(intensity1(:,i), markers(i))
    plot(intensity2(:,i), markers(i))
end
ylabel("Light Intensity Value");
xlabel("Image Number");
%legend("Camera 1","Camera 2","Location","best");
titleLabel = append("Plot of Light Intensity Value Using Center " +m+ " x " +n+
" pixels" +newline + Date);
```

```
title(titleLabel);
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'imreadCamera1Plot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'imreadCamera1Plot'), 'png')
end
hold off
intensityRatio = [];
%extract intensity values, with each column corresponding to a different
temperature
for i = 1:length(testNumbers)
    %intensityRatio = [intensityRatio, (pixelValue1(1+numImages*(i-1):numImages*i)
- Pi_SR101(1+numImages*(i-1):numImages*i).*pixelValue2(1+numImages*(i-
1):numImages*i))./ pixelValue2(1+numImages*(i-1):numImages*i)];
    intensityRatio = [intensityRatio, pixelValue1(1+numImages*(i-
1):numImages*i)./ pixelValue2(1+numImages*(i-1):numImages*i)];
end
figure()
hold on;
legendText = [];
for i = 1:5
    plot(intensityRatio(:,i))
    legendText = [legendText, tempCases(i)];
end
ylabel("Light Intensity Ratio, Camera 1 : Camera 2");
xlabel("Image Number");
%legend(legendText,"Location","best");
legend("Record 1","Record 2", "Record 3", "Record 4","Record 5")
titleLabel = append("Plot of Light Intensity Ratio Without Pi Function Using
Center " +m+ " x " +n+ " pixels" +newline + Date);
title(titleLabel);
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'intensityRatioPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'intensityRatioPlot'), 'png')
end
hold off
intensityRatio_v1 = intensityRatio(:,1)
intensityRatio_v2 = intensityRatio(:,2)
intensityRatio_v3 = intensityRatio(:,3)
intensityRatio_v4 = intensityRatio(:,4)
intensityRatio_v5 = intensityRatio(:,5)
intensityRatioSTD_v1 = zeros([1 24])
intensityRatioSTD_v2 = zeros([1 24])
intensityRatioSTD_v3 = zeros([1 24])
intensityRatioSTD_v4 = zeros([1 24])
intensityRatioSTD_v5 = zeros([1 24])
for i = 1:24
    intensityRatioSTD_v1(i) = std(intensityRatio_v1(1:i))
    intensityRatioSTD_v2(i) = std(intensityRatio_v2(1:i))
    intensityRatioSTD_v3(i) = std(intensityRatio_v3(1:i))
    intensityRatioSTD_v4(i) = std(intensityRatio_v4(1:i))
    intensityRatioSTD_v5(i) = std(intensityRatio_v5(1:i))
end
figure()
h = plot(linspace(1,24,24),intensityRatioSTD_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioSTD_v2(1,1:24),"-o")
```

```
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
hold on
plot(linspace(1,24,24),intensityRatioSTD_v2(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v3(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v4(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v5(1,1:24),"-o")
legend("Record 1","Record 2", "Record 3", "Record 4","Record 5")
xlabel("Number of Images")
ylabel("Intensity Ratio Standard Deviation")
title("Standard Deviation versus Number of Images Plot for Sulforhodamine B Sodium
Salt (0.2 g/L)")
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'STDPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'STDPlot'), 'png')
end
hold off

figure()
intensityRatioMean_v1 = zeros([1 24])
intensityRatioMean_v2 = zeros([1 24])
intensityRatioMean_v3 = zeros([1 24])
intensityRatioMean_v4 = zeros([1 24])
intensityRatioMean_v5 = zeros([1 24])
for i = 1:24
    intensityRatioMean_v1(i) = mean(intensityRatio_v1(1:i))
    intensityRatioMean_v2(i) = mean(intensityRatio_v2(1:i))
    intensityRatioMean_v3(i) = mean(intensityRatio_v3(1:i))
    intensityRatioMean_v4(i) = mean(intensityRatio_v4(1:i))
    intensityRatioMean_v5(i) = mean(intensityRatio_v5(1:i))
end
h = plot(linspace(1,24,24),intensityRatioMean_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
hold on
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v3(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v4(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v5(1,1:24),"-o")
xlabel("Number of Images")
ylabel("Intensity Ratio Mean")
legend("Record 1","Record 2", "Record 3", "Record 4","Record 5")
title("Mean versus Number of Images Plot for Sulforhodamine B Sodium Salt (0.2
g/L)")
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'MeanPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'MeanPlot'), 'png')
end
hold off
figure()
intensityRatioMean_v1 = zeros([1 24])
```

```
intensityRatioMean_v12 = zeros([1 24])
intensityRatioMean_v123 = zeros([1 24])
intensityRatioMean_v1234 = zeros([1 24])
intensityRatioMean_v12345 = zeros([1 24])
for i = 1:24
    intensityRatioMean_v1(i) = mean(intensityRatio_v1(1:i))
    intensityRatioMean_v12(i)
=mean((intensityRatio_v1(1:i)+intensityRatio_v2(1:i))/2);
    intensityRatioMean_v123(i)
=mean((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i
))/3);
    intensityRatioMean_v1234(i)
=mean((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i
)+intensityRatio_v4(1:i))/4);
    intensityRatioMean_v12345(i)
=mean((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i
)+intensityRatio_v4(1:i)+intensityRatio_v5(1:i))/5);
%     intensityRatioMean_v2(i) = mean(intensityRatio_v2(1:i));
%     intensityRatioMean_v12(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i)))/2;
%     intensityRatioMean_v123(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i)))/3;
%     intensityRatioMean_v1234(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i))+mean(intensityRatio_v5(1:i)))/4;
end
h = plot(linspace(1,24,24),intensityRatioMean_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
hold on
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
plot(linspace(1,24,24),intensityRatioMean_v12(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v123(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v1234(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioMean_v12345(1,1:24),"-o")
xlabel("Number of Images")
ylabel("Intensity Ratio Mean")
legend("1 Record","2 Records", "3 Records", "4 Records", "5 Records")
title("Mean versus Number of Images for varying Number of Records Plot for
Sulforhodamine B Sodium Salt (0.2 g/L)")
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'MeanNumImagePlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'MeanNumImagePlot'), 'png')
end
hold off
figure()
intensityRatioSTD_v1 = zeros([1 24])
intensityRatioSTD_v12 = zeros([1 24])
intensityRatioSTD_v123 = zeros([1 24])
intensityRatioSTD_v1234 = zeros([1 24])
intensityRatioSTD_v12345 = zeros([1 24])
for i = 1:24
    intensityRatioSTD_v1(i) = std(intensityRatio_v1(1:i))
```

```
    intensityRatioSTD_v12(i) =
std((intensityRatio_v1(1:i)+intensityRatio_v2(1:i))/2);
    intensityRatioSTD_v123(i) =
std((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i))
/3);
    intensityRatioSTD_v1234(i) =
std((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i)+
intensityRatio_v4(1:i))/4);
    intensityRatioSTD_v12345(i) =
std((intensityRatio_v1(1:i)+intensityRatio_v2(1:i)+intensityRatio_v3(1:i)+
intensityRatio_v4(1:i)+intensityRatio_v5(1:i))/5);
%     intensityRatioMean_v2(i) = mean(intensityRatio_v2(1:i));
%     intensityRatioMean_v12(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i)))/2;
%     intensityRatioMean_v123(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i)))/3;
%     intensityRatioMean_v1234(i) =
(mean(intensityRatio_v2(1:i))+mean(intensityRatio_v3(1:i))+mean(intensityRatio_v4(1
:i))+mean(intensityRatio_v5(1:i)))/4;
end
h = plot(linspace(1,24,24),intensityRatioSTD_v1(1,1:24),"-o")
%h = plot(linspace(1,24,24),intensityRatioMean_v2(1,1:24),"-o")
hold on
ax = ancestor(h, 'axes')
ax.YAxis.Exponent = 0
ytickformat('%.4f')
plot(linspace(1,24,24),intensityRatioSTD_v12(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v123(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v1234(1,1:24),"-o")
plot(linspace(1,24,24),intensityRatioSTD_v12345(1,1:24),"-o")

xlabel("Number of Images")
ylabel("Intensity Ratio Standard Deviation")
legend("1 Record","2 Records", "3 Records", "4 Records", "5 Records")
title("Standard Deviation versus Number of Images for varying Number of Records
Plot for Sulforhodamine B Sodium Salt (0.2 g/L)")
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'STDRecordPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'STDRecordPlot'), 'png')
end
hold off
figure()
hold on
[f1,xi1] = ksdensity(intensityRatio_v1);
plot(xi1,f1)
[f2,xi2] = ksdensity(intensityRatio_v2);
plot(xi2,f2)
[f3,xi3] = ksdensity(intensityRatio_v3);
plot(xi3,f3)
[f4,xi4] = ksdensity(intensityRatio_v4);
plot(xi4,f4)
[f5,xi5] = ksdensity(intensityRatio_v5);
plot(xi5,f5)
xlabel("Fluorescnce Intensity Ratio")
```

```
ylabel("Probability Density Function")
legend("Record 1","Record 2","Record 3","Record 4","Record 5")
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'IntensityRatioPDFPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'IntensityRatioPDFPlot'), 'png')
end
hold off

intensityratio_mean = mean(intensityRatio)
intensityratio_var = var(intensityRatio)
%30 images

% y =
[intensityRatio_v1;intensityRatio_v2;intensityRatio_v3;intensityRatio_v4;intensityR
atio_v5];
% group = repelem(1:5, 1, [numel(intensityRatio_v1),
numel(intensityRatio_v2),numel(intensityRatio_v3),numel(intensityRatio_v4),numel(in
tensityRatio_v5) ])
% p = anova1(y,group)
% xlabel("Record Number")
% ylabel("Fluorescence Intensity Ratio")
% title("ANOVA Test for Two Records at Same Temperature, 24 Images")
y =
[intensityRatio_v1;intensityRatio_v2;intensityRatio_v3;intensityRatio_v4;i
ntensityRatio_v5];
group = repelem(1:5, 1,
[numel(intensityRatio_v1),numel(intensityRatio_v2),numel(intensityRatio_v3
),numel(intensityRatio_v4),numel(intensityRatio_v5) ]);
p = anova1(y,group);
xlabel("Record Number")
ylabel("Fluorescence Intensity Ratio")
title("ANOVA Test for Five Records at Same Temperature, 24 Images")
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'ANOVAPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'ANOVAPlot'), 'png')
end
recordNum = ["Record 1"; "Record 2"; "Record 3"; "Record 4"; "Record 5"; "First
2"; "First 3"; "First 4"; "All 5" ]
meanval = [intensityRatioMean_v1(24); intensityRatioMean_v2(24);
intensityRatioMean_v3(24); intensityRatioMean_v4(24);
intensityRatioMean_v5(24);intensityRatioMean_v12(24) ;
intensityRatioMean_v123(24) ; intensityRatioMean_v1234(24);
intensityRatioMean_v12345(24) ]
stdval = [intensityRatioSTD_v1(24); intensityRatioSTD_v2(24);
intensityRatioSTD_v3(24);
intensityRatioSTD_v4(24);intensityRatioSTD_v5(24);
intensityRatioSTD_v12(24); intensityRatioSTD_v123(24);
intensityRatioSTD_v1234(24); intensityRatioSTD_v12345(24)]
relative_STD = stdval./meanval
percent_STD = 100*relative_STD
percent_diff = abs(meanval -
intensityRatioMean_v12345(24))./(meanval+intensityRatioMean_v12345(24))/2*
100
results = table(recordNum, meanval, stdval, percent_STD,percent_diff)
```

```
if saveData == 1
    writetable(results,fullfile(process_dir,
'TemporalResolutionSummary.xlsx'))
end
```

Published with MATLAB® R2021b

B.1.3 Temperature Calibration Code

```
clear; close all;
%dye 1
testNumbers = [105, 106, 107, 108,
109,110,111,112,113,114,115,116,117,118,119,120,121,122,123,124,125];
tempCases =
["19.6°C","19.6°C","19.6°C","26.2°C","26.2°C","26.2°C","29.6°C","29.6°C","29.6°C",
"34.2°C","34.2°C","34.2°C","41.6°C","41.6°C","41.6°C","46.4°C","46.4°C","46.4°C","
51.8°C","51.8°C","51.8°C"];
tempNumbers = [19.6, 19.6, 19.6, 26.2, 26.2, 26.2, 29.6, 29.6, 29.6, 34.2,
34.2, 34.2, 41.6, 41.6, 41.6, 46.4, 46.4, 46.4, 51.8, 51.8, 51.8];
dye = "Pyrromethene 597-8C9 (0.20 g/L)";
img_trials = [1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2,
3,];
bg_trials = [1, 2, 3];

% %dye 2
% testNumbers = [129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141,
142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152];
% tempCases =
["23.1°C","23.1°C","23.1°C","26.9°C","26.9°C","26.9°C","29.8°C","29.8°C","29.8°C","
33.3°C","33.3°C","33.3°C","39.2°C","39.2°C","39.2°C","43.4°C","43.4°C","43.4°C","48
.1°C","48.1°C","48.1°C","53.9°C","53.9°C","53.9°C"];
% tempNumbers = [23.1, 23.1, 23.1, 26.9, 26.9, 26.9, 29.8, 29.8, 29.8, 33.3, 33.3,
33.3, 39.2, 39.2, 39.2, 43.4, 43.4, 43.4, 48.1, 48.1, 48.1, 53.9, 53.9, 53.9];
% dye = "Sulforhodamine B Sodium Salt (0.5 g/L) and Eosin Y (1.6 g/L)";
% img_trials = [1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1,
2, 3];
% bg_trials = [1, 2, 3];

Date = "2023-07-03";
Date2 = 20230703;
Date3 = 20230703;
%use average of center m x n pixels of image
numImages = 15;
m=10;
n=10;
x_loc  = 1256;
y_loc = 2160-808;
%whether or not to save plots and tables in process_dir
saveData = 1;
imagesUsed = 1;
startingImage = 1;
finalImage = imagesUsed+startingImage-1;

%data paths

images_dir = "G:\Shared drives\TFEL - LIF Thermometry #3\";
```

```
process_dir = "G:\Shared drives\TFEL - LIF Thermometry\Processed Data and
Figures\TempCalibration_" +Date+"_2\";
mkdir(process_dir)
%plot markers
markers = ["-o", "-*", "-^", "-v", "-x", "-d", "-+", "-s", "->", "-<", "-p", "-
h", "-.", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-
d"];
testNumbers = string(num2cell(testNumbers));
imageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_'+testNumbers+'_cam1_'+tempNumbers+'C_'+img_tri
als);
filePattern1 = fullfile(imageTestFolder1, '*.tif');
tifFiles1 = [];
for i = 1:length(filePattern1)
    tifFiles1 = [tifFiles1; dir(filePattern1(i))];
end
tifFilesSorted1 = natsortfiles(tifFiles1);
pixelValue1 = zeros(length(tifFilesSorted1),1);
intensityValue1 = zeros(length(tifFilesSorted1),1);

imageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_'+testNumbers+'_cam2_'+tempNumbers+'C_'+img_tri
als);
filePattern2 = fullfile(imageTestFolder2, '*.tif');
tifFiles2 = [];
for i = 1:length(filePattern2)
    tifFiles2 = [tifFiles2; dir(filePattern2(i))];
end
tifFilesSorted2 = natsortfiles(tifFiles2);
pixelValue2 = zeros(length(tifFilesSorted2),1);
intensityValue2 = zeros(length(tifFilesSorted2),1);
imageCalibFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_101_cam1_calib\'+Date3+'_101_cam1_calib_X1.tif');
imageCalibFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_101_cam2_calib\'+Date3+'_101_cam2_calib_X1.tif');
imageCalib1 = imread(imageCalibFile1);
imageCalib2 = imread(imageCalibFile2);
imageCalib1 = imresize(imageCalib1,[2160 2560]);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
%imshow(imageCalib2,[0 300])
imageTranformFromCam2to1 = imregcorr(imageCalib2,imageCalib1);
%Rfixed = imref2d(size(imageCalib1));
%imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1,'OutputView',Rfixed);
imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
bgNumber = [102, 103, 104];
bgimageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_'+bgNumber+'_cam1_water_las_'+bg_trials);
bgfilePattern1 = fullfile(bgimageTestFolder1, '*.tif');
bgtifFiles1 = [];
for i = 1:length(bgfilePattern1)
    bgtifFiles1 = [bgtifFiles1; dir(bgfilePattern1(i))];
end
bgtifFilesSorted1 = natsortfiles(bgtifFiles1);
```

```
bgpixelValue1 = zeros(length(bgtifFilesSorted1),1);
bgintensityValue1 = zeros(length(bgtifFilesSorted1),1);

bgimageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_'+bgNumber+'_cam2_water_las_'+bg_trials);
bgfilePattern2 = fullfile(bgimageTestFolder2, '*.tif');
bgtifFiles2 = [];
for i = 1:length(bgfilePattern2)
    bgtifFiles2 = [bgtifFiles2; dir(bgfilePattern2(i))];
end
bgtifFilesSorted2 = natsortfiles(bgtifFiles2);
bgpixelValue2 = zeros(length(bgtifFilesSorted2),1);
bgintensityValue2 = zeros(length(bgtifFilesSorted2),1);
%EY:
imageBackgroundFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_102_cam1_water_las\'+Date2+'_102_cam1_water_las_X1.
tif');
imageBackgroundFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_102_cam2_water_las\'+Date2+'_102_cam2_water_las_X1.
tif');
imageBackground1 = zeros([2160 2560]);
imageBackground2 = zeros([2160 2560]);
for k = 1:numImages
    bgbaseFileName1 = bgtifFilesSorted1(k).name;
    bgbaseFileName2 = bgtifFilesSorted2(k).name;
    bgfullFileName1 = fullfile(bgimageTestFolder1(floor((k+numImages-
1)/numImages)),bgbaseFileName1);
    bgfullFileName2 = fullfile(bgimageTestFolder2(floor((k+numImages-
1)/numImages)),bgbaseFileName2);
    fprintf(1, 'Now reading %s\n', bgfullFileName1);
    fprintf(1, 'Now reading %s\n', bgfullFileName2);
    imageBackground1 = imageBackground1 + double(imread(bgfullFileName1));
    imageBackground2 = imageBackground2 + double(imread(bgfullFileName2));
end

imageBackground1 = imageBackground1/(numImages*3);
imageBackground2 = imageBackground2/(numImages*3);
imageBackground2 = imwarp(imageBackground2,imageTranformFromCam2to1);
imageBackground2 = imresize(imageBackground2,[2160 2560]);
for k = 1:length(tifFilesSorted1)
    baseFileName1 = tifFilesSorted1(k).name;
    baseFileName2 = tifFilesSorted2(k).name;
    fullFileName1 = fullfile(imageTestFolder1(floor((k+numImages-
1)/numImages)),baseFileName1);
    fullFileName2 = fullfile(imageTestFolder2(floor((k+numImages-
1)/numImages)),baseFileName2);
    fprintf(1, 'Now reading %s\n', fullFileName1);
    fprintf(1, 'Now reading %s\n', fullFileName2);
    %if image is 2160 x 2560 pixels, center roughly here:
    %imageValue = imread(fullFileName,'PixelRegion',{[1072,1087],[1272,1287]});
    imageValue1 = double(imread(fullFileName1));
    imageValue2 = double(imread(fullFileName2));
    Rfixed=imref2d(size(imageValue1));
    imageValue2 = imwarp(imageValue2,imageTranformFromCam2to1);
```

```
    imageValue2 = imresize(imageValue2,[2160 2560]);
    %imageValue2 = imwarp(imageValue2, imageTranformFromCam2to1);
     if mod(m,2) == 0 && mod(n,2) == 0 %%if m and n are even, center m x n
pixels
        pixelValue1(k) = mean(imageValue1(y_loc-
floor(m/2):y_loc+floor(m/2)-1,x_loc-floor(n/2):x_loc+floor(n/2)-1) -
imageBackground1(y_loc-floor(m/2):y_loc+floor(m/2)-1,x_loc-
floor(n/2):x_loc+floor(n/2)-1),'all');
        pixelValue2(k) = mean(imageValue2(y_loc-
floor(m/2):y_loc+floor(m/2)-1,x_loc-floor(n/2):x_loc+floor(n/2)-1) -
imageBackground2(y_loc-floor(m/2):y_loc+floor(m/2)-1,x_loc-
floor(n/2):x_loc+floor(n/2)-1),'all');
    elseif mod(m,2) == 1 && mod(n,2) == 1 %%if m and n are even, center m x n
pixels
        pixelValue1(k) = mean(imageValue1(y_loc-
floor(m/2):y_loc+floor(m/2),x_loc-floor(n/2):x_loc+floor(n/2)) -
imageBackground1(y_loc-floor(m/2):y_loc+floor(m/2),x_loc-
floor(n/2):x_loc+floor(n/2)),'all');
        pixelValue2(k) = mean(imageValue2(y_loc-
floor(m/2):y_loc+floor(m/2),x_loc-floor(n/2):x_loc+floor(n/2)) -
imageBackground2(y_loc-floor(m/2):y_loc+floor(m/2),x_loc-
floor(n/2):x_loc+floor(n/2)),'all');
    elseif m == 1 && n == 1 %center 1x1 pixel
        pixelValue1(k) = mean(imageValue1(y_loc,x_loc) -
imageBackground1(y_loc,x_loc),'all');
        pixelValue2(k) = mean(imageValue2(y_loc,x_loc) -
imageBackground1(y_loc,x_loc),'all');
    end
end
intensity1 = [];
intensity2 = [];
%extract intensity values, with each column corresponding to a different
temperature
for i = 1:length(testNumbers)
    intensity1 = [intensity1, pixelValue1(1+numImages*(i-1):numImages*i)];
    intensity2 = [intensity2, pixelValue2(1+numImages*(i-1):numImages*i)];
end
intensity1 =
reshape(mean(reshape(intensity1.',3,[])),size(intensity1,2)/3,[]).';
intensity2 =
reshape(mean(reshape(intensity2.',3,[])),size(intensity2,2)/3,[]).';
%dye 1
testNumbers = [105, 108, 111,114,117,120,123];
tempCases = ["19.6°C","26.2°C","29.6°C","34.2°C","41.6°C","46.4°C","51.8°C"];
tempNumbers = [19.6, 26.2, 29.6, 34.2, 41.6, 46.4, 51.8];
figure()
hold on;
legendText = [];
for i = 1:length(testNumbers)
    plot(intensity1(:,i), markers(i))
    legendText = [legendText, tempCases(i)];
end
ylabel("Light Intensity Value");
xlabel("Image Number");
legend(legendText,"Location","best");
```

```
titleLabel = append("Plot of Light Intensity Value for Camera 1 Using Center "
+m+ " x " +n+ " pixels" +newline + Date);
title(titleLabel);
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'imreadCamera1Plot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'imreadCamera1Plot'), 'png')
end
hold off
figure()
hold on;
legendText = [];
for i = 1:length(testNumbers)
    plot(intensity2(:,i), markers(i))
    legendText = [legendText, tempCases(i)];
end
ylabel("Light Intensity Value");
xlabel("Image Number");
legend(legendText,"Location","best");
titleLabel = append("Plot of Light Intensity Value for Camera 2 Using Center "
+m+ " x " +n+ " pixels" +newline + Date);
title(titleLabel);
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'imreadCamera2Plot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'imreadCamera2Plot'), 'png')
end
hold off
intensityRatio = [];
intensityRatioAvg = [];
%extract intensity values, with each column corresponding to a different
temperature
for i = 1:length(testNumbers)
    %intensityRatio = [intensityRatio, (pixelValue1(1+numImages*(i-1):numImages*i)
- Pi_SR101(1+numImages*(i-1):numImages*i).*pixelValue2(1+numImages*(i-
1):numImages*i))./ pixelValue2(1+numImages*(i-1):numImages*i)];
    intensityRatio = [intensityRatio, pixelValue1(1+numImages*(i-
1):numImages*i)./ pixelValue2(1+numImages*(i-1):numImages*i)];
end
for i = 1:length(testNumbers)
    %intensityRatio = [intensityRatio, (pixelValue1(1+numImages*(i-1):numImages*i)
- Pi_SR101(1+numImages*(i-1):numImages*i).*pixelValue2(1+numImages*(i-
1):numImages*i))./ pixelValue2(1+numImages*(i-1):numImages*i)];
    intensityRatioAvg = [intensityRatioAvg,
mean(pixelValue1(1+numImages*(i-1):numImages*i))./
mean(pixelValue2(1+numImages*(i-1):numImages*i))];
end
figure()
hold on;
legendText = [];
for i = 1:length(testNumbers)
    plot(intensityRatio(:,i), "k")
    legendText = [legendText, tempCases(i)];
end
ylabel("Light Intensity Ratio, Camera 1 : Camera 2");
xlabel("Image Number");
legend(legendText,"Location","best");
```

```
titleLabel = append("Plot of Light Intensity Ratio Without Pi Function Using
Center " +m+ " x " +n+ " pixels" +newline + Date);
title(titleLabel);
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'intensityRatioPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'intensityRatioPlot'), 'png')
end
hold off
intensityAverage = [];
standardDeviation = [];
intensityAverage1 = [];
intensityAverage2 = [];
Min1 = [];
Min2 = [];
Max1 = [];
Max2 = [];
standardDeviation1= [];
standardDeviation2 = [];
Min = [];
Max = [];
for i = 1:length(testNumbers)
    intensityAverage = [intensityAverage, mean(intensityRatio(:,i))];
    standardDeviation = [standardDeviation, std(intensityRatio(:,i))];
    Min =[Min,min(intensityRatio(:,i))];
    Max =[Max,max(intensityRatio(:,i))];
    intensityAverage1 = [intensityAverage1, mean(intensity1(:,i))];
    intensityAverage2 = [intensityAverage2, mean(intensity2(:,i))];
    standardDeviation1 = [standardDeviation1, std(intensity1(:,i))];
    standardDeviation2 = [standardDeviation2, std(intensity2(:,i))];
    Min1 =[Min1,min(intensity1(:,i))];
    Min2 =[Min2,min(intensity2(:,i))];
    Max1 =[Max1,max(intensity1(:,i))];
    Max2 =[Max2,max(intensity2(:,i))];
    Min =[Min,min(intensityRatio(:,i))];
    Max =[Max,max(intensityRatio(:,i))];
end
stdAvg =  [];
for i = 1:length(testNumbers)
    stdAvg =
[stdAvg,sqrt((standardDeviation1(i)./intensityAverage1(i))^2+(standardDevi
ation2(i)./intensityAverage2(i))^2).*intensityRatioAvg(i)];
end
figure()
dataStruct = polyfitn(tempNumbers, intensityRatioAvg, 1);
numFitPoints = 1000; % Enough to make the plot look continuous.
xFit = linspace(min(tempNumbers), max(tempNumbers), numFitPoints);
yFit = polyval(dataStruct.Coefficients, xFit);
hold on;
plot(xFit, yFit, 'r-', 'LineWidth', 2);
grid on;
equation = ['^{I1}/_{I2} = f(T) = ',num2str(dataStruct.Coefficients(1)),'*T +
','(',num2str(dataStruct.Coefficients(2)),')'];
legendText = [equation; ];
for i = 1:length(testNumbers)
```

```
    errorbar(tempNumbers(i),intensityRatioAvg(i), stdAvg(i), stdAvg(i),
"ko")
    legendText = [legendText, tempCases(i)];
end
ylim([0.285 0.32])
legend(equation,"Location","best","FontSize",14)
ylabel("Light Intensity Value Ratio","FontSize",18)
xlabel("Temperature (°C)","FontSize",18)
title({"Plot of Fluorescence Intensity Ratio for at Various Temperatures
for" ,dye},"FontSize",24)
R2text = "R^{2} = " +num2str(dataStruct.R2);
SW = [min(xlim) min(ylim)]+[diff(xlim) diff(ylim)]*0.05;
text(SW(1), SW(2),R2text,"FontSize",14);
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'intensityRatioTemperatureAllData'),
'fig')
    saveas(gcf,fullfile(process_dir, 'intensityRatioTemperatureAllData'),
'png')
end
figure()
dataStruct = polyfitn(tempNumbers, intensityAverage, 1);
numFitPoints = 1000; % Enough to make the plot look continuous.
xFit = linspace(min(tempNumbers), max(tempNumbers), numFitPoints);
yFit = polyval(dataStruct.Coefficients, xFit);
hold on;
plot(xFit, yFit, 'r-', 'LineWidth', 2);
grid on;
equation = ['^{I1}/_{I2} = f(T) = ',num2str(dataStruct.Coefficients(1)),'*T +
',','(',num2str(dataStruct.Coefficients(2)),')'];
legendText = [equation; ];
for i = 1:length(testNumbers)
    errorbar(tempNumbers(i),intensityAverage(i), standardDeviation(i),
standardDeviation(i), "ko")
    legendText = [legendText, tempCases(i)];
end

legend(equation,"Location","best","FontSize",14)
ylabel("Light Intensity Value Ratio","FontSize",18)
xlabel("Temperature (°C)","FontSize",18)
title({"Plot of Fluorescence Intensity Ratio for at Various Temperatures
for" ,dye},"FontSize",24)
R2text = "R^{2} = " +num2str(dataStruct.R2);
SW = [min(xlim) min(ylim)]+[diff(xlim) diff(ylim)]*0.05;
text(SW(1), SW(2),R2text,"FontSize",14);
ylim([0.285 0.32])
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'intensityRatioTemperatureAllData'),
'fig')
```

```
    saveas(gcf,fullfile(process_dir, 'intensityRatioTemperatureAllData'),
'png')
end
figure()
dataStruct1 = polyfitn(tempNumbers, intensityAverage1, 1);
numFitPoints = 1000; % Enough to make the plot look continuous.
xFit = linspace(min(tempNumbers), max(tempNumbers), numFitPoints);
yFit = polyval(dataStruct1.Coefficients, xFit);
hold on;
plot(xFit, yFit, 'r-', 'LineWidth', 2);
grid on;
equation1 = ['I_{1}(T) = ',num2str(dataStruct1.Coefficients(1)),'*T +
','(',num2str(dataStruct1.Coefficients(2)),')'];
legendText = [equation1; ];
for i = 1:length(testNumbers)
    errorbar(tempNumbers(i),intensityAverage1(i), standardDeviation1(i),
standardDeviation1(i), "ro"); legend('')
end


R2text = "R^{2}_{1} = " +num2str(dataStruct1.R2);
SW = [min(xlim) min(ylim)]+[2*diff(xlim) 0.25*diff(ylim)]*0.05;
text(SW(1), SW(2),R2text,"FontSize",14);
hold on
dataStruct2 = polyfitn(tempNumbers, intensityAverage2, 1);
numFitPoints = 1000; % Enough to make the plot look continuous.
xFit = linspace(min(tempNumbers), max(tempNumbers), numFitPoints);
yFit = polyval(dataStruct2.Coefficients, xFit);
hold on;
plot(xFit, yFit, 'b-', 'LineWidth', 2);
grid on;
equation2 = ['I_{2}(T) = ',num2str(dataStruct2.Coefficients(1)),'*T +
','(',num2str(dataStruct2.Coefficients(2)),')'];
%legendText = [legendText; equation2];
for i = 1:length(testNumbers)
    errorbar(tempNumbers(i),intensityAverage2(i),  standardDeviation2(i),
standardDeviation2(i), "bo"); legend('')
end

%legend(legendText,"Orientation","Vertical")
legend(equation1,'Camera 1','','','','','','','',equation2,"Camera
2","FontSize",14)
ylabel("Camera Voltage (a.u.)","FontSize",18)
xlabel("Temperature (°C)","FontSize",18)
title({"Plot of Camera Voltage for Camera 1 and Camera 2 at Various Temperatures
for" ,dye},"FontSize",24)
R2text = "R^{2}_{2} = " +num2str(dataStruct2.R2);
SW = [min(xlim) min(ylim)]+[3*diff(xlim) diff(ylim)]*0.05;
%for camera 2 SRB
% SW = [min(xlim) min(ylim)]+[30 diff(ylim)]*0.6;
text(SW(1), SW(2),R2text,"FontSize",14);
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
```

```
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'intensityTemperatureAllData'), 'fig')
    saveas(gcf,fullfile(process_dir, 'intensityTemperatureAllData'), 'png')
end
hold off
figure()
[minTemp, minTempIndex] = min(tempNumbers);
intensityAverage1Normalized = [];
standardDeviation1Normalized = [];
intensityAverage2Normalized = [];
standardDeviation2Normalized = [];
for i = 1:length(testNumbers)
    intensityAverage1Normalized = [intensityAverage1Normalized,
mean(intensity1(:,i))/mean(intensity1(:,minTempIndex))];
    intensityAverage2Normalized = [intensityAverage2Normalized,
mean(intensity2(:,i))/mean(intensity2(:,minTempIndex))];
    standardDeviation1Normalized = [standardDeviation1Normalized,
std(intensity1(:,i)/mean(intensity1(:,minTempIndex)))];
    standardDeviation2Normalized = [standardDeviation2Normalized,
std(intensity2(:,i)/mean(intensity2(:,minTempIndex)))];
end
dataStruct1 = polyfitn(tempNumbers, intensityAverage1Normalized, 1);
numFitPoints = 1000; % Enough to make the plot look continuous.
xFit = linspace(min(tempNumbers), max(tempNumbers), numFitPoints);
yFit = polyval(dataStruct1.Coefficients, xFit);
hold on;
plot(xFit, yFit, 'r-', 'LineWidth', 2);
grid on;
equation1 = ['I_{1}(T) = ',num2str(dataStruct1.Coefficients(1)),'*T +
','(',num2str(dataStruct1.Coefficients(2)),')'];
legendText = [equation1; ];
for i = 1:length(testNumbers)
    errorbar(tempNumbers(i),intensityAverage1Normalized(i),
standardDeviation1Normalized(i), standardDeviation1Normalized(i), "ro");
legend('')
end


R2text = "R^{2}_{1} = " +num2str(dataStruct1.R2);
SW = [min(xlim) min(ylim)]+[2*diff(xlim) 0.25*diff(ylim)]*0.05;
text(SW(1), SW(2),R2text,"FontSize",14);
hold on
dataStruct2 = polyfitn(tempNumbers, intensityAverage2Normalized, 1);
numFitPoints = 1000; % Enough to make the plot look continuous.
xFit = linspace(min(tempNumbers), max(tempNumbers), numFitPoints);
yFit = polyval(dataStruct2.Coefficients, xFit);
hold on;
plot(xFit, yFit, 'b-', 'LineWidth', 2);
grid on;
equation2 = ['I_{2}(T) = ',num2str(dataStruct2.Coefficients(1)),'*T +
','(',num2str(dataStruct2.Coefficients(2)),')'];
%legendText = [legendText; equation2];
for i = 1:length(testNumbers)
```

```
    errorbar(tempNumbers(i),intensityAverage2Normalized(i),
standardDeviation2Normalized(i),  standardDeviation2Normalized(i), "bo");
legend('')
end

%legend(legendText,"Orientation","Vertical")
legend(equation1,'Camera 1','','','','','','',equation2,"Camera
2","FontSize",14)
ylabel("Camera Voltage/Camera Voltage at "
+tempCases(minTempIndex),"FontSize",18)
xlabel("Temperature (°C)","FontSize",18)
title({"Plot of Normalized Camera Voltage for Camera 1 and Camera 2 at Various
Temperatures for" ,dye},"FontSize",24)
R2text = "R^{2}_{2} = " +num2str(dataStruct2.R2);
SW = [min(xlim) min(ylim)]+[diff(xlim) diff(ylim)]*0.05;
%for camera 2 SRB
% SW = [min(xlim) min(ylim)]+[30 diff(ylim)]*0.6;
text(SW(1), SW(2),R2text,"FontSize",14);
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'intensityTemperatureNormalizedAllData'),
'fig')
    saveas(gcf,fullfile(process_dir, 'intensityTemperatureNormalizedAllData'),
'png')
end
hold off
%headers = ;
T = table( intensityAverage1', intensityAverage2',standardDeviation1',
standardDeviation2','VariableNames',["Camera 1 Intensity Average","Camera 2
Intensity Average","Camera 1 Intensity STD","Camera 2 Intensity STD"]);
%T.Properties.VariableNames = ["Camera 1 Intensity Average","Camera 2 Intensity
Average","Camera 1 Intensity STD","Camera 2 Intensity STD"];
writetable(T,process_dir+"ProcessedData.csv")
T=table(pixelValue1, pixelValue2,'VariableNames',["Camera 1 Average
Voltage","Camera 2 Average Voltage"]);
writetable(T,process_dir+"VoltageData.csv")
```

Published with MATLAB® R2021b

C.1.4 Temperature Contour Code

```
clear; close all;
%%PM 597
testNumbers = [105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116,
117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131,
132, 133, 134, 135, 136, 137, 138, 139, 140];
tempCases =
["loc1","loc1","loc1","loc2","loc2","loc2","loc3","loc3","loc3","loc4","loc4","lo
c4","loc5","loc5","loc5","loc6","loc6","loc6","loc7","loc7","loc7","loc8","loc8",
"loc8","loc9","loc9","loc9","loc10","loc10","loc10","loc11","loc11","loc11","loc12
","loc12","loc12"];
%tempNumbers = [loc1, loc1, loc1, loc2, loc2, loc2, loc3, lo];
dye = "Fluorescein Disodium Salt Hydrate (2.0 g/L)";
img_trials = [1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2,
3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3];
```

```
bg_trials = [1, 2, 3];

Date = "2023-07-13";
Date2 = 20230713;
Date3 = 20230713;
%use average of center m x n pixels of image
numImages = 15;
m=10;
n=10;
x_loc  = 1181;
y_loc = 2160-794;

y_min = 1266;
y_max = 1465;
x_min = 1081;
x_max = 1280;

%whether or not to save plots and tables in process_dir
saveData = 0;
imagesUsed = 1;
startingImage = 1;
finalImage = imagesUsed+startingImage-1;

%data paths

images_dir = "G:\Shared drives\TFEL - LIF Thermometry #3\";
process_dir = "G:\Shared drives\TFEL - LIF Thermometry\Processed Data and
Figures\" +Date+"\Two Camera Images";

%plot markers
markers = ["-o", "-*", "-^", "-v", "-x", "-d", "-+", "-s", "->", "-<", "-p", "-
h", "-."];
imageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_'+testNumbers
+'_cam1_'+tempCases+'_'+img_trials);
filePattern1 = fullfile(imageTestFolder1, '*.tif');
tifFiles1 = [];
for i = 1:length(filePattern1)
    tifFiles1 = [tifFiles1; dir(filePattern1(i))];
end
tifFilesSorted1 = natsortfiles(tifFiles1);
pixelValue1 = zeros(length(tifFilesSorted1),1);
intensityValue1 = zeros(length(tifFilesSorted1),1);

imageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_'+testNumbers+'_cam2_'+tempCases+'_'+img_trials
);
filePattern2 = fullfile(imageTestFolder2, '*.tif');
tifFiles2 = [];
for i = 1:length(filePattern2)
    tifFiles2 = [tifFiles2; dir(filePattern2(i))];
end
tifFilesSorted2 = natsortfiles(tifFiles2);
pixelValue2 = zeros(length(tifFilesSorted2),1);
```

```
intensityValue2 = zeros(length(tifFilesSorted2),1);
imageCalibFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_101_cam1_calib\'+Date3+'_101_cam1_calib_X1.tif');
imageCalibFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_101_cam2_calib\'+Date3+'_101_cam2_calib_X1.tif');
imageCalib1 = imread(imageCalibFile1);
imageCalib2 = imread(imageCalibFile2);
imageCalib1 = imresize(imageCalib1,[2160 2560]);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
%imshow(imageCalib2,[0 300])
imageTranformFromCam2to1 = imregcorr(imageCalib2,imageCalib1);
%Rfixed = imref2d(size(imageCalib1));
%imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1,'OutputView',Rfixed);
imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
bgNumber = [102, 103, 104];
bgimageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_'+bgNumber+'_cam1_water_las_'+bg_trials);
bgfilePattern1 = fullfile(bgimageTestFolder1, '*.tif');
bgtifFiles1 = [];
for i = 1:length(bgfilePattern1)
    bgtifFiles1 = [bgtifFiles1; dir(bgfilePattern1(i))];
end
bgtifFilesSorted1 = natsortfiles(bgtifFiles1);
bgpixelValue1 = zeros(length(bgtifFilesSorted1),1);
bgintensityValue1 = zeros(length(bgtifFilesSorted1),1);

bgimageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_'+bgNumber+'_cam2_water_las_'+bg_trials);
bgfilePattern2 = fullfile(bgimageTestFolder2, '*.tif');
bgtifFiles2 = [];
for i = 1:length(bgfilePattern2)
    bgtifFiles2 = [bgtifFiles2; dir(bgfilePattern2(i))];
end
bgtifFilesSorted2 = natsortfiles(bgtifFiles2);
bgpixelValue2 = zeros(length(bgtifFilesSorted2),1);
bgintensityValue2 = zeros(length(bgtifFilesSorted2),1);
%EY:
imageBackgroundFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_102_cam1_water_las\'+Date2+'_102_cam1_water_las_X1.
tif');
imageBackgroundFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_102_cam2_water_las\'+Date2+'_102_cam2_water_las_X1.
tif');
imageBackground1 = zeros([2160 2560]);
imageBackground2 = zeros([2160 2560]);
for k = 1:numImages
    bgbaseFileName1 = bgtifFilesSorted1(k).name;
    bgbaseFileName2 = bgtifFilesSorted2(k).name;
    bgfullFileName1 = fullfile(bgimageTestFolder1(floor((k+numImages-
1)/numImages)),bgbaseFileName1);
    bgfullFileName2 = fullfile(bgimageTestFolder2(floor((k+numImages-
1)/numImages)),bgbaseFileName2);
    fprintf(1, 'Now reading %s\n', bgfullFileName1);
```

```
    fprintf(1, 'Now reading %s\n', bgfullFileName2);
    imageBackground1 = imageBackground1 + double(imread(bgfullFileName1));
    imageBackground2 = imageBackground2 + double(imread(bgfullFileName2));
end

imageBackground1 = imageBackground1/(numImages*3);
imageBackground2 = imageBackground2/(numImages*3);
imageBackground2 = imwarp(imageBackground2,imageTranformFromCam2to1);
imageBackground2 = imresize(imageBackground2,[2160 2560]);
intensityRatio = [];
for k = 1:length(tifFilesSorted1)
    baseFileName1 = tifFilesSorted1(k).name;
    baseFileName2 = tifFilesSorted2(k).name;
    fullFileName1 = fullfile(imageTestFolder1(floor((k+numImages-
1)/numImages)),baseFileName1);
    fullFileName2 = fullfile(imageTestFolder2(floor((k+numImages-
1)/numImages)),baseFileName2);
    fprintf(1, 'Now reading %s\n', fullFileName1);
    fprintf(1, 'Now reading %s\n', fullFileName2);

    imageValue1 =
imread(fullFileName1,'PixelRegion',{[y_min,y_max],[x_min,x_max]});
    imageValue2 = imread(fullFileName2);
    Rfixed=imref2d(size(imageValue1));
    imageValue2 =
imwarp(imageValue2,imageTranformFromCam2to1,'OutputView',imref2d([2160
2560]));
    imageValue2 = imageValue2(y_min:y_max, x_min:x_max);
    imageValue1 = double(imageValue1) - imageBackground1(1:200,1:200);
    imageValue2 = double(imageValue2) - imageBackground2(1:200,1:200);
    intensityRatio(:,:,k) = imageValue1;
end
%Use non-normalized temperature calibration function
%f(T) = -0.020863*T + 6.4316
temperature_locations = [26.667, 29.092, 31.582, 34.105, 36.71, 39.45,
42.266, 45.237, 48.387, 51.651, 55.152, 58.721];
temperature = [];
for k = 1:length(testNumbers)
    temperature = [temperature, mean(((intensityRatio(:,:,(k-
1)*numImages+1:k*numImages)+27.6932)/14.914),"all")];
end
temperaturecontour = [];
for y = 1:(y_max-y_min+1)
    for x = 1:(x_max-x_min+1)
        for k = 1:length(testNumbers)
            %temperature = [temperature, (mean(intensityRatio(:,:,((k-
1)*10)+1:k*10),"all")- 6.4316)./(-0.020863)];
            temperaturecontour(y,x,k) = mean((intensityRatio(y,x,(k-
1)*numImages+1:k*numImages)+27.6932)/14.914);
        end
    end
end
tempconttemp = temperaturecontour;
temperaturecontouraverage= [];
for y = 1:(y_max-y_min+1)/m
```

```
    for x = 1:(x_max-x_min+1)/m
        for k = 1:3:length(testNumbers)
            %temperature = [temperature, (mean(intensityRatio(:,:,((k-
1)*10)+1:k*10),"all") - 6.4316)./(-0.020863)];
            temperaturecontouraverage(m*(y-1)+1:m*(y-1)+m,m*(x-1)+1:m*(x-
1)+m,(k-1)/3+1) = (mean(temperaturecontour(m*(y-1)+1:m*(y-1)+m,m*(x-
1)+1:m*(x-1)+m,k),"all")+mean(temperaturecontour(m*(y-1)+1:m*(y-1)+m,m*(x-
1)+1:m*(x-1)+m,k+1),"all")+mean(temperaturecontour(m*(y-1)+1:m*(y-
1)+m,m*(x-1)+1:m*(x-1)+m,k+2),"all"))/3;
        end
    end
end


for k = 1:12
    figure()
    surf(transpose(temperaturecontouraverage(:,:,k)),"EdgeColor","none");
    xlabel("Distance to the Right of Pixel " +x_min +" (um)")
    ylabel("Distance Below Pixel " +y_min + " (um)")
    xlim([0 (x_max-x_min+1)*0.52])
    ylim([0 (y_max-y_min+1)*0.52])
    %zlabel("Temperature (°C)")
    view(2);
    title("Temperature Contour Plot for 15 Averaged Images with 10 x 10 Pixel
Averaging at Location " +k +char(10)+" Reference Average Temperature Simulated
is: " + temperature_locations(k)+" °C" +char(10) + "for " +dye);
    colormap jet
    colorbar
end
```

Published with MATLAB® R2021b

B.1.4 Temperature Gradient Code

```
clear; close all;
%all images
testNumbers = [105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116,
117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131,
132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143];
tempCases =
["loc1","loc1","loc1","loc2","loc2","loc2","loc3","loc3","loc3","loc4","loc4","lo
c4","loc5","loc5","loc5","loc6","loc6","loc6","loc7","loc7","loc7","loc8","loc8",
"loc8","loc9","loc9","loc9","loc10","loc10","loc10","loc11","loc11","loc11","loc12
","loc12","loc12","loc13","loc13","loc13",];
%tempNumbers = [loc1, loc1, loc1, loc2, loc2, loc2, loc3, lo];
dye = "Pyrromethene 597-8C9 (0.2 g/L)";
img_trials = [1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2,
3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3];
bg_trials = [1, 2, 3];

Date = "2023-09-09";
Date2 = 20230909;
Date3 = 20230909;
Date4 = 20230705;
%use average of center m x n pixels of image
```

```
numImages = 15;
m=10;
n=10;
x_loc  = 1465;
y_loc = 2160-794;
%whether or not to save plots and tables in process_dir
saveData = 1;
imagesUsed = 1;
startingImage = 1;
finalImage = imagesUsed+startingImage-1;

%data paths

images_dir = "G:\Shared drives\TFEL - LIF Thermometry\Images\";
process_dir = "G:\Shared drives\TFEL - LIF Thermometry\Processed Data and
Figures\Gradient_" +Date+"\";
mkdir(process_dir)

%plot markers
markers = ["-o", "-*", "-^", "-v", "-x", "-d", "-+", "-s", "->", "-<", "-p", "-
h", "-.", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-d", "-
d"];
testNumbers = string(num2cell(testNumbers));
imageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_'+testNumbers
+'_cam1_'+tempCases+'_'+img_trials);
filePattern1 = fullfile(imageTestFolder1, '*.tif');
tifFiles1 = [];
for i = 1:length(filePattern1)
    tifFiles1 = [tifFiles1; dir(filePattern1(i))];
end
tifFilesSorted1 = natsortfiles(tifFiles1);
pixelValue1 = zeros(length(tifFilesSorted1),1);
intensityValue1 = zeros(length(tifFilesSorted1),1);

imageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_'+testNumbers+'_cam2_'+tempCases+'_'+img_trials
);
filePattern2 = fullfile(imageTestFolder2, '*.tif');
tifFiles2 = [];
for i = 1:length(filePattern2)
    tifFiles2 = [tifFiles2; dir(filePattern2(i))];
end
tifFilesSorted2 = natsortfiles(tifFiles2);
pixelValue2 = zeros(length(tifFilesSorted2),1);
intensityValue2 = zeros(length(tifFilesSorted2),1);
imageCalibFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date2+'_101_cam1_calib\'+Date3+'_101_cam1_calib_X1.tif');
imageCalibFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date2+'_101_cam2_calib\'+Date3+'_101_cam2_calib_X1.tif');
imageCalib1 = imread(imageCalibFile1);
imageCalib2 = imread(imageCalibFile2);
imageCalib1 = imresize(imageCalib1,[2160 2560]);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
```

```
%imshow(imageCalib2,[0 300])
imageTranformFromCam2to1 = imregcorr(imageCalib2,imageCalib1);
%Rfixed = imref2d(size(imageCalib1));
%imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1,'OutputView',Rfixed);
imageCalib2 = imwarp(imageCalib2,imageTranformFromCam2to1);
imageCalib2 = imresize(imageCalib2,[2160 2560]);
bgNumber = [102, 103, 104];
bgimageTestFolder1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date4+'_'+bgNumber+'_cam1_water_las_'+bg_trials);
bgfilePattern1 = fullfile(bgimageTestFolder1, '*.tif');
bgtifFiles1 = [];
for i = 1:length(bgfilePattern1)
    bgtifFiles1 = [bgtifFiles1; dir(bgfilePattern1(i))];
end
bgtifFilesSorted1 = natsortfiles(bgtifFiles1);
bgpixelValue1 = zeros(length(bgtifFilesSorted1),1);
bgintensityValue1 = zeros(length(bgtifFilesSorted1),1);

bgimageTestFolder2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date4+'_'+bgNumber+'_cam2_water_las_'+bg_trials);
bgfilePattern2 = fullfile(bgimageTestFolder2, '*.tif');
bgtifFiles2 = [];
for i = 1:length(bgfilePattern2)
    bgtifFiles2 = [bgtifFiles2; dir(bgfilePattern2(i))];
end
bgtifFilesSorted2 = natsortfiles(bgtifFiles2);
bgpixelValue2 = zeros(length(bgtifFilesSorted2),1);
bgintensityValue2 = zeros(length(bgtifFilesSorted2),1);
%EY:
imageBackgroundFile1 = fullfile(images_dir
+Date+'\Camera1_456\'+Date4+'_102_cam1_water_las\'+Date4+'_102_cam1_water_las_X1.
tif');
imageBackgroundFile2 = fullfile(images_dir
+Date+'\Camera2_451\'+Date4+'_102_cam2_water_las\'+Date4+'_102_cam2_water_las_X1.
tif');
imageBackground1 = zeros([2160 2560]);
imageBackground2 = zeros([2160 2560]);
for k = 1:numImages
    bgbaseFileName1 = bgtifFilesSorted1(k).name;
    bgbaseFileName2 = bgtifFilesSorted2(k).name;
    bgfullFileName1 = fullfile(bgimageTestFolder1(floor((k+numImages-
1)/numImages)),bgbaseFileName1);
    bgfullFileName2 = fullfile(bgimageTestFolder2(floor((k+numImages-
1)/numImages)),bgbaseFileName2);
    fprintf(1, 'Now reading %s\n', bgfullFileName1);
    fprintf(1, 'Now reading %s\n', bgfullFileName2);
    imageBackground1 = imageBackground1 + double(imread(bgfullFileName1));
    imageBackground2 = imageBackground2 + double(imread(bgfullFileName2));
end

imageBackground1 = 6*imageBackground1/(numImages*3);
imageBackground2 = 6*imageBackground2/(numImages*3);
imageBackground2 = imwarp(imageBackground2,imageTranformFromCam2to1);
imageBackground2 = imresize(imageBackground2,[2160 2560]);
```

```
for k = 1:length(tifFilesSorted1)
    baseFileName1 = tifFilesSorted1(k).name;
    baseFileName2 = tifFilesSorted2(k).name;
    fullFileName1 = fullfile(imageTestFolder1(floor((k+numImages-
1)/numImages)),baseFileName1);
    fullFileName2 = fullfile(imageTestFolder2(floor((k+numImages-
1)/numImages)),baseFileName2);
    fprintf(1, 'Now reading %s\n', fullFileName1);
    fprintf(1, 'Now reading %s\n', fullFileName2);
    %if image is 2160 x 2560 pixels, center roughly here:
    %imageValue = imread(fullFileName,'PixelRegion',{[1072,1087],[1272,1287]});
    imageValue1 = double(imread(fullFileName1));
    imageValue2 = double(imread(fullFileName2));
    Rfixed=imref2d(size(imageValue1));
    imageValue2 = imwarp(imageValue2,imageTranformFromCam2to1);
    imageValue2 = imresize(imageValue2,[2160 2560]);
    %imageValue2 = imwarp(imageValue2, imageTranformFromCam2to1);
     if mod(m,2) == 0 && mod(n,2) == 0 %%if m and n are even, center m x n
pixels
        pixelValue1(k) = mean(imageValue1(y_loc-
floor(m/2):y_loc+floor(m/2)-1,x_loc-floor(n/2):x_loc+floor(n/2)-1) -
imageBackground1(y_loc-floor(m/2):y_loc+floor(m/2)-1,x_loc-
floor(n/2):x_loc+floor(n/2)-1),'all');
        pixelValue2(k) = mean(imageValue2(y_loc-
floor(m/2):y_loc+floor(m/2)-1,x_loc-floor(n/2):x_loc+floor(n/2)-1) -
imageBackground2(y_loc-floor(m/2):y_loc+floor(m/2)-1,x_loc-
floor(n/2):x_loc+floor(n/2)-1),'all');
    elseif mod(m,2) == 1 && mod(n,2) == 1 %%if m and n are even, center m x n
pixels
        pixelValue1(k) = mean(imageValue1(y_loc-
floor(m/2):y_loc+floor(m/2),x_loc-floor(n/2):x_loc+floor(n/2)) -
imageBackground1(y_loc-floor(m/2):y_loc+floor(m/2),x_loc-
floor(n/2):x_loc+floor(n/2)),'all');
        pixelValue2(k) = mean(imageValue2(y_loc-
floor(m/2):y_loc+floor(m/2),x_loc-floor(n/2):x_loc+floor(n/2)) -
imageBackground2(y_loc-floor(m/2):y_loc+floor(m/2),x_loc-
floor(n/2):x_loc+floor(n/2)),'all');
    elseif m == 1 && n == 1 %center 1x1 pixel
        pixelValue1(k) = mean(imageValue1(y_loc,x_loc) -
imageBackground1(y_loc,x_loc),'all');
        pixelValue2(k) = mean(imageValue2(y_loc,x_loc) -
imageBackground1(y_loc,x_loc),'all');
    end
end
intensity1 = [];
intensity2 = [];
%extract intensity values, with each column corresponding to a different
temperature
for i = 1:length(testNumbers)
    intensity1 = [intensity1, pixelValue1(1+numImages*(i-1):numImages*i)];
    intensity2 = [intensity2, pixelValue2(1+numImages*(i-1):numImages*i)];
end
intensity1 =
reshape(mean(reshape(intensity1.',3,[])),size(intensity1,2)/3,[]).';
```

```
intensity2 =
reshape(mean(reshape(intensity2.',3,[])),size(intensity2,2)/3,[]).';
%dye 1
testNumbers = [105, 108, 111, 114, 117, 120, 123, 126,
129,132,135,138,141];
tempCases =
["loc1","loc2","loc3","loc4","loc5","loc6","loc7","loc8","loc9","loc10","loc11","
loc12","loc13"];
tempNumbers = [1,2,3,4,5,6,7,8,9,10,11,12,13];
figure()
hold on;
legendText = [];
for i = 1:length(testNumbers)
    plot(intensity1(:,i), markers(i))
    legendText = [legendText, tempCases(i)];
end
ylabel("Light Intensity Value");
xlabel("Image Number");
legend(legendText,"Location","best");
titleLabel = append("Plot of Light Intensity Value for Camera 1 Using Center "
+m+ " x " +n+ " pixels" +newline + Date);
title(titleLabel);
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'imreadCamera1Plot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'imreadCamera1Plot'), 'png')
end
hold off
figure()
hold on;
legendText = [];
for i = 1:length(testNumbers)
    plot(intensity2(:,i), markers(i))
    legendText = [legendText, tempCases(i)];
end
ylabel("Light Intensity Value");
xlabel("Image Number");
legend(legendText,"Location","best");
titleLabel = append("Plot of Light Intensity Value for Camera 2 Using Center "
+m+ " x " +n+ " pixels" +newline + Date);
title(titleLabel);
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'imreadCamera2Plot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'imreadCamera2Plot'), 'png')
end
hold off
intensityRatio = [];
intensityRatioAvg = [];
%extract intensity values, with each column corresponding to a different
temperature
for i = 1:length(testNumbers)
    %intensityRatio = [intensityRatio, (pixelValue1(1+numImages*(i-1):numImages*i)
- Pi_SR101(1+numImages*(i-1):numImages*i).*pixelValue2(1+numImages*(i-
1):numImages*i))./ pixelValue2(1+numImages*(i-1):numImages*i)];
    intensityRatio = [intensityRatio, pixelValue1(1+numImages*(i-
1):numImages*i)./ pixelValue2(1+numImages*(i-1):numImages*i)];
```

```
end
for i = 1:length(testNumbers)
    %intensityRatio = [intensityRatio, (pixelValue1(1+numImages*(i-1):numImages*i)
- Pi_SR101(1+numImages*(i-1):numImages*i).*pixelValue2(1+numImages*(i-
1):numImages*i))./ pixelValue2(1+numImages*(i-1):numImages*i)];
    intensityRatioAvg = [intensityRatioAvg,
mean(pixelValue1(1+numImages*(i-1):numImages*i))./
mean(pixelValue2(1+numImages*(i-1):numImages*i))];
end
figure()
hold on;
legendText = [];
for i = 1:length(testNumbers)
    plot(intensityRatio(:,i), "k")
    legendText = [legendText, tempCases(i)];
end
ylabel("Light Intensity Ratio, Camera 1 : Camera 2");
xlabel("Image Number");
legend(legendText,"Location","best");
titleLabel = append("Plot of Light Intensity Ratio Without Pi Function Using
Center " +m+ " x " +n+ " pixels" +newline + Date);
title(titleLabel);
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'intensityRatioPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'intensityRatioPlot'), 'png')
end
hold off
intensityAverage = [];
standardDeviation = [];
intensityAverage1 = [];
intensityAverage2 = [];
Min1 = [];
Min2 = [];
Max1 = [];
Max2 = [];
standardDeviation1= [];
standardDeviation2 = [];
Min = [];
Max = [];
for i = 1:length(testNumbers)
    intensityAverage = [intensityAverage, mean(intensityRatio(:,i))];
    standardDeviation = [standardDeviation, std(intensityRatio(:,i))];
    Min =[Min,min(intensityRatio(:,i))];
    Max =[Max,max(intensityRatio(:,i))];
    intensityAverage1 = [intensityAverage1, mean(intensity1(:,i))];
    intensityAverage2 = [intensityAverage2, mean(intensity2(:,i))];
    standardDeviation1 = [standardDeviation1, std(intensity1(:,i))];
    standardDeviation2 = [standardDeviation2, std(intensity2(:,i))];
    Min1 =[Min1,min(intensity1(:,i))];
    Min2 =[Min2,min(intensity2(:,i))];
    Max1 =[Max1,max(intensity1(:,i))];
    Max2 =[Max2,max(intensity2(:,i))];
    Min =[Min,min(intensityRatio(:,i))];
    Max =[Max,max(intensityRatio(:,i))];
```

```
end
stdAvg =  [];
for i = 1:length(testNumbers)
    stdAvg =
[stdAvg,sqrt((standardDeviation2(i)./intensityAverage2(i))^2+(standardDevi
ation2(i)./intensityAverage2(i))^2).*intensityRatioAvg(i)];
end
figure()
increment = (18.4-6.6)/12;
measure_locations = [6.6, 6.6+1*increment, 6.6+2*increment,
6.6+3*increment, 6.6+4*increment, 6.6+5*increment, 6.6+6*increment,
6.6+7*increment, 6.6+8*increment, 6.6+9*increment, 6.6+10*increment,
6.6+11*increment];
dataStruct2 = polyfitn(measure_locations, intensityAverage2(1:12), 1);
numFitPoints = 1000; % Enough to make the plot look continuous.
xFit = linspace(min(measure_locations), max(measure_locations),
numFitPoints);
yFit = polyval(dataStruct2.Coefficients, xFit);
hold on;
plot(xFit, yFit, 'b-', 'LineWidth', 2);
grid on;
%equation2 = ['I_{2}(T) = ',num2str(dataStruct2.Coefficients(1)),'*T +
','(',num2str(dataStruct2.Coefficients(2)),')'];
%legendText = [legendText; equation2];
for i = 1:length(testNumbers)-1
    errorbar(measure_locations(i),intensityAverage2(i),
standardDeviation2(i),  standardDeviation2(i), "bo");
end

%legend(legendText,"Orientation","Vertical")
%legend(equation2,"Camera 2","FontSize",14)
ylabel("Camera Voltage (a.u.)","FontSize",18)
xlabel("Distance from Cold Reservoir (mm)","FontSize",18)
title({"Plot of Camera Voltage for Camera 2 for Temperature
Gradient" ,dye},"FontSize",24)
R2text = "R^{2}_{2} = " +num2str(dataStruct2.R2);
SW = [min(xlim) min(ylim)]+[3*diff(xlim) diff(ylim)]*0.05;
%for camera 2 SRB
% SW = [min(xlim) min(ylim)]+[30 diff(ylim)]*0.6;
text(SW(1), SW(2),R2text,"FontSize",14);
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'intensityRatioTemperatureAllData'),
'fig')
    saveas(gcf,fullfile(process_dir, 'intensityRatioTemperatureAllData'),
'png')
end
hold off
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'VoltageGradientPlot'), 'fig')
    saveas(gcf,fullfile(process_dir, 'VoltageGradientPlot'), 'png')
end
tempAverage1 = [];
```

```
tempAverage2 = [];
tempSTD1 = [];
tempSTD2 = [];
for i = 1:length(testNumbers)
    tempAverage2 = [tempAverage2, mean((intensity2(:,i)-3839.8628)/(-
42.8568))];
    tempSTD2 = [tempSTD2, std((intensity2(:,i)-3839.8628)/(-42.8568))];

end
figure()
increment = (18.4-6.6)/12;
measure_locations = [6.6, 6.6+1*increment, 6.6+2*increment,
6.6+3*increment, 6.6+4*increment, 6.6+5*increment, 6.6+6*increment,
6.6+7*increment, 6.6+8*increment, 6.6+9*increment, 6.6+10*increment,
6.6+11*increment];
temperature_locations = [26.667, 29.092, 31.582, 34.105, 36.71, 39.45,
42.266, 45.237, 48.387, 51.651, 55.152, 58.721] ;
hold on
plot(measure_locations,temperature_locations,"-o")
dataStruct2 = polyfitn(measure_locations, tempAverage2(1:12), 1);
numFitPoints = 1000; % Enough to make the plot look continuous.
xFit = linspace(min(measure_locations), max(measure_locations),
numFitPoints);
yFit = polyval(dataStruct2.Coefficients, xFit);
hold on;
plot(xFit, yFit, 'b-', 'LineWidth', 2);
grid on;
%equation2 = ['I_{1}(T) = ',num2str(dataStruct2.Coefficients(1)),'*T +
','(',num2str(dataStruct2.Coefficients(2)),')'];
%legendText = [equation2; ];
for i = 1:(length(testNumbers)-1)
    errorbar(measure_locations(i),tempAverage2(i), tempSTD2(i),
tempSTD2(i), "bo");
end


R2text = "R^{2}_{1} = " +num2str(dataStruct2.R2);
SW = [min(xlim) min(ylim)]+[2*diff(xlim) 0.25*diff(ylim)]*0.05;
text(SW(1), SW(2),R2text,"FontSize",14);



%legend(legendText,"Orientation","Vertical")
%legend(equation2,"Camera 2","FontSize",14)
ylabel("Temperature (°C)","FontSize",18)
xlabel("Distance from Cold Reservoir (mm)","FontSize",18)
title({"Plot of Temperature for Camera 2 for Temperature
Gradient" ,dye},"FontSize",24)
ax = gca
ax.XAxis.FontSize = 14;
ax.YAxis.FontSize = 14;
if saveData == 1
    saveas(gcf,fullfile(process_dir, 'TemperatureGradient'), 'fig')
    saveas(gcf,fullfile(process_dir, 'TemperatureGradient'), 'png')
```

```
end
hold off
%headers = ;
T = table( intensityAverage1', intensityAverage2',standardDeviation1',
standardDeviation2', tempAverage2', tempSTD2','VariableNames',["Camera 1
Intensity Average","Camera 2 Intensity Average","Camera 1 Intensity STD","Camera 2
Intensity STD","Average Temperature","Temperature STD"]);
%T.Properties.VariableNames = ["Camera 1 Intensity Average","Camera 2 Intensity
Average","Camera 1 Intensity STD","Camera 2 Intensity STD","Average
Temperature","Temperature STD"];
writetable(T,process_dir+"ProcessedData.csv")
T=table(pixelValue1, pixelValue2,'VariableNames',["Camera 1 Average
Voltage","Camera 2 Average Voltage"]);
writetable(T,process_dir+"VoltageData.csv")
```

Published with MATLAB® R2021b