SINGLE CAMERA PHOTOGRAMMETRY MATLAB SOLVER DEVELOPED FOR AUTOMATION OF THE OIL INTERFEROMETRY PROCESS

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Aerospace Engineering

by

Hunter Michael Dunn

January 2018

COMMITTEE MEMBERSHIP

TITLE:	Single Camera Photogrammetry MATLAB Solver Developed for Automation of the Oil Interferometry Process
AUTHOR:	Hunter Michael Dunn
DATE SUBMITTED:	January 2018
COMMITTEE CHAIR:	David Marshall, Ph.D. Department Chair Professor of Aerospace Engineering
COMMITTEE MEMBER:	Russell Westphal, Ph.D. Professor of Mechanical Engineering
COMMITTEE MEMBER:	Aaron Drake, Ph.D. Associate Professor of Aerospace Engineering
COMMITTEE MEMBER:	Graham Doig, Ph.D. Assistant Professor of Aerospace Engineering

ABSTRACT

Single Camera Photogrammetry MATLAB Solver Developed For Automation of the Oil Interferometry Process

Hunter Michael Dunn

Over the last 20 years, Gregory G. Zilliac of the NASA AMES Research Center has been in continuous development of a fringe-imaging skin friction PC application used in oil interferometry analysis. This application, CXWIN5G, allows users to analyze propagation of oil smears across an aerodynamic surface using photogrammetry. The purpose of this thesis is to investigate the feasibility of increasing the level of automation currently found in CXWIN5G by developing a MATLAB solver capable of determining oil smear geometry with minimal user input.

There are two main automation goals of this thesis that are reflected in the core of the MATLAB solver: the determination of oil smear centerline propagation without user input and the calculation of fringe spacing without the use of fiduciary markings on the test surface. In CXWIN5G, oil smear propagation centerlines must be drawn by a user with their computer mouse. The MATLAB solver removes the necessity for this by utilizing the centroid location of each fringe as a reference for centerline propagation. The solver's ability to calculate fringe spacing without the use of fiduciary markings is a result of its ability to accurately determine the physical dimensions captured in an image. This is done by separating the camera's field of view into its pixel components and calculating the horizontal and vertical object length captured in each pixel.

Validation of the MATLAB solver's ability to define fringe propagation and fringe spacing is performed at multiple different camera positions. When the camera location is not directly overhead an oil smear the camera is in a state of skew. Camera skew is measured in degrees, and can occur in the horizontal or vertical direction. Images analyzed in this thesis feature representative hand-drawn oil smears, as well as oil smears created in the Cal Poly 3' x 4' low speed wind tunnel.

The MATLAB solver's ability to create accurate centerlines is accessed by comparing pixel coordinates of the MATLAB centerlines with pixel coordinates of centerlines created on an identical image in Microsoft Paint. During experimentation, 18 images were analyzed under both horizontal (X) and vertical (Y) skew camera conditions, with skew angles ranging from zero to 13.2 degrees. Under X-skew camera conditions the average position error between MATLAB and hand drawn centerlines is 0.6 %, while average position error under Y-skew camera conditions is 1.0 %. Fringe spacing accuracy is defined by how closely fringe spacing determined by the MATLAB solver is to fringe spacing measured by hand with a 1/16th inch ruler. Spacing analysis is performed on the same photos used in centerline determination. For X-skew camera positions, the average fringe spacing error is 6.1 %, while the average spacing error in Y-skew conditions is 4.3 %. As is discussed in later sections of this text, the X-skew fringe spacing error is artificially inflated due to human error during data collection.

ACKNOWLEDGMENTS

First and foremost I would like to acknowledge my parents, Michael and Linda Dunn, for their undying love and support during the years that I have been pursuing my degrees at Cal Poly. Their financial support during my undergraduate career made it possible for me to continue my education and see this thesis to fruition. A special thanks to my father, who despite not having a technical background, tirelessly read and edited every word of this paper. I would also like to thank my cousin A.J. Prudhomme, who purchased most of my undergraduate text books for me, your contribution was greatly appreciated and will never be forgotten. To the rest of my family in Utah and Idaho, I am thankful for your continuous support over the years that I have been pursuing my degrees.

A special thank you is extended to the Boeing Missile Defense team at Vandenberg Airforce Base and the Software Test Group at Tapestry Solutions, as they graciously provided flexible work hours and always allowed me to prioritize my classes and thesis over office work. These two jobs allowed me to obtain my graduate degree without taking on any student debt, and for that I am eternally grateful.

An extremely special thank you to Dr. Marshall, who has graciously remained my advisor despite the extended period of time that I have been a student at Cal Poly. You have always stressed the importance of learning to code, and your enthusiasm for fluid dynamics has always inspired me. I thank you for not only helping me throughout thesis, but also for exposing me to computational fluid dynamics.

To Dr. Doig, you have always been a pleasure to work with. I thank you for not only for agreeing to be on my committee, but for enthusiastically sharing your passion for computational fluid dynamics and wind tunnel testing. The class time that I have shared with you has been invaluable and will always be appreciated.

To Dr. Drake, though we have not spent much time together, your experience with oil interferometry has made you an invaluable member of this team. I have valued your continued input over the course of this thesis, and hope to work with you more in the future.

Finally to Dr. Westphal, I thank you for agreeing to be a committee member despite having never worked with me in the past. Your advice and willingness to assist in wind tunnel testing is admirable and much appreciated. Also, your generosity in allowing me to use some of your supplies during wind tunnel testing did not go unnoticed.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

English Symbols

AoV	=	Angle of View
H	=	Hypotenuse
L	=	Object Length
Z	=	Camera Height
c	=	Coefficient
f	=	Focal Length
l	=	Individual Pixel Length
n	=	Number of Pixels in one image half
q	=	Dynamic pressure
t	=	Time
X	=	Surface Distance Captured in Image
Δs	=	Fringe Spacing

Greek Symbols

α	=	Angle of View
β	=	Camera Pivot Angle
θ_i	=	Light Incidence Angle
θ_{light}	=	Light Refraction Angle
λ	=	Wavelength
μ_0	=	Absolute Viscosity

Subscripts

camera	=	Camera
cross	=	Cross-over Pixel
f	=	Friction
horz	=	Horizontal
left	=	Left Portion of Crossover Pixel
right	=	Right Portion of the Crossover Pixel
tot	=	Area Between Camera Plane and Image Center
vert	=	Vertical
∞	=	Freestream Condition
1, 2, 3	=	Numbering Convention

1.0 INTRODUCTION

The inspiration for this thesis arose from a desire to increase the level of automation in the Fringe-Imaging Skin Friction Technique (FISF) process. FISF friction determination works by analyzing the propagation of viscous oil along an aerodynamic surface when it is exposed to airflow. After oil propagation, the distance between the oil 'fringes' can be measured to determine the skin friction on the surface of the aerodynamic body. FISF analysis was utilized to determine skin friction on the joint Cal Poly/NASA AMELIA Project that began in 2009. Oil smears were photographed during the wind tunnel testing phase of the AMELIA Project, and analyzed with the use of the CXWIN5G software package developed by Gregory Z. Zilliac at NASA's Ames research center in 1999[17]. Though CXWIN5G software is effective in its current configuration, it is the goal of this thesis to explore the viability of increasing the level of automation in CXWIN5G process. It is the hope that automation of the friction determining process will save the user time, as well as increase the level of accuracy within the friction determination process.

CXWIN5G gives its users the ability to analyze FISF oil smears faster than if they were being measured by hand. Users place oil on the surface of an aerodynamic body that has been placed in a wind tunnel, with the tunnel speed and test duration defined by the viscosity of the oil used. Photos are then taken of the oil smear(s) that have propagated along the aerodynamic body during the tunnel test. Photos containing oil smears are uploaded into CXWIN5G for analysis. In CXWIN5G, users then draw line(s) through the center of fringe propagation with their computer mouse. Once the center line(s) have been drawn over the oil smear(s), an intersection algorithm is run that determines where the user-drawn lines and the fringes intersect[14]. With the positions of the fringe's known, the number of pixels between the oil fringes is calculated by CXWIN5G, giving the distance between the fringes and thus the skin friction on that portion of the aerodynamic body.

The main drawback to this software is that that the user must manually input the centerline of each fringe. This is problematic for two reasons: this method makes human error impossible to avoid, and it is time consuming. The ultimate goal of this project is twofold: to automate the process of determining the centerlines of oil smears, and to facilitate fringe spacing determination via calculating the dimensions of the test surface captured in each pixel. Implementation of these goals would result in a user saving time on two fronts. The first being that user would not need to spend the time drawing fringe centerlines with their computer mouse, and the second being that fringe spacing can be found without the use of fiduciary markings (more on these in the coming sections), The scope of this thesis extends only to flat test surfaces. Automation is performed in MATLAB and features analysis of both hand-drawn representative oil smears and oil smears obtained during wind tunnel testing. In order to understand the context of why both aforementioned goals are relevant and worth–while pursuits, the process and history of the Fringe-Imaging Skin Friction Technique (FISF) process will be discussed in the following sections.

1.1 Fringe-Imaging Skin Friction Technique (FISF)

The FISF technique is a method of determining the skin friction of a surface by quantifying the propagation of oil over the surface in the presence of airflow (most commonly via wind tunnel). A drop (or line) of oil of a known viscosity is placed on a surface of interest. Once the oil is applied to the surface, air flow is introduced and the oil distributes over the surface in the same direction as the flow that is being generated by the wind tunnel. Depending on the oil's viscosity, and the velocity of the airflow, wind tunnel testing nominally lasts between 2 and 10 minutes[17]. Once the wind tunnel has been turned off, a camera assembly is placed into the wind tunnel, and photos of the oil propagation (smear), are taken. While it is easier to remove the model from the tunnel before taking photos of the oil smears, this cannot be done for two reasons; first, the oil smears degrade rapidly, and, second, any movement of the model can create further propagation of the oil smears.

In order to enhance the visibility of oil smears in the photographs, the aerodynamic body containing the oil smears are illuminated by a quasi-monochromatic light source. This light source creates an interference pattern that is a result of light reflecting off both the upper surface of the oil and the surface of the wind tunnel model. The interference pattern created is shown below by Figure 1. The dark bands represent the areas where the light reflections off the oil and model surface destructively interfere with one another. Skin friction is determined using Equation (1); where Δs is the distance between the destructive interference bands (fringes), n_0 is the oil index of refraction, μ_0 represents the oil's absolute viscosity, t is the length of time that the oil was subjected to wind, $q_\infty$is the freestream dynamic pressure, θ_{light} is the light refraction angle through the air-to-oil interface and λ is the wavelength of the light source.

$$c_f = \frac{2n_0\mu_0\Delta s}{q_\infty\lambda t}cos(\theta_{light}) \qquad (1)$$

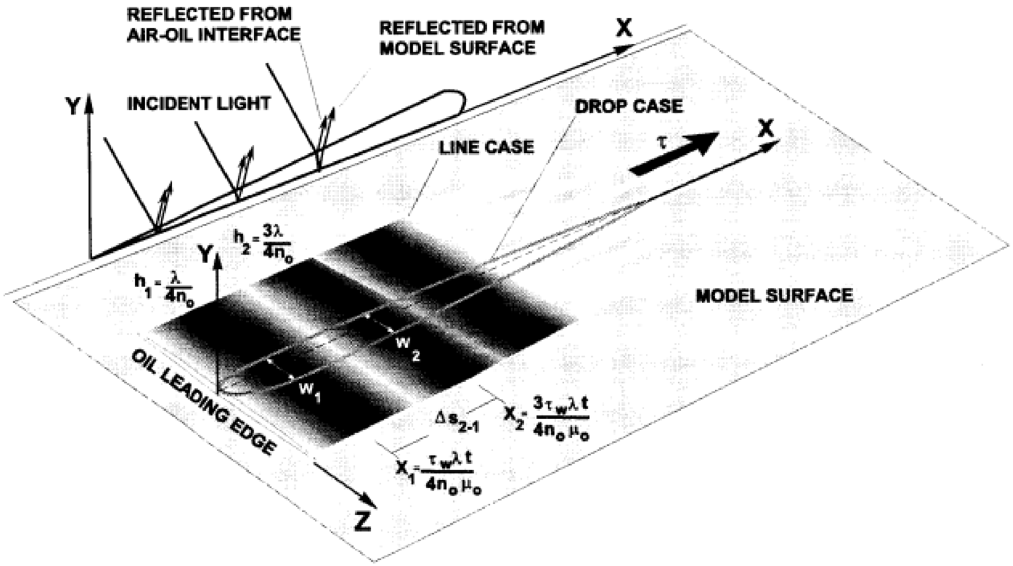

REFLECTED FROM AIR-OIL INTERFACE
REFLECTED FROM MODEL SURFACE
X
Y
INCIDENT LIGHT
DROP CASE
X
τ
LINE CASE
$h_2 = \frac{3\lambda}{4n_o}$
Y
$h_1 = \frac{\lambda}{4n_o}$
MODEL SURFACE
W_2
W_1
OIL LEADING EDGE
$\Delta s_{2\text{-}1}$
$X_2 = \frac{3\tau_w\lambda t}{4n_o\mu_o}$
$X_1 = \frac{\tau_w\lambda t}{4n_o\mu_o}$
Z

Figure 1. Fringe-Imaging Skin Friction oil flow geometry schematic, notice the fringes created by light interfering with different oil thickness[17]

For this thesis, the monochromatic light source is provided by mercury-vapor lamps. These lamps emit light at a wavelength of 546.1 nm and illuminate the oil smears in a green hue. In order to ensure that the oil smears are only illuminated by the mercury lamps, a shroud made of opaque material can be constructed that fully encapsulates the mercury lamps and the aerodynamic surface containing the oil smears. A diagram of the mercury-lamp shroud assembly is illustrated in Figure 2 below. As seen in Figure 2, the optimal placement for the mercury laps is beneath the test surface. In this configuration, the light from the lamps reflects off the surface of the diffuse reflector and illuminates the entire area under the reflector with equal-intensity light. If a shroud is being used during testing, a hole is cut in the reflective shroud that is just large enough to accommodate the camera lens. Due to the difficulties involved in cutting accurate holes in the shroud and in repositioning the camera to get images of oil smears from different angles, directly lighting the test location with lower intensity light is often favored.

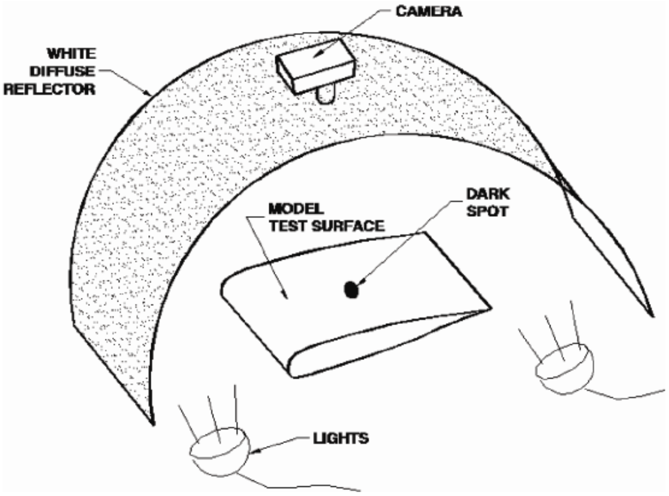


Figure 2. Mercury lamp and light shroud assembly, camera shown in an orthogonal position relative to the model, lights placed below the model oriented towards shroud.[3]

Equation (1) is a function of the light refraction angle θ_{light} because fringe spacing is dependent on the location of the light source relative to the path that the light takes through the oil. If the light source is normal to the surface, then θ_{light} is zero, resulting in the cosine term within Equation (1) equaling one. The test set up ideally used for testing incorporates the use of a reflective cover that will ensure complete and even illumination of the model surface. The light refraction angle through the air-oil interface is related to the light incidence angle on the oil by the simple geometry given in Equation (2)

$$\theta_r = asin\left({}^{sin(\theta_i)}/_{n_o} \right) \tag{2}$$

θ_i is the light incidence angle and is determined by the relationship between the position vector of the camera relative to the location of the oil smear, and the surface normal vector at the location of the oil smear. The schematic illustrating how to determine θ_i is given in Figure 3. As the MATLAB solver only deals with flat plate surfaces, if the scope of this thesis were to be extended beyond the determination of fringe centerline and fringe spacing, calculating friction would be relatively trivial, as the vector normal to the model is easily calculated.

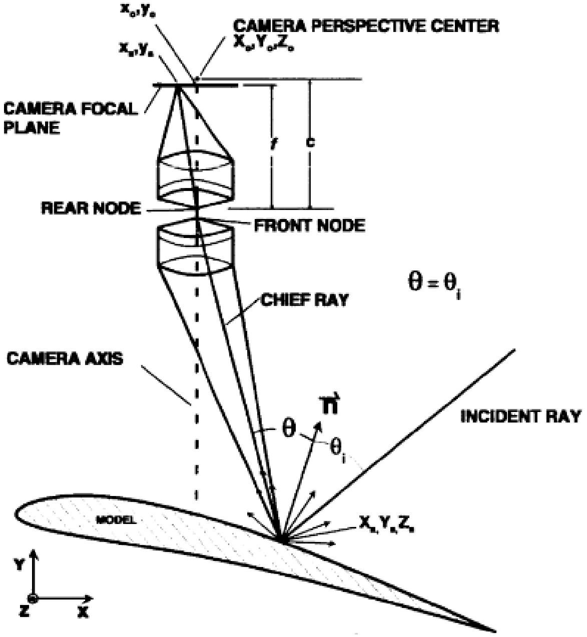


Figure 3. Optical path of camera focal array and light reflection against model surface, illustration of light incidence angle. *[3]*

1.2 History of the Oil-Fringe Skin Friction (FISF) Technique

Initial development of the FISF process applied to skin friction determination was completed in 1993 by NASA AMES researchers Daryl J. Monson, George G. Mateer and Florian R. Menter[13]. The basis of their analysis was observing the propagation of lines of oil placed along the surface of a two-dimensional wing. The trio quantified oil propagation with the use of measurement points located on the surface of the two-dimensional wing. Once oil propagation had been achieved via wind tunnel test, images of the oil smears were taken with a CCD-array digital camera so that the spacing of the oil fringes could be determined at a later date.

The development of this analysis was prompted by the shortcomings in the methods of skin friction determination in use in the early 1990s. These methods included floating-element balances, Preston tubes, surface thin-film heat-transfer gauges and pitot-tube surveys combined with the use of a Clauser chart. Mainly, these analysis methods suffer from requiring permanent installation of test equipment, containing delicate and complex components, being tedious to use or that they are intrusive to the flow. Monson, Mateer and Mentor also recognized that the laser interferometer skin friction (LISF) technique is an improvement over the previously mentioned methods. However, that being said, the LISF technique is not without its own drawbacks, as it is sensitive to dust contamination, tunnel vibration, model movement, and oil evaporation. In addition, while using the LISF technique only one measurement can be made in each tunnel run[13].

While the LISF method requires glass inserts to be fit to the model, or permanent construction of glass model sections; the FISF technique can be used on any solid surface, without the use of laser light, and at multiple model surface locations at the same time. This is advantageous because it allows for universal skin friction determination with minimal model setup and tunnel run time. The FISF process used by Monson, Mateer and Mentor was identical to what was described in the previous section; with the shear force from the tunnel flow propagating the oil at a local slope that decreased the further the oil was from its origin. Monson, Mateer and Mentor performed FISF analysis on three test surfaces: chromium-plated glass, highly polished stainless steel, and adhesive Monocote[13]. The best results were found to be produced when using chromium-plated glass and highly polished stainless steel. While ideal for testing, unfortunately the majority of wind tunnel models are not constructed from either of these two materials. Monocote, on the other hand, can be adhered to any solid, smooth surface. While is does not provide the level of fringe visibility that the other two test surfaces do, the fact that it can be adhered to most models resulted in its widespread use by the trio during their development of the FISF process. The oil used during experimentation was Dow DC-200. This oil was chosen for its 200 cSt. viscosity level, which allows for oil propagation in the presence of flow, but has high enough vapor pressure that it does not rapidly evaporate.

Using 200 cSt. oil, the trio noted that the oil fringes were visible on the test surface for up to an hour. The FISF process developed was the first to illuminate the oil smears with a monochromatic light source opposed to a laser. The monochromatic light was proven to be advantageous as it can successfully be used on curved surfaces, while a laser can only illuminate a flat surface. In order to get complete light coverage over curved surfaces, the team constructed a light box similar to what is shown in Figure 2 of this text. Green-colored light was shown to result in the highest fringe intensity. When combined with a narrow-band camera lens filter, the green monochromatic light source produced the best overall fringe visibility.

At the conclusion of experimentation, Monson, Mateer and Mentor summarized their conclusions with five statements. First, the FISF technique is easy to use, mainly due to its ability to be applied on nearly any test surface and in any type of flow. Second, due to not requiring laser light, FISF analysis does not require optical access during the tunnel run. Also, analysis can provide skin-friction distribution on any number of test surfaces, as opposed to other methods that only provide single-point measurement. Furthermore, the trio established that FISF analysis is mostly insensitive to model movement, vibration gravity, and pressure gradients; and moderately insensitive to oil evaporation, shear gradients and flow curvature. And finally, that the FISF technique can be successfully used to determine boundary-layer transition[13].

In 1996, Gregory Zilliac, also of NASA’s AMES research center, continued development of the FISF technique. Zilliac’s work focused on making three enhancements to the FISF technique: expanding the analysis capability to handle three-dimensional flow fields, refining the optical and imaging systems, and developing a PC-based application to automate the process of determining fringe spacing[16].

The key to developing three-dimensional analysis capability was the optical and imaging advancements that Zilliac pioneered during his research. The first step in this process the development of a test bed to examine the optical performance of several different light sources. The test bed featured a shop-air powered, 48-inch long, rectangular flow channel featuring interchangeable test surfaces. During experimentation, the flow channel produced a uniform skin friction coefficient (c_f) of 0.00485. The reference test surface was constructed of 0.25 inch thick SF11 Schott glass with an index of refraction of 1.78. The success of the light sources were evaluated using the intensity magnitude of the pixels contained in test images. The 8-bit camera used by Zilliac provided the capability to differentiate between 256 intensity levels, with white pixels having an intensity of zero, and completely black pixels defined as having an intensity of 256. Images were taken of oil smears illuminated by each of the light sources that were available to Zilliac. Fringe viability for each light source was quantified by looking at the ‘peak-to-peak’ intensity differential; where the top peak is the pixel intensity of an oil fringe, and the lower peak is the minimum pixel intensity of the region in-between two fringes. The peak-to-peak analysis is shown below in Figure 4.

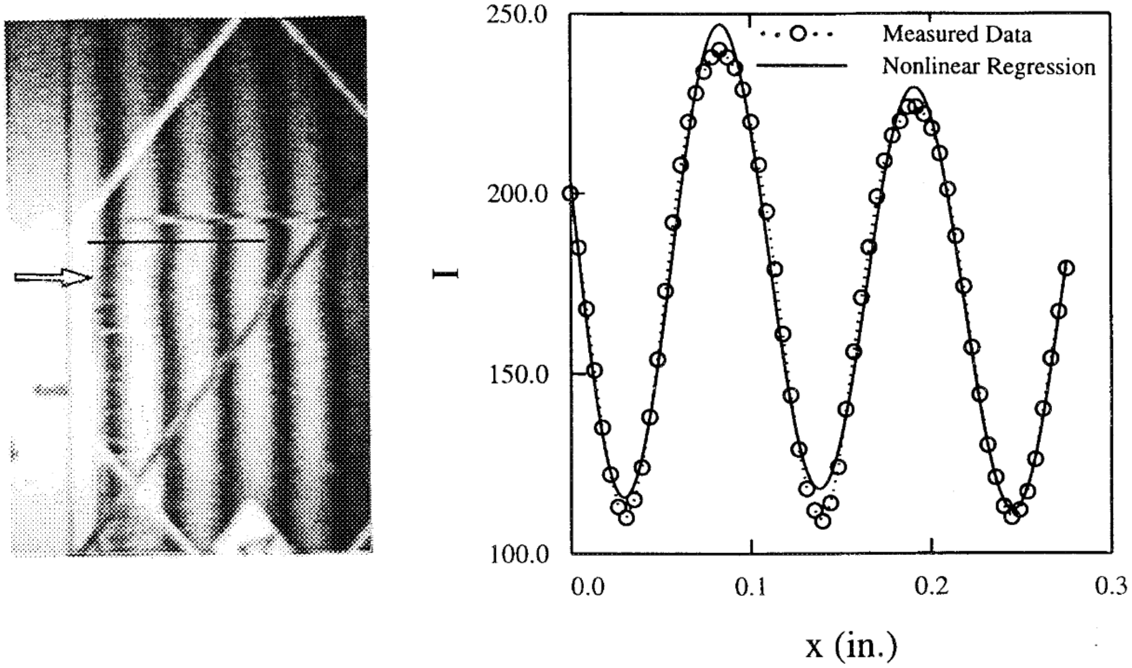


Figure 4. Peak-to-peak fringe intensity analysis applied to two-dimensional propagation of lines of oil; SF11 Schott glass[16].

The first light source used by Zilliac was a HeNe diode laser. Testing showed a promising intensity peak-to-peak value of 121. However, the laser proved to be difficult to set up, and its effectiveness was found to dramatically reduce when oil smears were analyzed over a curved surface. The second light source tested was a Philips Halogen spotlight, emitting at a 540 nm wave length. In addition to the Halogen light, Zilliac fitted a narrow band lens filter to the camera that only allowed light of a similar wavelength pass through to the camera lens. Despite the use of the lens filter, this lighting configuration only produced a peak-to-peak intensity value of 30. The light source that Zilliac settled on was a green 9-watt compact fluorescent lamp featuring a spectral peak of 542 nm, and multiple other smaller peaks ranging from 490 nm to 590 nm. Like the Halogen test, a lens filter was applied to the camera lens that reduced the effect of smaller peaks. Using the lens filter, the peak-to-peak intensity value between the first fringe and the adjacent model surface was determined to be 124[16].

The camera/light setup developed by Zilliac is shown in Figure 5 below. The camera and light chassis allowed for all the tools required for FISF analysis to be in a single, compact device. The beam splitter adjacent to the camera lens was oriented such that the monochromatic propagated normal to the model surface. In addition, the camera lens was fitted with a filter that narrowed its light absorption to a very narrow wavelength of ~542 nm, reducing interference from ambient light sources, and increasing oil fringe visibility. Another benefit to having a fixed camera and focal length is that the dimensions of the model surface captured in an image are constant across all images. In Monson, Mateer and Mentor's experimentation, measurement markers were required to analyze fringe spacing. By knowing the dimensions of the model surface captured in an image, Zilliac's PC application is then able to estimate the object length captured in each pixel fairly accurately. With the pixel object lengths known, Zilliac's PC application could then determine fringe spacing by summing the pixel lengths between each spectral peak[16].

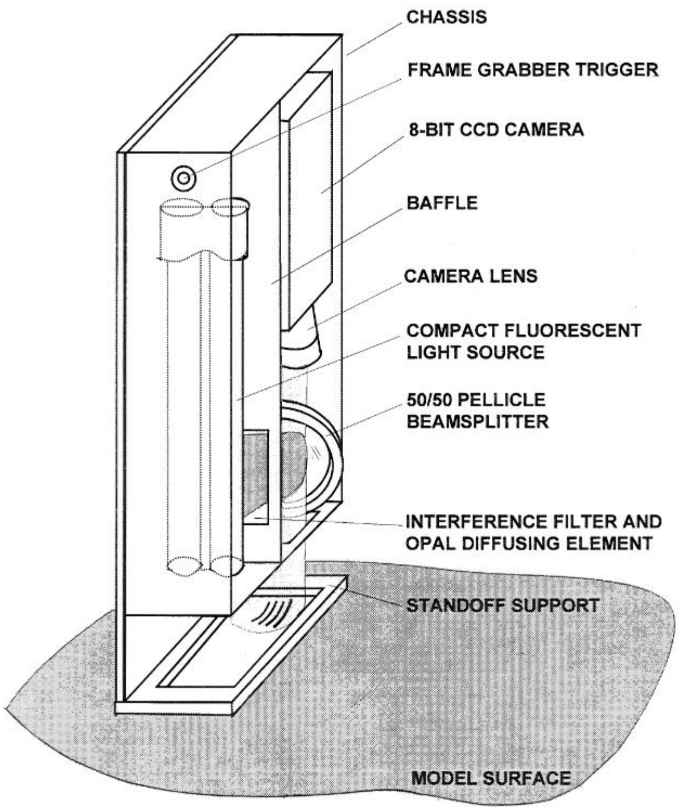

CHASSIS
FRAME GRABBER TRIGGER
8-BIT CCD CAMERA
BAFFLE
CAMERA LENS
COMPACT FLUORESCENT LIGHT SOURCE
50/50 PELLICLE BEAMSPLITTER
INTERFERENCE FILTER AND OPAL DIFFUSING ELEMENT
STANDOFF SUPPORT
MODEL SURFACE

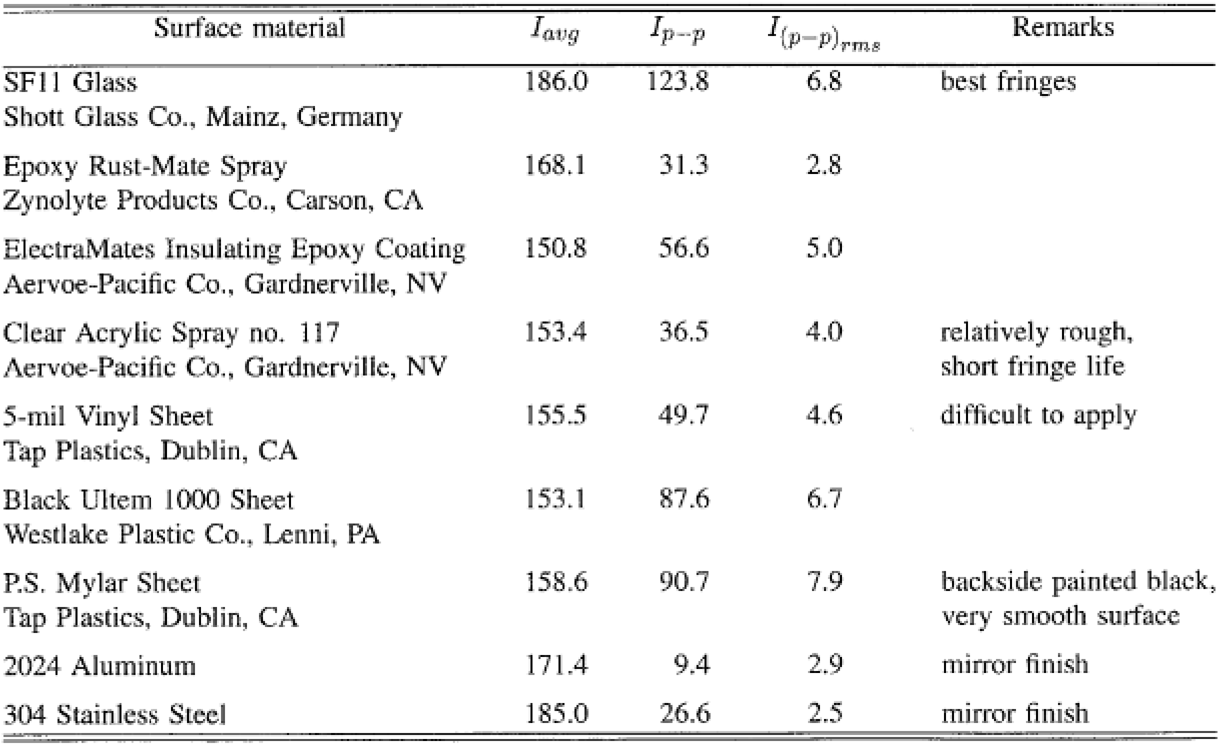
Figure 5. Camera/light chassis developed by Gregory Zilliac for capturing images of oil propagation over a model surface[16]

The camera used by Zilliac contains a square digital sensor, with only 512 pixels in both the vertical and horizontal direction. Due to the relatively poor camera resolution, each oil fringe is captured by only 15-60 pixels. As the oil propagates from its origin, the fringes become less defined, limiting Zilliac's PC application to performing fringe spacing analysis of only the first two fringes.

Several different methods of fringe identification were attempted by Zilliac during the initial creation of his PC application. These included fast Fourier transforms, linear least-squares curve fitting, maximum entropy determination and nonlinear least squares regression. In terms of the fast Fourier transforms and max energy methods, pixel density proved to be too poor to obtain the required spectral peaks. Determination of fringe spacing from identifying their physical characteristics failed due to image noise and poor fringe resolution. As mentioned in previous paragraphs, and shown in Figure 4, Zilliac decided that using the pixel-intensity detection capabilities of the camera’s 8-bit sensor was the most effective means of analysis to move forward with. The intensity of each pixel in relation to a pre-determined coordinate plane originating at the lower-left corner of each image is plotted as a function of horizontal distance. A user-generated horizontal line is shown in the native image portion of Figure 4. The pixel intensity plot is depicts the intensity of the pixels that lie along the user-generated line. Unlike the MATLAB solver discussed in this thesis, Zilliac’s PC application works under the assumption that each pixel captures an equal portion of model’s surface. Because the camera is only a few inches from the oil smear and is oriented orthogonal to the model’s surface, this pixel assumption likely did not result in much error.

Another component of Zilliac’s analysis was an investigation into how different test surfaces affect fringe visibility. This analysis looked at oil propagation over nine different test surfaces. Testing was performed using a line of 50 cSt. oil in the same flow channel used for light source testing. With this test setup Zilliac was able to produce c_f values within five percent of one another. The results of Zilliac’s intensity analysis are given below in Table 1.

Table 1. Fringe Intensity characteristics of various surface materials[16]



Surface material	I_{avg}	I_{p-p}	$I_{(p-p)_{rms}}$	Remarks
SF11 Glass Shott Glass Co., Mainz, Germany	186.0	123.8	6.8	best fringes
Epoxy Rust-Mate Spray Zynolyte Products Co., Carson, CA	168.1	31.3	2.8	
ElectraMates Insulating Epoxy Coating Aervoe-Pacific Co., Gardnerville, NV	150.8	56.6	5.0	
Clear Acrylic Spray no. 117 Aervoe-Pacific Co., Gardnerville, NV	153.4	36.5	4.0	relatively rough, short fringe life
5-mil Vinyl Sheet Tap Plastics, Dublin, CA	155.5	49.7	4.6	difficult to apply
Black Ultem 1000 Sheet Westlake Plastic Co., Lenni, PA	153.1	87.6	6.7	
P.S. Mylar Sheet Tap Plastics, Dublin, CA	158.6	90.7	7.9	backside painted black, very smooth surface
2024 Aluminum	171.4	9.4	2.9	mirror finish
304 Stainless Steel	185.0	26.6	2.5	mirror finish

The most significant measurement taken by Zilliac is given in the second intensity column of Table 1. This column gives the max intensity differential between fringe pixels and the pixels of the inter-fringe areas. The higher the intensity differential, the more visible the fringes are on a model surface. As was found by Monson, Mateer and Mentor, Zilliac's analysis also showed that SF11 glass is the ideal surface for FISF testing. While ideal, SF11 is rarely used, as few wind tunnel models are constructed of this material. The best spray coating was shown to be ElectraMates Epoxy coating. In terms of plastic sheet material, Ultem 100 and Mylar were approximately equal. Polished stainless steel without any adhesive or paint coating was found to perform quite poorly as an FISF test surface, producing a maximum intensity differential of less than 27. Though his analysis showed marginal peak-to-peak intensity using Mylar plastic coating, he noted a Mylar surface can be improved by heat treating the model surface once the Mylar has been adhered. By heating the Mylar to its melting point (~650 degrees Fahrenheit), surface imperfections can be smoothed, increasing theoretical fringe visibility.

Zilliac also concluded that the FISF method is relatively insensitive to the effects of pressure gradients. In the most extreme case Zilliac performed, the area surrounding the oil smear was subjected to a pressure gradient (dC_p/dx) of -0.42/in. After reproducing a similar oil smear subjected to zero pressure gradient, Zilliac determined that the impact of the pressure gradient was only 0.14% of c_f.

Zilliac's PC application was also the first to allow for the analysis of three-dimensional oil propagation. Three-dimensional propagation is most common when oil is placed on surfaces with a high degree of surface curvature. Three-dimensional oil propagation is determined with the use of oil drops opposed to oil lines. Drops of oil are advantageous in this case because their propagation makes observing the direction of the skin-friction vector fairly straightforward. In two-dimension oil propagation, Zilliac measured a noticeable difference in skin friction results between lines and drops of oil. However, in three-dimensional oil propagation Zilliac measured no discernable difference in skin friction between lines and drops of oil. Zilliac's PC application could only determine the length of the model captured by each pixel in one dimension. For testing applications that did not produce oil smears that propagate horizontally across the resultant image fiduciary markings had to be placed on the surface of the model adjacent to the oil smears before testing occurred. The fiducial markings were made on a flat-bed pen plotter and defined a 12x16 inch grid on the surface of the model. An example of these grid lines is given below in Figure 6. Because the spacing between the fiduciary markings are known, the pixel location of the each fringe and the markings (identified by their intensity peaks) can be compared in order to obtain fringe spacing.

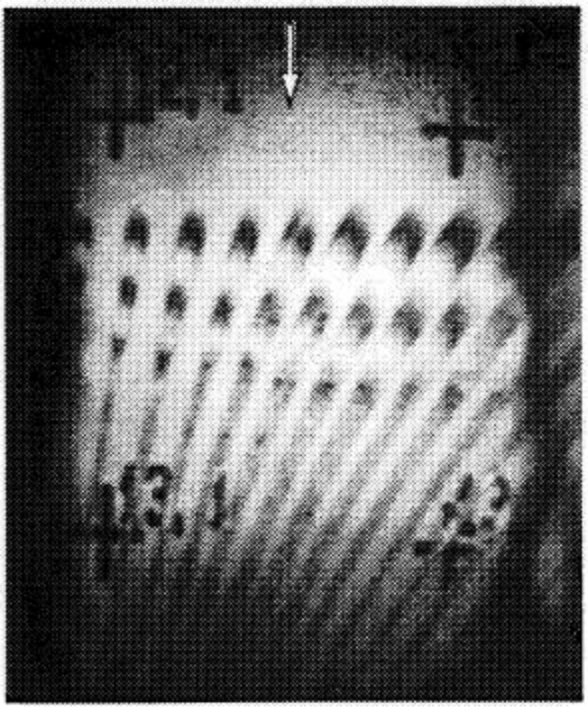
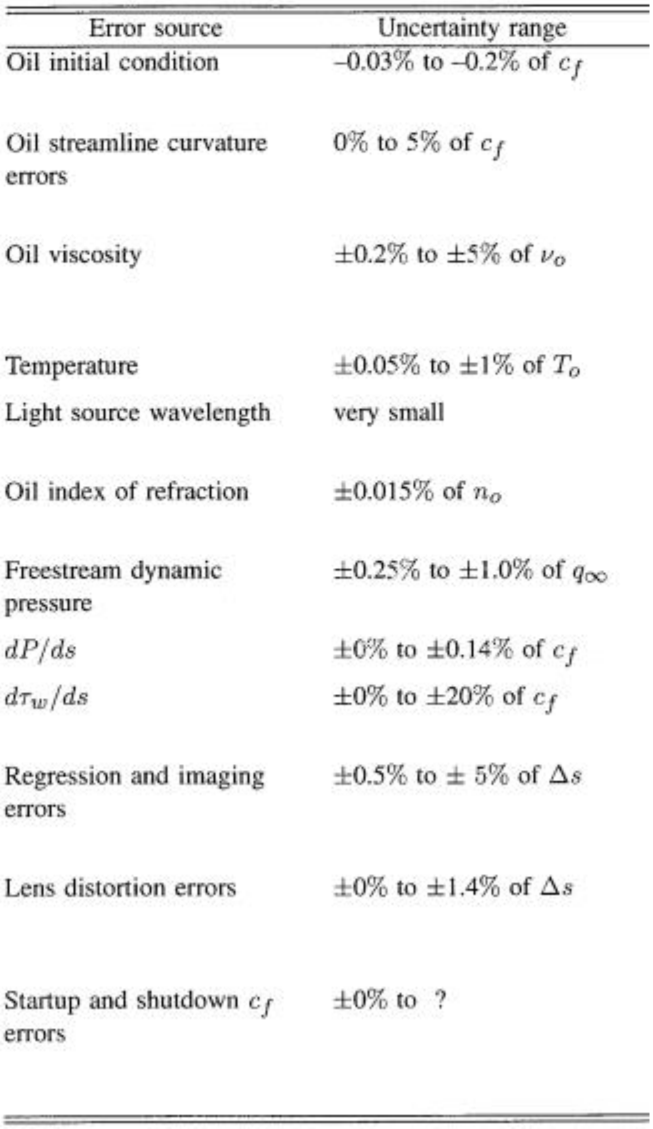


Figure 6. Four fiduciary markings shown on a model surface with three-dimensional oil smear propagation. Flow direction from image top to image bottom[16]

Zilliac separated the accuracy of his PC application into two categories: c_f magnitude uncertainty and c_f direction uncertainty. His regression error experimentation yielded no surprises, as it showed that increased fringe visibility and increased fringe resolution result in higher c_f accuracy. His results showed data accuracy of 2.1% assuming 30 pixels/fringe resolution on an epoxy-based model surface, 1% accuracy assuming 25 pixels/fringe resolution on a Mylar-based test surface and 0.5% accuracy on highly reflective glass surfaces. Zilliac validated his regression testing by using his PC application to analyze a greyscale sin wave with a peak-to-peak intensity variation of 100. Over 15 trials, the PC application replicated the sin wave to 1.15% accuracy. Table 2 below is a compilation of the error encountered by Zilliac during the development of the FISF PC application. As the propagation of oil is a function of surface shear, it is not surprising that the most significant source of error is shear gradients. As Zilliac's method only uses one location to obtain a shear force reference, any deviation in the shear force along the path of the oil will result in error.

Table 2. FISF error sources, accumulated by Zilliac in 1993[16]



Error source	Uncertainty range
Oil initial condition	–0.03% to –0.2% of c_f
Oil streamline curvature errors	0% to 5% of c_f
Oil viscosity	±0.2% to ±5% of ν_o
Temperature	±0.05% to ±1% of T_o
Light source wavelength	very small
Oil index of refraction	±0.015% of n_o
Freestream dynamic pressure	±0.25% to ±1.0% of q_∞
dP/ds	±0% to ±0.14% of c_f
$d\tau_w/ds$	±0% to ±20% of c_f
Regression and imaging errors	±0.5% to ± 5% of Δs
Lens distortion errors	±0% to ±1.4% of Δs
Startup and shutdown c_f errors	±0% to ?

One method of reducing this error that mentioned by Zilliac is to use a reliable CFD code to obtain the shear forces at multiple locations along the path of the oil propagation. The error associated with lens distortion is largely caused by the inability of the PC application to determine the dimensions of the test surface captured by each individual pixel. Regression and imaging error refers to the degradation of oil fringe quality, and imperfections in camera setup. Obtaining accurate fringe spacing is often made difficult by the imperfect nature of the fringes, as their dark bands often ‘bleed’, making their precise location difficult to obtain by Zilliac’s PC application. Furthermore, the precision of the images is dependent on the camera assembly being orthogonal to the oil smear when the photo is taken. On curved surfaces (such as wings), ensuring complete orthogonality of the camera can be difficult.

In his final thoughts, Zilliac proposes that using a high gloss paint over a polished metal is the most practical method of performing FISF analysis, and that analysis should begin no later than 48 hours after the paint has been applied. This ensures that testing begins while the paint is still at its peak gloss. He also notes that removing all of the oil residue is quite difficult to do, so subsequent tests should always be performed upstream, as to reduce the possibility of cross oil flow contamination. One of the more difficult aspect of testing is limiting oil contamination once it is applied to the surface of the model. The most obvious contaminant is dust. Dust contamination can be mitigated by running the tunnel at a higher speed than what testing will be performed at for several seconds, clearing the tunnel of contaminants that would appear during tunnel run-up. More difficult to address is contamination from water vapor condensation and tunnel component lubricating oil. When testing at high tunnel speeds, the surface of the model can cool to sub-freezing temperatures. When the tunnel doors are opened to obtain photographs of the oil smears, the water vapor in the air can rapidly condense on the surface of the model and dissipate the oil. In terms of contamination from lubricating oil, Zilliac found that minor adjustments to the position of the model can mitigate these issues. At the completion of experimentation and code development, Zilliac designated his PC application CXWIN1G[16]. Zilliac continued to expand the capabilities of his PC application in the years following its initial release. The final revision, CXWIN5G, will be discussed in the next section of this text.

1.3 CXWIN5G Overview

Between 1996 and 2005, Gregory Zilliac updated the CXWIN1G PC application with four major software releases. The final release, and the version that is currently NASA’s standard for FISF analysis is called CXWIN5G. CXWIN5G is a nine-parameter model that is fit to the fringe-intensity distribution using a nonlinear regression algorithm. The algorithm makes allowances for surface curvature, optical imperfections and nonuniform lighting. The skin friction coefficient is determined using fringe spacing calculated from pixel-based fringe locations. The pixel location of the fringes are translated into physical locations on the test model surface, thereby yielding accurate fringe spacing over the model's surface. There are several advancements to CXWIN5G compared to the initial release of CXWIN1G. The most obvious being that a single image can contain multiple oil smears to be analyzed. Also, the analysis capability over curved surfaces is dramatically increased. Unfortunately, the greater capability came at the expense of user involvement. In order to obtain images of multiple oil smears, camera placement relative to the model must vary. Like the three-dimensional oil analysis in CXWIN1G, all analysis performed in CXWIN5G requires the use of fiduciary markings. Oil smear analysis is performed largely in the same way that was in CXWIN1G, with the intensity of the pixels used to determine the pixel location of each fringe. Also similar, is that the user must select the path of intensity analysis by drawing a line down the center of each oil smear with their computer mouse.

The process of using CXWIN5G begins with a user loading an image into the program. Once the image of the oil smear(s) has been uploaded, information regarding the test environment is fed into CXWIN5G by the user. This information includes free stream dynamic pressure, oil temperature, the total time the oil smear was subjected to air flow, as well as the wavelength of the light illuminating the oil smears when the photographs were taken. The aspect ratio of the camera's sensor, its position relative to the test surface, and the focal length of the camera's lens all must be input into CXWIN5G as well[17]

The next step in the process is calibration of the image. Over a curved surface such as a wing, CXWIN5G requires three-dimensional accuracy. This accuracy is obtained with the use of fiduciary markings on the test surface. For each image, a minimum of six markings are required for analysis, with 12 recommended by Zilliac. Each fiduciary's X, Y, and Z position location relative to the camera is recorded by the user and input manually to the CXWIN5G GUI. During calibration, the algorithm is able to report surface distance to 5 percent accuracy. The fiducial markings can be seen below in Figure 7. They can either be etched into the model surface or placed on the surface via sticker. Stickers often result in greater boundary layer disruption, but their placement is much simpler than etchings.

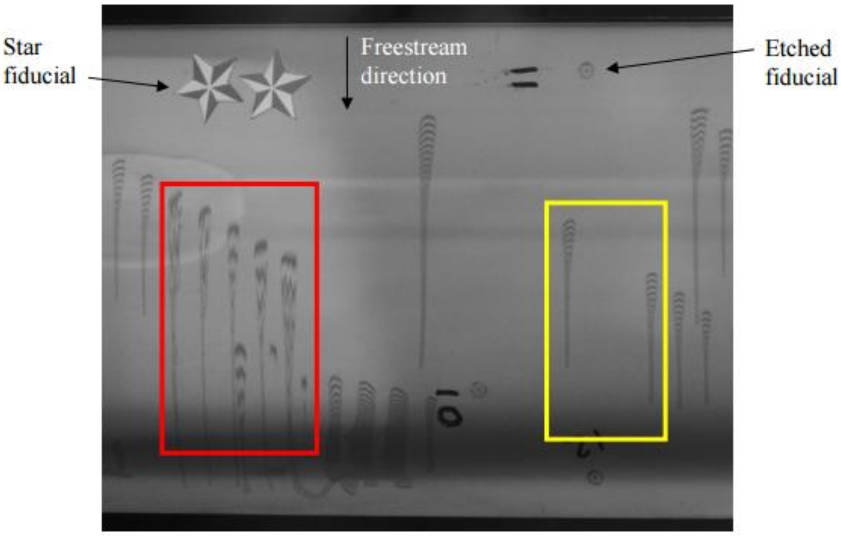


Figure 7. Sticker and etched fiducial markings comparison, etched fiducial markings result in a less turbulent boundary layer than compared to sticker markings.*[3]*

Next, the user must manually, with their computer mouse, draw the centerline of propagation for every oil smear captured in the image. Once the lines are drawn, an intersection function is run that determines where the boundary of each fringe intersects the centerline[14]. With the intersection points known, the number of pixels between each fringe is determined. With the distance and number of pixels between each fiduciary marker known, fringe spacing is determined by calculating the number of pixels between the fringes, and comparing the fringe pixel-spacing to the spacing of adjacent fiduciary markings. Up to six points can be used to model the propagation of every centerline, however the centerline analysis is limited to second-order equations. Fringe spacing is nominally measured from the leading edge of one fringe to the leading edge of an adjacent fringe. While a single image may contain more than one oil smear, analysis can only be done one fringe at a time. As a result, the process of drawing lines and extracting spacing data can be quite lengthy. Figure 8, below, depicts the hand-drawn center line over oil smear.

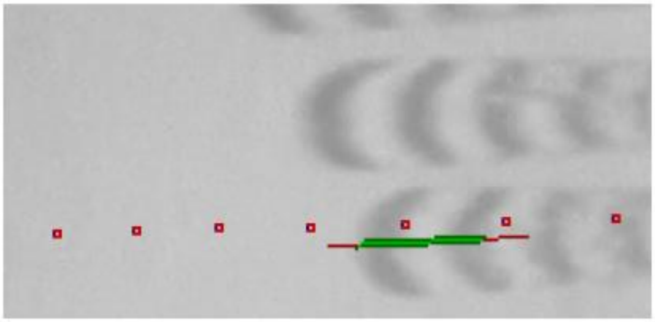


Figure 8. Hand drawn oil smear centerlines in CXWIN5G, user draws the green and red line and CXWIN5G determines where the line intersects with the oil smear fringes.[3]

1.4 MATLAB Solver Overview

The goal of the MATLAB solver is to obtain accurate fringe centerline approximations and total fringe spacing while still maintaining a high level of automation. While these two goals are interconnected to a degree, their significance and implementation are separate from one another, as both sections of the MATLAB solver are designed such that either can be implemented into current FISF solvers independently of the other. The fundamental difference between oil smear centerline determination in the MATLAB solver and Zilliac's PC applications is the higher level of automation afforded to users of the MATLAB solver. Using the solver outlined in this thesis, a user only needs to upload an image of an oil smear in order to obtain a centerline. This is in contrast to CXWIN5G, which requires it's users to manually draw centerlines over each oil smear. Automating the centerline process has two distinct advantages, it removes the possibility of human error from untrained users and it expedites the centerline creation process.

The fringe determination portion of the MATLAB solver also provides advancements over CXWIN5G. Like the centerline determination portion of the code, the MATLAB solver's fringe determination aims to lower the user's workload. This is achieved by altering the method by which fringe spacing is measured. By determining the dimensions of the test surface captured by each pixel of an image, the solver can operate using less external data than what is required for Zilliac's CXWIN5G PC application. Specifically, the external data refers to the fiduciary markings required in CXWIN5G. Because the dimensions of the test surface captured in each pixel are determined by the MATLAB solver, fiduciary markings are not required. The absence of fiduciary markings has the potential to save users large amounts of time in model prep and experimental setup. The proceeding paragraphs and sections give a detailed overview of the MATLAB solver. Because the solver was written in a manner that integrated centerline determination and fringe spacing determination into one package, it is presented as such in this paper. It is important to remember, however, that the solver is highly modular; with image analysis, centerline determination, and fringe spacing functions that can be implemented independently of one another.

The MATLAB solver is able to analyze the oil smears propagating over a smooth, flat surface, regardless the direction of the oil's propagation. Analysis begins with a user uploading a picture of an oil smear into the MATLAB script and providing the solver with relevant camera information. Camera information includes the dimensions of the camera's sensor, the vertical distance (Z), in inches, from the camera lens assembly to the test surface, as well as the skew (or pivot) angle of the camera. The angle of skew is defined as the angle created as the camera is rotated so that the center of the field of view is at the center of the oil smear propagation, as shown in Figure 9 below.

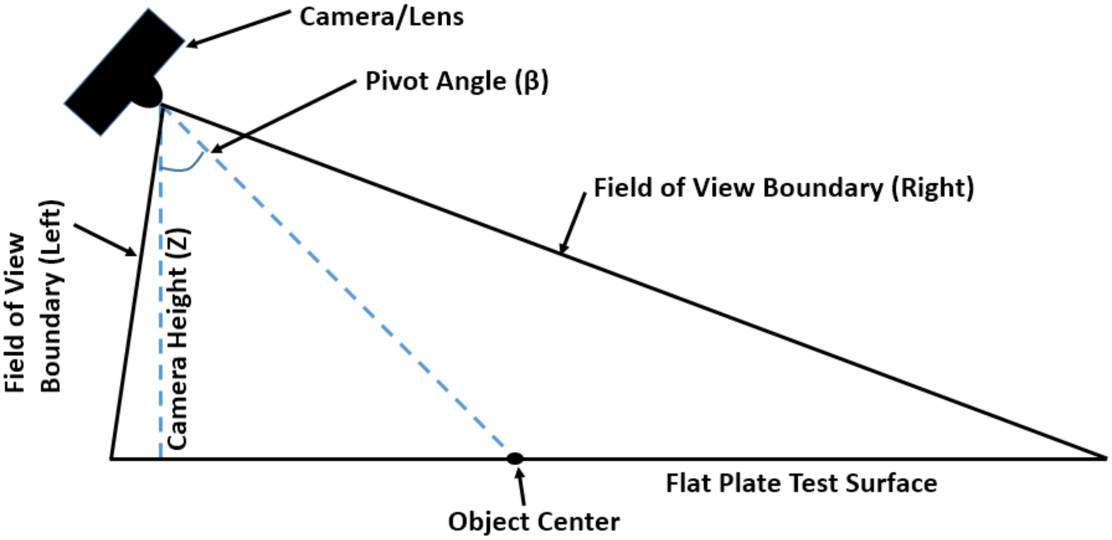


Figure 9. Basic camera skew (pivot angle) geometry, defined as the angle between the vertical plane extending from the lens focal point and the center of the lens field of view.

In its current state, the MATLAB solver requires that some constraints be adhered to with respect to the camera and test surface geometry. First, the test surface is restricted to flat geometries at this time. While it was the original intent to develop the ability to analyze more complex geometries, time constraints limit the scope of the solver at this time. The second constraint on the solver is that camera skew can only occur in one-dimension per photograph. This is not a major constraint however, as two-dimensional camera skew would never need to occur in the camera-tripod set up used in this thesis. The addition of a two-dimensional skew capability will be considered should the solver be adapted to more complex geometries in the future.

The first portion of the solver is devoted to refining the native image uploaded by a user and the creation of centerline points on each fringe. Once the user uploads an image for analysis, the first goal of the solver is to determine the centroid of each oil smear fringe. Centroidal analysis in MATLAB requires that the native red-green-blue image be translated into a binary image. Binary image creation is achieved with the use of MATLAB's powerful image toolbox. Translation to a binary image results in the isolation of the fringe boundary edges. With each fringe edge isolated, the centroids of each fringe are determined by analyzing the surface area of each fringe's interior. With the locations of the fringe centroids, centerline creation is made using the centroids as position markers.

The second portion of the MATLAB solver is dedicated to determining the geometry of the image propagation. This involves the calculation of the horizontal and vertical test surface dimensions, in inches. This is done so the individual pixel lengths can be verified later in the solver, as the sum of the individual pixel lengths should match the length of the flat test surface captured in the image. There are three possible camera-flat plate geometries that can occur during testing, depending on the skew angle of the camera lens set up and lens focal length.

When there is zero camera skew in the system the camera directly overhead the oil smear and the first of the three geometries occur. Under this geometry, the focal arrays emit in a symmetrical manner; with the field of view to the left of the oil smear identical to the field of to the right of the oil smear and with the field of view on the top of the oil smear identical to the bottom half of the image. The second of the three possible geometries occur when camera pivot angle is greater than half the angle of view. In this geometry, the image half that is closest to the camera captures less geometrical area of the test surface than the image half furthest from the camera. The most important distinction between this camera geometry and the third one is that the image does not capture the area directly under, or ‘behind’, the camera. Lastly, when the camera skew angle is less than half the angle of view, the third and final geometry must be considered. When the field of view extends both in front of and behind the camera, the field of view must be broken into three component triangles, the third triangle being the field of view of the single pixel that captures the area directly under the camera/lens assembly.

The next step in the process is determining how much of the test surface each individual pixel captures. This is measured by calculating the vertical and horizontal test surface length represented by each pixel. Each pixel of the camera’s sensor has its own field of view that, like the field of view of the whole image, increases in size as the distance from the camera to the test surface increases. In order to simplify the calculation of each individual pixel length, the horizontal and vertical component lengths are calculated independent of each other so that one-dimensional right triangle analysis can be used. Identical to determining the overall dimensions of the image, the geometry used in calculating individual pixel length depends on the skew of the camera and the focal length of the lens.

The process of calculating the fringe spacing begins with the centroid locations of each fringe. Fringe spacing is calculated by isolating two adjacent fringes at a time. Fringe spacing is defined as the distance, in inches, between the centers of two adjacent fringes. The idea behind choosing fringe centroids as a reference point is that they lie along the geometric centerline of the fringe, assuming that the oil smear propagation is relatively symmetric. Under this assumption, the centroids of any two-fringe system lie on the same plane as the leading and trailing edges of the two fringes. In the MATLAB solver, the geometric center is defined as the midpoint between the leading and trailing points of each fringe. Using the slope of the line that connects the two centroids, the leading and trailing edge points can then be found by extending the line that connects the two centroids and observing where the projected line intersects the boundaries of the two fringes. Once the center of each fringe is determined, the sum of the pixel lengths, in the vertical and horizontal direction, are calculated. The total fringe spacing is determined by the use of Pythagorean's Theorem.

1.5 Experimental Outlook and MATLAB Solver Validation

After completion of the MATLAB solver, centerline accuracy and fringe spacing accuracy were evaluated independently of each other. Experimentation was conducted primarily with the use of hand-drawn representative oil smears. Five total image datasets are analyzed in proceeding sections of this thesis. Each dataset is differs from the others in that each depict unique oil smear propagation. One dataset contains images taken at orthogonal camera positions, two contain X-skew camera conditions ranging for 0 degrees to 12.7 degrees, while the remaining two datasets feature Y-skew camera orientations ranging from 0 degrees to 13.2 degrees. An example of a hand drawn oil smear is shown below in Figure 10. While there is a valid argument that the 'mock' oil smears should be created in a more precise manner so that the smear spacing can be measured to a higher degree of accuracy, hand drawing the fringes allows the user to test how the solver responds to imperfections in the fringes.

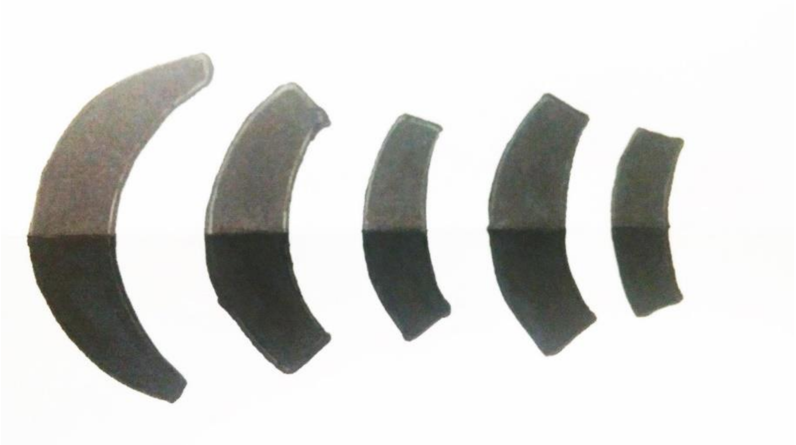


Figure 10. Five-fringe hand drawn representative oil smear, colored with black ink, left to right horizontal propagation.

Centerline accuracy is assessed by comparing hand-drawn centerlines to the centerlines created by the MATLAB solver. Initial analysis is performed by comparing the slopes and midpoint location of the hand-drawn, and MATLAB centerlines. Secondary analysis is conducted by relating centerline data to a human repeatability study conducted by Robert Schaefer Ehrmann[3] as part of his Cal Poly San Luis Obispo Master’s thesis. Fringe spacing determination is more straight forward than centerline analysis. The fringe spacing calculated by the MATLAB solver is compared to fringe spacing measured by hand with the use of a standard 1/16th inch accuracy.

2.0 MATLAB SOLVER

One of the main goals of this thesis is to develop the MATLAB code in such a way that would allow for the code to be easily expanded upon in the future. This is achieved by ensuring the modularity of the code throughout its ongoing development. Each function embedded within the script has defined inputs and outputs that will remain valid should the capabilities be expanded at a later date. The progression of the solver is illustrated below in Figure 11. Each portion of the MATLAB solver is explained in greater detail in the following sections

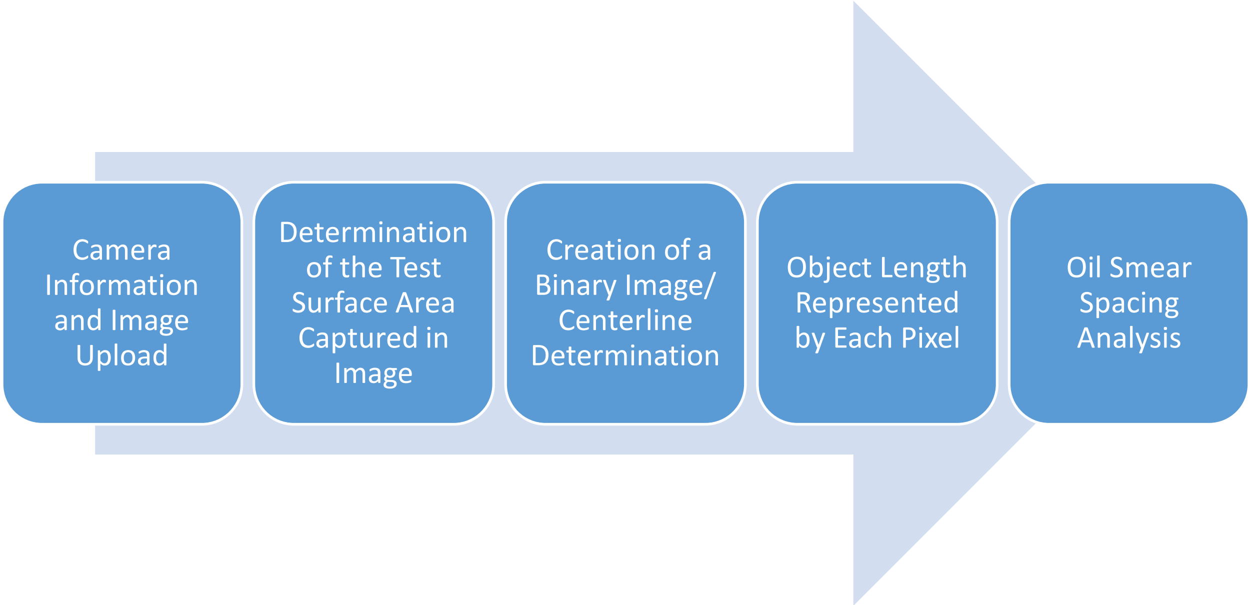

Camera Information and Image Upload
Determination of the Test Surface Area Captured in Image
Creation of a Binary Image/ Centerline Determination
Object Length Represented by Each Pixel
Oil Smear Spacing Analysis

Figure 11. Process flow of the MATLAB solver, the code is comprised of five main parts, beginning with a user uploading a photo and ending with fringe spacing determination.

2.1 Camera Information and Image Propagation

The first section of the MATLAB solver establishes the physical properties of the image being analyzed. In their raw form, images do not accurately represent true the dimensions of the objects captured in them because of the camera’s imperfect perspective, and the two-dimensional limitation of a camera’s digital sensor. Camera perspective can be described as how the image spatially represents the objects that are captured within it. Modern digital camera sensors translate 3D object space onto a 2D image plane. The image plane is confined by the dimensions of the camera’s sensor and the focal length of the lens attached to the camera. Digital sensors are rectangular in shape, resulting in vertical and horizontal fields of view.

Resulting images incorporate both fields of view, producing a rectangular image plane with the same aspect ratio as the digital sensor. The vertical and horizontal angles of view (α) are the angles at which the field of view emanates outward from the lens focal point and are given below by Equations (3) and (4). An image plane is assumed to have no area if the distance from the sensor to the object is zero. As the distance to the object increases, so too does the area of the image plane. This concept is illustrated in Figure 12 below. The dimensions of the image plane can be determined with the use of simple trigonometry, assuming that the distance from the camera's sensor to the object is known. In the two equations below, l_{vert} represents the vertical length of the digital sensor while l_{horz} represents the horizontal length of the sensor. f is the focal length of the camera lens.

$$AoV_{vert} = arctan\left(\frac{l_{vert}}{2f}\right) \tag{3}$$

$$AoV_{horz} = arctan\left(\frac{l_{horz}}{2f}\right) \tag{4}$$

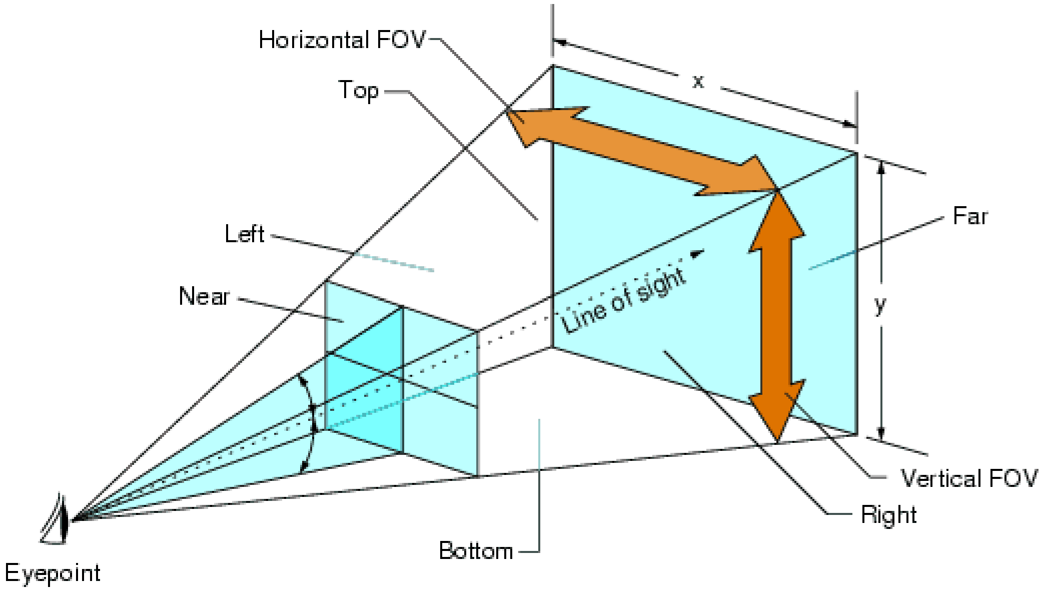


Figure 12. Propagation of an image plane where the 'eyepoint' represents the focal point of the lens; image area plane increases with plane distance.[1]

In order to begin the process of determining accurate length measurements, the physical dimensions of the test surface captured in the image must be determined. As explained in the introduction section of this text, the geometry between the field of view and flat test surface changes depending on the degree of camera skew and focal length. The next three sections of this text are devoted to explaining how different combinations of camera skew and lens focal length affect the overall dimensions of the test surface captured in an image. The following sections address length determination in one dimension. Because the analysis is limited to flat surfaces, pixel surface length in the second dimension can be determined using identical equations.

2.1.1 Orthogonal Overhead Camera Location

When an image is taken from directly above an object, the rectangular image plane has no perspective skew regardless of lens focal length, meaning that object proportions are accurate throughout the image. A lack of perspective skew also means that the image represents equal object distance on both image halves. This camera orientation is illustrated below in Figure 13, which depicts a simplified two-dimensional case of the resultant geometry. If Figure 13 were a three-dimensional sketch, there would be an identical triangle propagating into and out of the paper. For this thesis, camera height (Z) is a known value, measured by hand for each image. With the use of isosceles triangle geometry, the object distance captured by each half of the image is determined with the application of Equation (5), given below; X_1 and X_2 are equal for this camera geometry. As will be seen in the next section, once camera skew becomes non-zero, the surface dimensions captured by the two halves of the image will not be of equal length.

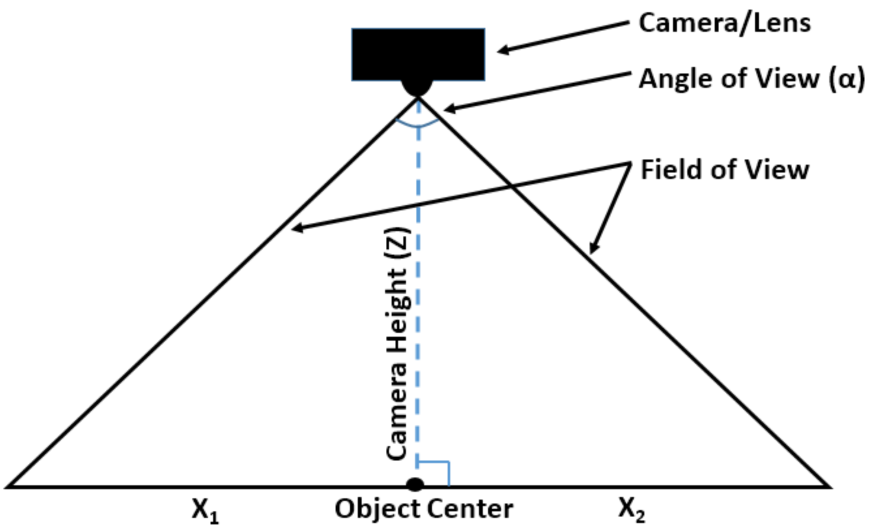


Figure 13. 2-D image geometry with overhead camera location, both image halves capture identical object lengths due to orthogonal symmetry.

$$X_n = tan\left(\frac{\alpha}{2}\right)Z \tag{5}$$

2.1.2 Camera Pivot Angle Greater than Half the Angle of View

When camera skew is non-zero, there are two geometries that can occur. This section addresses the geometry that occurs when the camera pivot angle is less than half the angle of view. As shown in Figure 14, under this camera set up, propagation ends before capturing the area immediately below or behind the camera. Also apparent from Figure 14, non-zero camera pivot angle (β) results in X_1 and X_2 being unequal in length, with X_2 representing a greater object length than X_1.

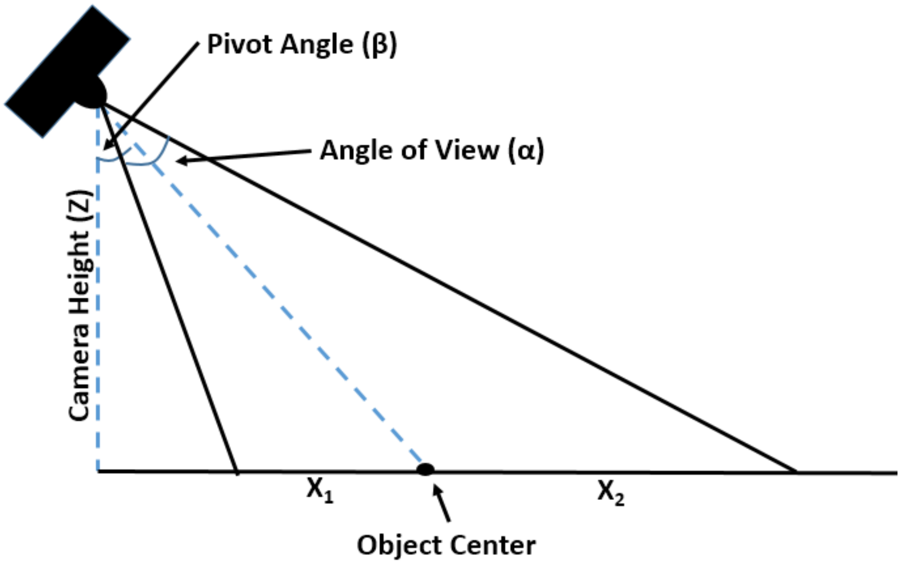


Figure 14. Image geometry for camera pivot angle less than half the angle of view, image half closest to camera captures less object length than the image half furthest from camera.

In order to determine X_1 and X_2, the geometry shown in Figure 14 is separated into component triangles so that simple right-triangle relationships can be taken advantage of. The first of the two component triangles is shown in Figure 15. X_1 is a product of L_1 and L_2, and is determined using Equations (6-8).

$$L_1 = tan\left(\beta - \frac{\propto}{2}\right)Z \tag{6}$$

$$L_2 = tan(\beta)Z \tag{7}$$

$$X_1 = L_2 - L_1 \tag{8}$$

In order to obtain X_2, a second component right triangle is used that incorporates the field of view furthest from the camera. This is shown below in Figure 16. X_2 is a function of L_3, L_1, and X_1; it is calculated with the use of Equations (9 & 10).

$$L_3 = tan\left(\beta + \frac{\propto}{2}\right)Z \tag{9}$$

$$X_2 = L_3 - (L_1 + X_1) \tag{10}$$

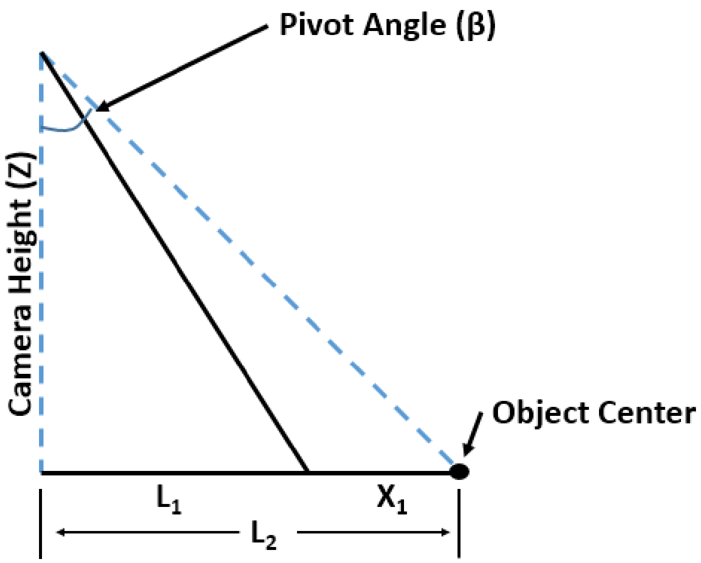


Figure 15. Component triangle one, camera pivot angle greater than half the angle of view, incorporates the Z plane and line of sight to the object center.

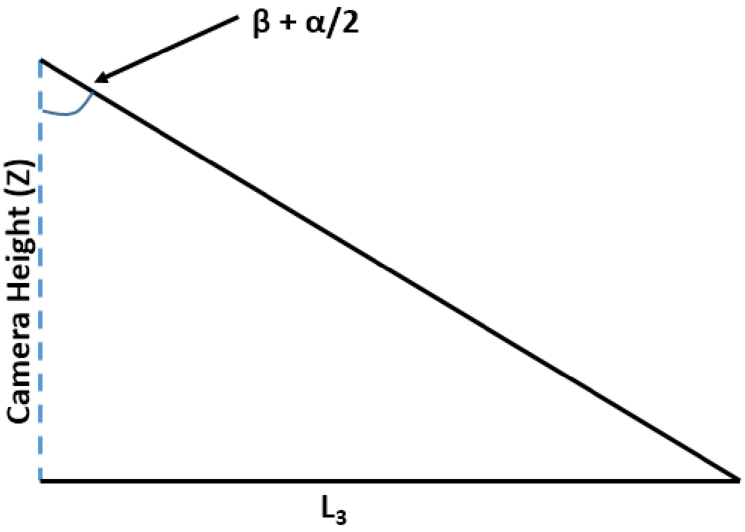


Figure 16. Component triangle two, camera pivot angle greater than half the angle of view, boundaries are the Z plane and right array of the field of view.

2.1.3 Camera Pivot Angle Less than Half the Angle of View

When the camera's pivot angle is less than half of the angle of view, a third geometry must be considered. This is necessary because the field of view encompasses the area directly below, and 'behind' the camera. X_1 and X_2 are again unequal in length in this configuration, as the perspective skew orientates the field of view such that the background of the photograph captures more of the test surface's area. The geometry for camera pivot angle less than half the angle of view is given below in Figure 17.

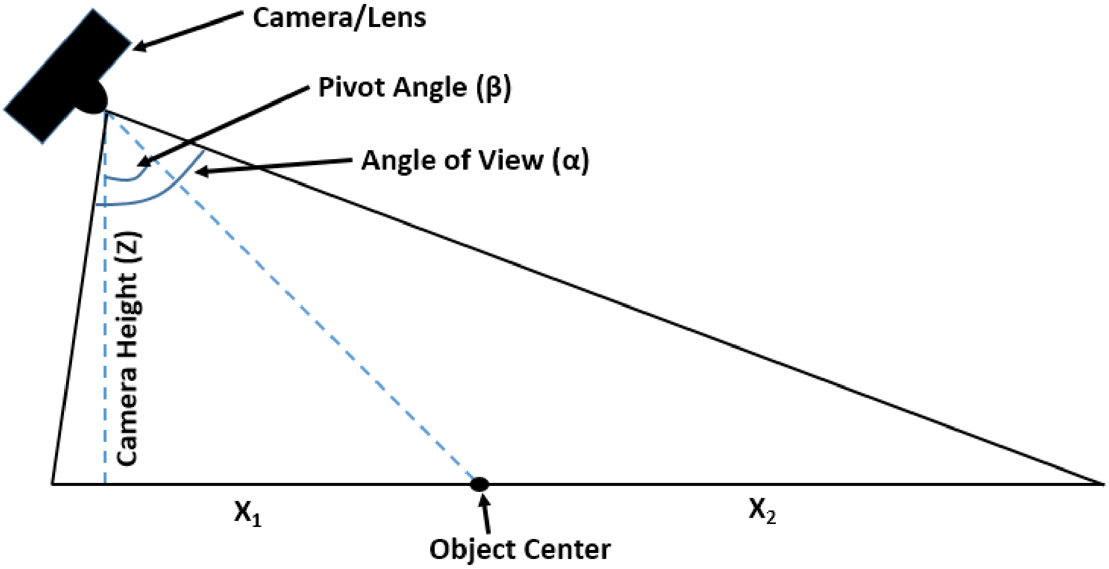


Figure 17. Image geometry for camera pivot angles less than half the AoV, field of view extends to either side of the Z plane.

Like Figure 14, Figure 17 can be broken down into component triangles in order to obtain X_1 and X_2. X_1 is determined from the first component triangle, shown in Figure 18. Equations (11-13) are used to determine its length.

$$L_1 = tan\left(\frac{\propto}{2} - \beta\right) Z \tag{11}$$

$$L_2 = tan(\beta)Z \tag{12}$$

$$X_1 = L_2 + L_1 \tag{13}$$

The component triangle used in determining X_2 is identical to Figure 16. X_2 is calculated using Equations (14 and 15).

$$L_3 = tan\left(\beta + \frac{\propto}{2}\right) Z \tag{14}$$

$$X_2 = L_3 - (X_1 - L_1) \tag{15}$$

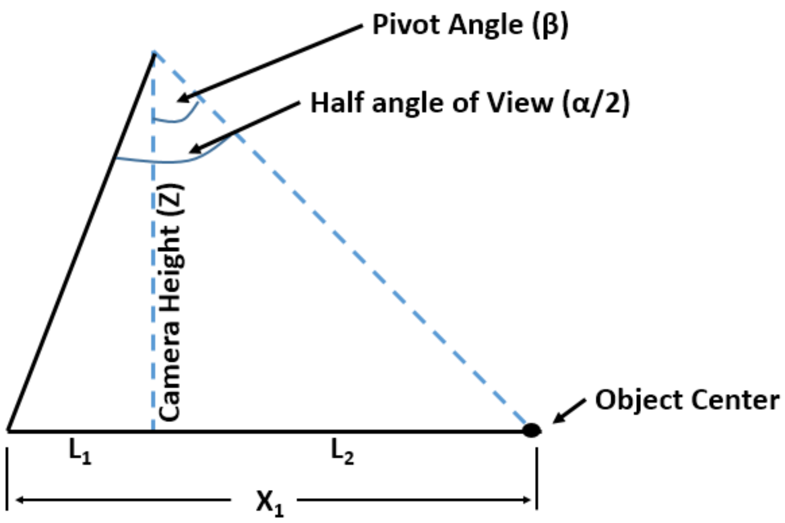


Figure 18. Component triangle one, camera pivot angle less than half the angle of view, boundaries are the left array of the field of view and the line of sight to the object center.

2.2 Initial Image Processing: Creating and Refining a Binary Image

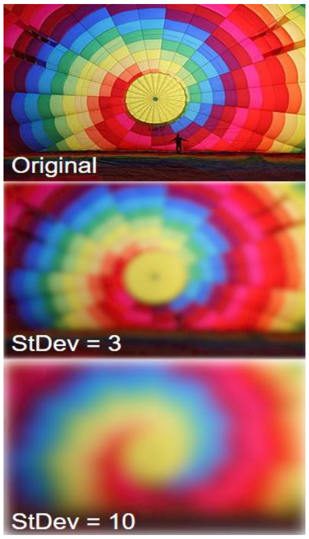


Figure 19. Effect that Gaussian Blur has on an image, as more adjacent pixels are taken into account for the blur of a pixel of interest, the more blurred the image becomes. *[2]*

Initial image processing takes advantage of the powerful image processing toolbox that is included within the MATLAB software suite. Once the image has been read into MATLAB, the image quality is sharpened with the use of an unsharp mask filter[15]. This process utilizes blurred negative images to create a 'mask' overlay of the image. The mask is then combined with the original image to create a resulting image filter that is less blurry than the original. The mask is created with the use of a Gaussian blur, which reduces image noise and detail. Figure 19 gives an example of how Gaussian blur affects a photograph. The standard deviation referred to in Figure 19 is the standard deviation of the Gaussian distribution. Because the images used in this thesis are high quality, sigma values of two are used to generate the blurred mask. This means that when each pixel is evaluated by the Gaussian function, only the pixels that are within 2σ distance from the pixel of interest will be evaluated when determining the transformation of the pixel of interest. The Gaussian function used is given below in Equation (16). This equation is two dimensional, where x is the distance from the origin on the horizontal axis, and y is the distance from the origin in the vertical axis. Luckily, Gaussian blur is a predefined option in the FSPECIAL 2-D filter MATLAB function, so implementing a filter of this type is simple[].

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (16)$$

The next step is to apply the Gaussian-created filter to the native image. This is accomplished with the use of the MATLAB function IMFILTER[12], which takes in the original and filtered image to construct a completely new image using a double-precision floating point process.

In order to detect the edges of the oil fringes, MATLAB requires a binary image. The binary image is accomplished by passing the sharpened image through the MATLAB function RGB2GRAY[6]. This function transforms each individual pixel from a complex red-blue-green identifier into a binary (one or zero) identifier. Pixels are defined as a one if they are encompassed within a continuous enclosed space, while pixels outside of any defined, closed edge are designated a zero identifier. In the case of this thesis, this means that the edges of each oil fringe, as well has everything interior of the fringe edges are defined as a one in the binary image; while the pixels in areas of the 'white space' outside the fringes are designated as zeros.

Edge detection is accomplished with the use of a Sobel Operator[15]. For this application, the Sobel algorithm detects a continuous edge and defines the pixels that are a part of that edge in a matrix of pixel coordinates. Once all edges are detected, the binary image is redefined and the pixels interior to each continuous edge are given the binary designation of zero. The resulting binary image contains only an outline of the fringe sections, with all other portions of the image appearing as white negative space.

Initial results with the Sobel method were less than ideal, as the oil fringes often do not form perfect, continuous edges. To remedy this, the binary image can be dilated and filled with the use of the

MATLAB functions IMDILATE[8] and IMFILL[10]. The function IMDILATE expands the edge boundaries to adjacent pixels, aiding in the removal of discontinuous lines. Furthermore, IMFILL is effective at filling in the edge gaps by expanding the edge boundaries beyond what IMDILATE is capable of.

Unfortunately, the use of IMDILATE had an undesirable side effect in that stray oil blotches not part of the oil fringes became noticeably more prominent in the image. These were problematic, as later functions of the MATLAB solver require that only the FISF smears be visible in the binary image. This issue was addressed in two ways. First, the MATLAB function IMERODE[9] was incorporated into a WHILE loop that ran until either there were only three-edged bodies left or the loop had been executed ten times. The function works by slowly removing pixels from object edges in the image. This was effective to a point, but it was discovered that the continuous edges of the oil smears would be broken before the errant marks completely dissipated from the image. As a final fix, after the WHILE loop was executed, a catch was implemented in the analysis portion of the solver that would ignore any edges that had a length of less than 10 pixels.

2.3 Fringe Centroid and Centerline Determination

In order to begin the process of centerline determination, the solver requires a reference point that remains unchanged regardless of how the oil smear propagates across an image. During development of the MATLAB solver, using the centroid of each fringe as a reference point was determined to be most effective. Assuming relatively symmetric fringes, fringe centroids are positioned along the fringe's centerline, near its geometric center. This means that each centroid lies on the same plane as the leading and trailing edges of their respective fringe. When the centroids of two adjacent fringes are looked at as one system, the linear line connecting the centroids can be used to determine the location of the leading and trailing edge points. The trailing edge point of the first fringe and the leading edge point of the second fringe are determined by observing where the linear line connecting the two centroids intersects the edges of each fringe. In order to define the pixel coordinates of the leading edge of the first fringe, and the trailing edge of the second fringe, the linear line connecting the two centroids is extended. The point at which the extended line intersects the leading edge of the first fringe is the leading edge centerline point of that fringe, and the point at which the extended line intersects the trailing edge of the second fringe is the trailing edge centerline point of the second fringe.

Fortunately for this thesis, MATLAB features a built-in function that determines the centroid location of any shape that has a closed boundary. The matrices containing the pixel coordinates of each fringe edge are passed into this function and the centroid locations, in terms of their X and Y pixel location, are returned to the user.

As alluded to in the previous paragraphs, fringe distance is determined two fringes at a time. Isolating two fringes at a time increases the accuracy of the solver because it allows for more accurate modeling of the leading and trailing edge points. Despite the oil smear propagating in the presence of omnidirectional airflow, there is little chance that every fringe centroid will lie on a perfectly linear plane. This means that a line connecting all centroids would most likely be an n^{th} order equation. While MATLAB has the capability to fit high order equations to data sets, there is a possibility for the n^{th} order equations to have erratic behavior in between the centroids. This poses a risk, as errant line-fit behavior between centroids does not yield accurate leading and trailing edge point locations.

In every photograph, fringe analysis begins with the two left most fringes in the image. After determining the leading and trailing edge points for both fringes, the solver shifts one fringe to the right and the leading/trailing edge determination begins again. Figure 16 below shows a two fringe system with superimposed leading and trailing edge points. The left portion of the image depicts the centerline of the two fringes once it has been extended to the leading edge of the first fringe and to the trailing edge of the second fringe. Once the linear line has been extended to all four edges of the of the fringe system, the leading and trailing edges of both fringes can be determined with a MATLAB intersection program[14].

The intersection function determines leading and trailing edge points one fringe at a time. The function has two inputs: the boundary coordinates of the linear line spanning the length of the two-fringe system, and the matrix of edge coordinates for one of the two fringes. The curve inputs are two-rowed matrices, with the X indices propagating on the first row, and the Y indices on the second row. The function returns two intersection points; the leading edge centerline point, and the trailing edge centerline point, in the form of a two-by-two matrix. The MATLAB solver populates the first row of the matrix with the X-coordinates of the leading and trailing edge points, while the Y-coordinates are populated in the second row of the matrix. The process is then repeated for the second fringe in the system, as well as for all other two-fringe systems in the image.

The fringe center points, not shown in Figure 20, lie midway between the leading and trailing edge point of each fringe. This figure prompts the question that the solver may be better off determining fringe spacing directly from the centroids, and that determining the leading and trailing edge points is an unnecessary venture. As centroid location is dependent on the shape of the fringes, their location can be unpredictable, especially at higher degrees of camera skew. As shown in Figure 20, the centroid location is aft on both fringes. This does not pose an issue for this specific example because both of the centroid locations are aft on the oil fringe. With the unpredictable fringe quality that is inherent in wind tunnel testing, finding fringe spacing using the center point of each fringe is a more predictable method of obtaining accurate results.

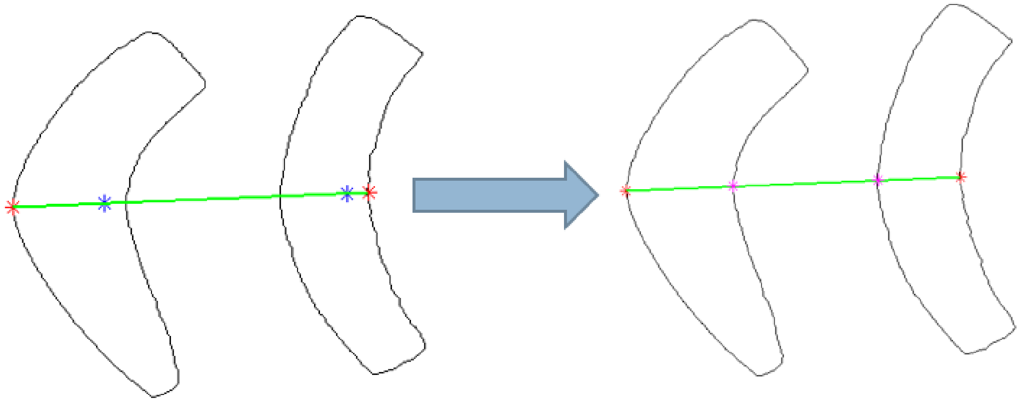


Figure 20. Leading and trailing edge point location determination, edge point locations determined one two-fringe system at a time.

Much like Zilliac's CXWIN5G, the centerline determination portion of the MATLAB solver uses more than two data points to construct the centerline of fringe propagation. The main difference between the two codes is that CXWIN5G uses up to six data points to approximate a second order curve, while the MATLAB solver creates a centerline for each two-fringe system. Because this thesis features analysis of oil smears propagating on a smooth, flat surface under omnidirectional flow condition; comparison of hand-drawn and MATLAB centerlines is accomplished under the assumption of linear centerlines. This decision was made for two reasons: because it simplifies analysis to only two points per centerline, and because second order curve fitting is rarely used by CXWIN5G users anyway.

Up to this point in the MATLAB solver, the primary focus of the code is achieving the first goal of this thesis; which is to obtain accurate fringe centerline approximations while achieving a high level of automation. Centerline analysis is presented in **Section 5**. The proceeding sections of this text address the second goal of this thesis, as they lay the foundation for fringe spacing determination.

2.4 Object Length Captured by Each Pixel

In order to measure the distance between the oil fringes, the area of the test surface captured by each pixel must be determined. The surface area is determined by calculating the length of the test surface represented in each pixel along the image's vertical and horizontal planes. Unfortunately, this is more complex than dividing the camera's resolution by the image length found in **Section 2.1**. Each individual pixel can be thought of as having its own field of view. If the object of interest is infinitely close to a camera's sensor, then the pixel's field of view is infinitely small. As the object plane moves outward, the field of view for each individual pixel expands proportionally with it. Like the propagation of the image's entire field of view, the geometry behind determining each pixel length is dependent on the camera skew angle and focal length of the camera lens.

The MATLAB solver is capable of analyzing fringe spacing regardless of the direction the oil smear propagates across an image. When an oil smear propagates across an image in a non-horizontal or non-vertical manner, fringe distance is a product of horizontal and vertical distance components. The subsequent sections focus on the calculation of pixel lengths along the horizontal (X) axis. Pixel length along the vertical axis are calculated in much the same manner, the only difference being that the vertical pixel's field of view stem from the vertical component of the rectangular camera sensor.

2.4.1 Pixel Length: Orthogonal Overhead Camera Location

First, pixel length for a camera directly over an object of interest will be addressed. The camera geometry is given below in Figure 21. This figure depicts a camera setup with a horizontal resolution of 6-pixels. While images nominally have upwards of 1000 horizontal pixels, depicting all of them in this overview would only serve to add unnecessary complexity. Because this camera orientation affords a mirrored pixel distribution across the image centerline, only one half of the image is shown with pixel field-of-view lines. One assumption afforded to the analysis is that the first pixel's field of view begins directly at the image's center. In all likelihood, this is an accurate assumption, but there is really no way to determine the true pixel accuracy of a camera's sensor without placing the sensor under a microscope and quantifying its pixel irregularity. In any case, even if the field of view of the first pixel is not directly at the image center, the large number of pixels in an image would result in error on the order of one-thousandths of a degree.

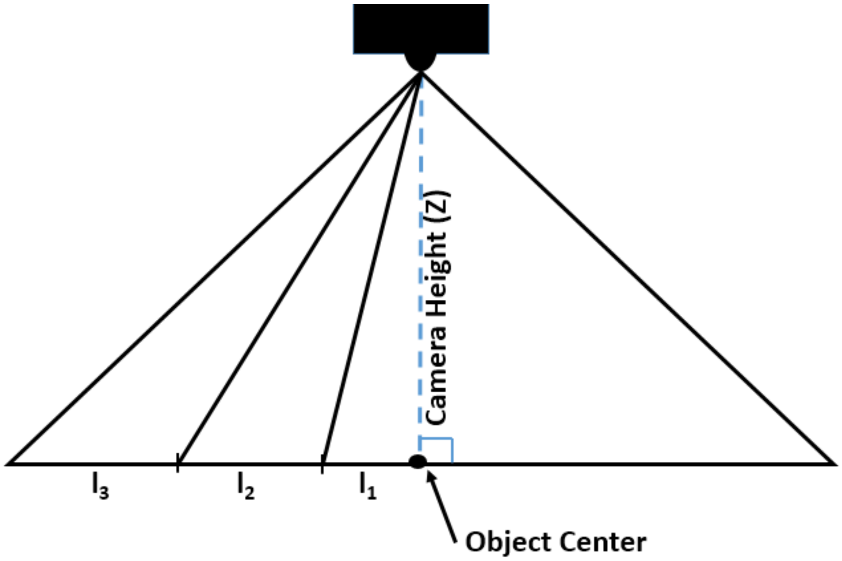


Figure 21. Determination of pixel length for an orthogonal overhead camera location, simplified three-pixel example.

The orthogonal case is the simplest of the three geometries. The MATLAB solver makes use of the right triangle that is created between the image center line (Z-plane) and the left edge of each pixel's field of view. The specific geometry used to determine the length of the first pixel, l_1, is given below in Figure 22; where *n* is the number of pixels in the horizontal direction on one half of the image, and H_1 is the hypotenuse of the triangle. Determining l_1 is accomplished with the use of Equation (17).

$$l_1 = tan\left(\frac{\alpha}{2n}\right)Z \tag{17}$$

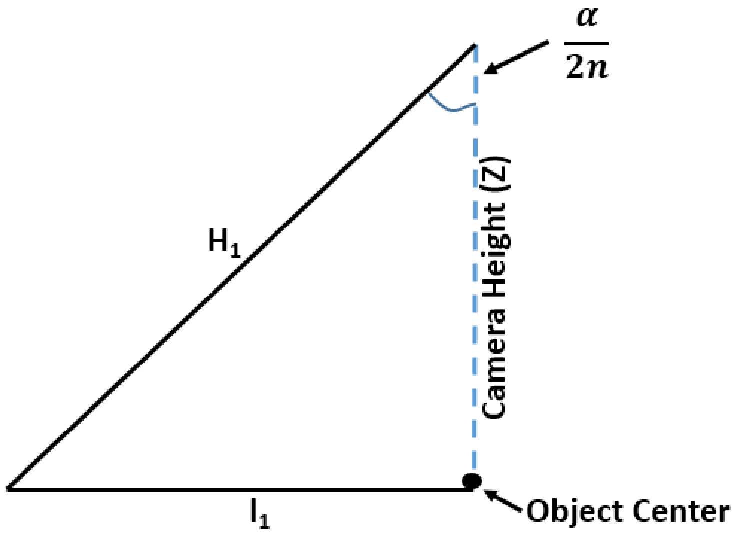


Figure 22. Individual pixel length, orthogonal overhead camera location, depicts determination of pixel length adjacent to the Z plane (center of image).

For subsequent pixel lengths, the method of determination is a bit more complex. After the first pixel, the object length from a pixel's outer field-of-view boundary to the center of the image is the sum of the current pixel of interest's length and all previous pixel lengths. Therefore, the sum of the previous pixel lengths must be subtracted from the total length to obtain only the length of the current pixel of interest. The angle between the outboard array of the pixel's field of view and the image centerline increases proportionally with pixel propagation. For the third pixel from the image center, the angle between the Z-Plane and hypotenuse would be 3 times the angle shown in Figure 19 and the angle of the fourth pixel is four times that of the first pixel. Determining pixel length for this geometry is accomplished with the use of Equations (18) and (19). The MATLAB solver calculates each pixel length individually via a FOR loop that iterates through vector *i* that is of length *n*.

$$l_{1\rightarrow i} = tan\left(\frac{\alpha}{2n}(i)\right)Z \tag{18}$$

$$l_i = l_{1\rightarrow i} - l_{1\rightarrow (i-1)} \tag{19}$$

2.4.2 Pixel Length: Camera Pivot Angle Greater than Half the Angle of View

As was described in **Section 2.1**, when the camera's pivot angle is non-orthogonal, the geometry of image propagation changes. The main difference is that the orientation of each pixel's field of view becomes a function of the camera pivot angle in addition to the angle of view (α). Unlike the orthogonal case, the two halves of the image have dissimilar pixel propagation. This occurs because the two halves of the image do not capture the same object length, but do have an identical number of pixels. Because of this, each half of the image is examined separately by the MATLAB solver, as the geometry equations that define the pixel lengths in one half do not accurately define the lengths in the other half.

Image geometry when the camera pivot angle is greater than half the angle of view is addressed first. The geometry of the image half closest to the camera is given in Figure 23 below. Right triangles are again used in the calculation of the pixel lengths in order to keep analysis as simple as possible. Pixel propagation begins at the center of the image and propagates outward, ending with the *n*th pixel at the boundary of the camera's field of view. As with all of the other geometry analysis, the equations given in this section of the text remain valid regardless of whether the camera is positioned to the left or to the right of the oil smear.

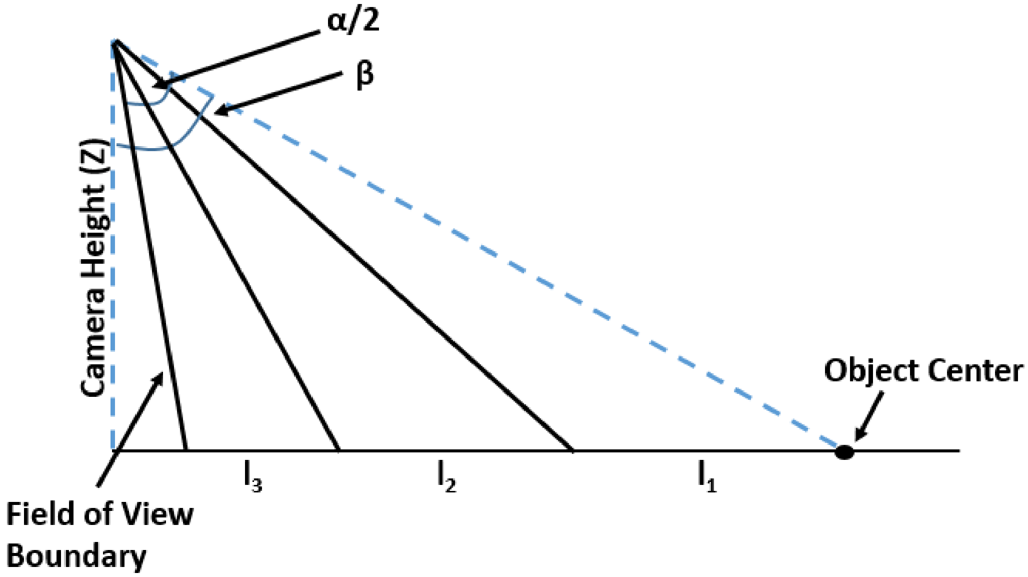


Figure 23. Determination of pixel length for camera pivot angle greater than half the angle of view, simplified three-pixel example, image half closest to camera.

The geometry specific to determining the first pixel length is given in Figure 24 below. The horizontal distance between the object center and the camera (l_{tot}) is determined with the use of Equation (20). The horizontal distance between the outer array of each pixel and the camera (l_{camera}) is calculated by Equation (21). As is shown in Equation (22), the first pixel length is determined by subtracting l_{camera} from l_{tot}. For subsequent pixels, the sum previous pixel lengths is also subtracted from l_{tot}.

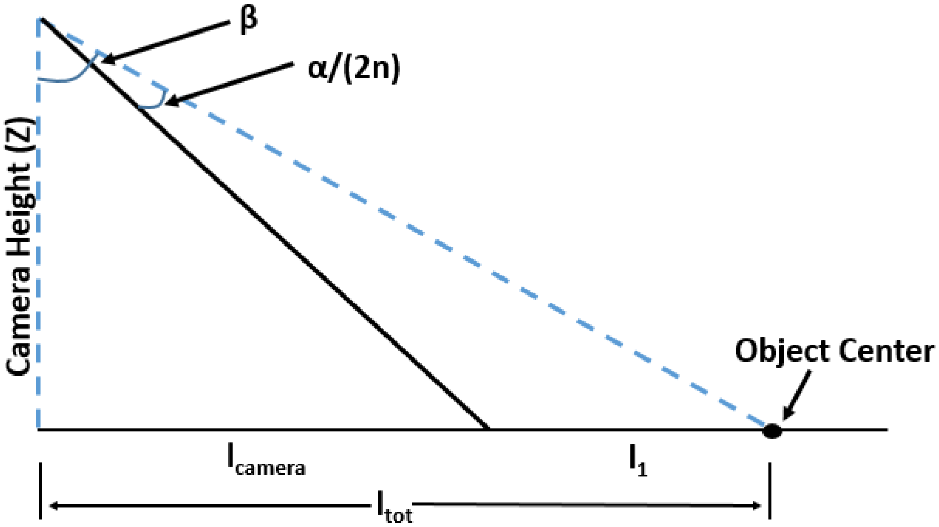


Figure 24. Individual pixel length, camera pivot angle greater than half the angle of view, pixel length adjacent to the center of the line of sight plane, image half closest to camera.

$$l_{tot} = \tan(\beta)Z \qquad (20)$$

$$l_{camera(i)} = \tan\left(\beta - \frac{\alpha}{2n}(i)\right)Z \qquad (21)$$

$$l_i = l_{tot} - l_{1\rightarrow(i-1)} - l_{camera(i)} \qquad (22)$$

The pixel lengths on image half furthest from the camera are calculated in much the same fashion as they are for the image half closest to the camera. It is important to remember that even though a greater object distance is captured in this portion of the image, the number of pixels are identical to the other half of the image. This results in individual pixel lengths that are greater than those on the opposite half of the image. A three-pixel geometry of the propagation is given below in Figure 25.

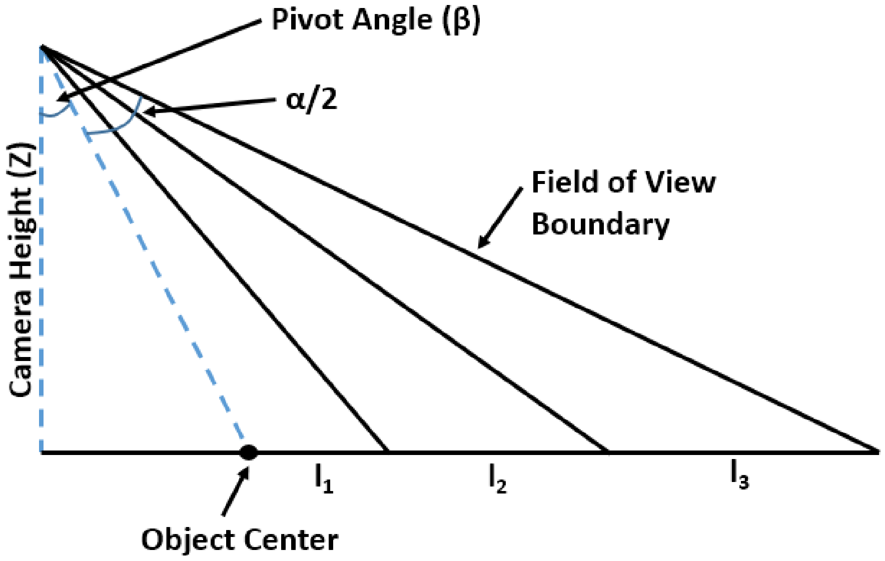


Figure 25. Pixel length, Camera Pivot Angle Greater than Half the Angle of View (image half furthest from camera)

As in previous cases, pixel length in this portion of the geometry is determined with the use of right triangles. For this geometry, the triangles are defined by camera height (Z) plane, and the outermost boundary of each pixel's field of view array. As in the other half of the image, the MATLAB solver begins calculating pixel length immediately adjacent to the image center, and moves outward, solving pixel length until the image's field of view boundary is reached. The geometry used to determine these pixel lengths is shown in Figure 26. l_{tot} is defined as the horizontal distance between the object center and the camera. The value l_{camera} extends from the camera location to the outer array of each individual pixel. As the solver propagates from the object center towards the outer edge of the image, the length of l_{cam} increases with every new solver iteration. Equations (23-25) are used to determine each of the *n* pixel lengths.

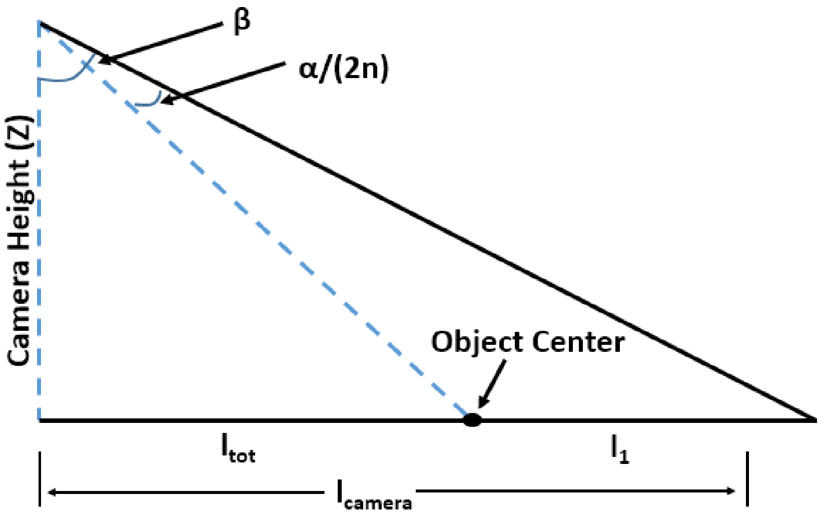


Figure 26. Individual pixel length, camera pivot angle greater than half the angle of view, pixel length adjacent to the center of the line of sight plane, image half furthest from camera.

$$l_{tot} = \tan(\beta)Z \tag{23}$$

$$l_{cam(i)} = \tan\left(\beta + \frac{\alpha}{2n}(i)\right)Z \tag{24}$$

$$l_i = l_{camera(i)} - (l_{1\rightarrow(i-1)} + l_{tot}) \tag{25}$$

2.4.3 Pixel Length: Camera Pivot Angle less than Half the Angle of View

When the camera-lens combination is oriented such that the pivot angle is less than half the angle of view, a third and more complex geometry must be considered. For this camera orientation, the image encompasses the area that is directly below and behind the camera. For geometries of this type, pixel propagation is unique in that it begins at the outer edge of the image and ends at the center of the image. Determining pixel length for this camera orientation is more complex than the other geometries due to the transition area under the camera. In order to preserve the right triangle pixel determination scheme, the left half of the image was separated into three component triangles: one extending from the left field of view boundary to the Z-plane, another from the Z-plane to the image center, and the third encompassing the Z-plane cross-over pixel. Figure 27 depicts pixel propagation over the entire left half of the image. The individual component triangles will be discussed in the coming paragraphs.

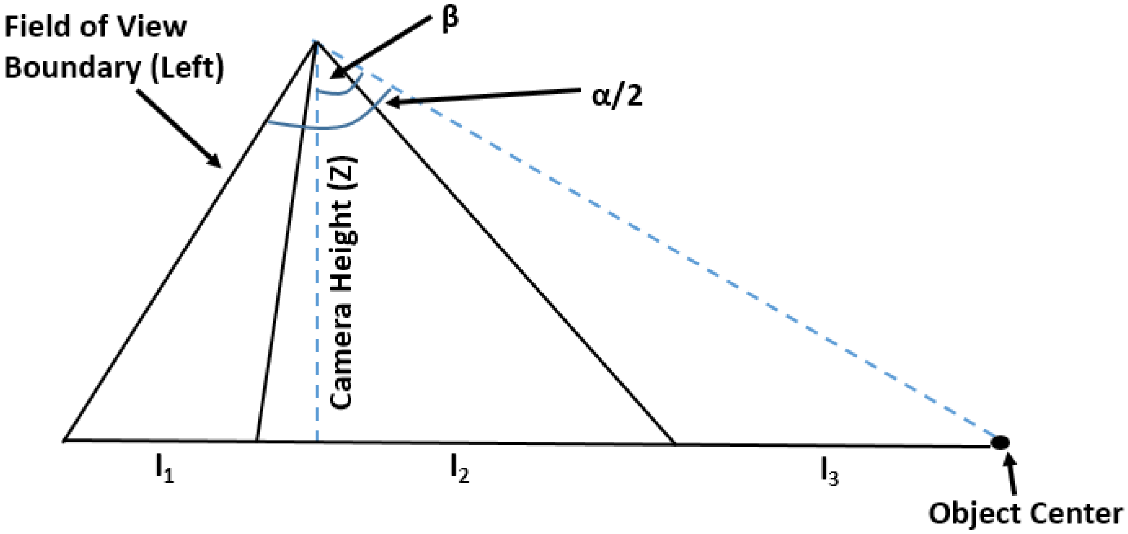


Figure 27. Simplified three pixel length geometry, camera pivot angle less than half the angle of view, l_1 located behind camera, l_2 is the cross-over pixel.

For pixels to the left of the cross over pixel, determining their pixel length is similar to the other geometries already discussed. The geometry used for pixels in this area is shown in Figure 28. The MATLAB solver utilizes an IF statement with the condition outlined in Equation (26). As long as this condition remains valid, the solver determines that the pixel propagation has not yet reached the cross-over pixel, and Equations (27-29) are used for the determination of pixel length. The main difference between this geometry and the others is that l_{camera} decreases in length as the solver iteration gets closer to *n*. For every pixel, l_{camera} is the length from the pixel's left array to the camera height plane, Z. The length of l_{camera} and the sum of all previous pixel lengths are subtracted from l_{tot} in order to obtain each individual pixel length.

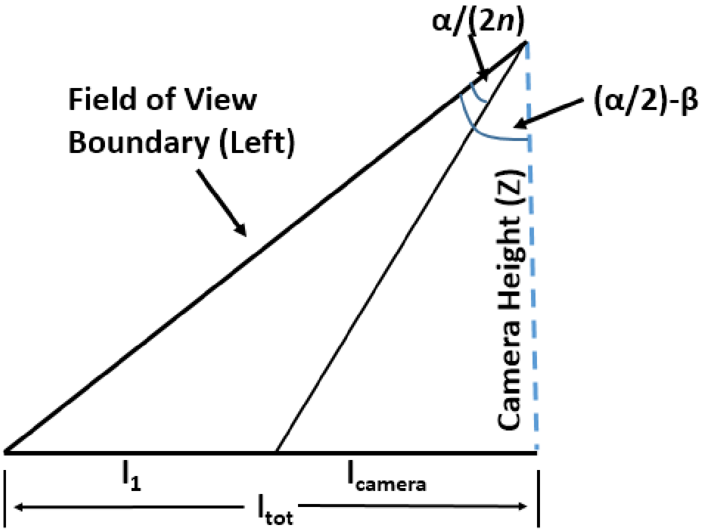


Figure 28. Individual pixel length, pivot angle less than half the angle of view, portion of image geometry behind the camera, before cross-over pixel

$$\left(\frac{\alpha}{2}-\beta\right)-\left(\frac{\alpha}{2n}i\right)>0 \tag{26}$$

$$l_{tot}=\tan\left(\frac{\alpha}{2}-\beta\right)Z \tag{27}$$

$$l_{camera(i)}=\tan\left(\frac{\alpha}{2}-\left(\beta+\frac{\alpha}{2n}i\right)\right)Z \tag{28}$$

$$l_i=l_{tot}-(l_{1\rightarrow(i-1)}+l_{camera}) \tag{29}$$

When the solver reaches the cross-over pixel, it is necessary to split the pixel into its left and right halves so to maintain right triangle geometry analysis. The MATLAB solver determines the cross over pixel with the use of a two condition IF statement: Equation (30) and Equation (31). Equation (30) is true for the cross-over pixel, as well as every pixel after it. Since the cross-over pixel only partially extends over the Z-plane, Equation (31) is used to limit the conditions of the IF statement to only the crossover pixel as the angle between the Z-plane and the right array of the crossover pixel is less than the angle of a full pixel array. The geometry for the crossover pixel is shown in Figure 29. The pixel length of the crossover pixel is determined with the use of Equations (32-36). The entire length of the crossover pixel is denoted as l_{cross}, while the individual components of l_{cross} are l_{left} and l_{right}.

$$\left(\frac{\alpha}{2}-\beta\right)-\left(\frac{\alpha}{2n}i\right)<0 \tag{30}$$

$$abs\left(\left(\frac{\alpha}{2}-\beta\right)-\left(\frac{\alpha}{2n}i\right)\right)<\left(\frac{\alpha}{2n}\right) \tag{31}$$

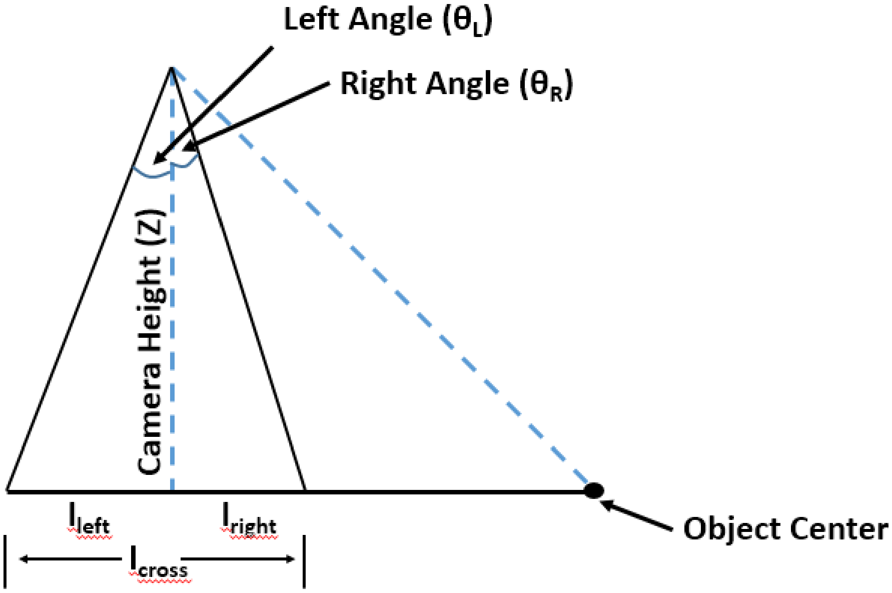


Figure 29. Cross-over pixel geometry, a portion of the cross-over pixel is on either side of the Z plane.

$$\theta_R = abs\left(\frac{\alpha}{2}-\left(\beta-\frac{\alpha}{2n}i\right)\right) \tag{32}$$

$$\theta_L = abs\left(\frac{\alpha}{2n}-\theta_R\right) \tag{33}$$

$$l_{right} = tan(\theta_R)Z \tag{34}$$

$$l_{left} = tan(\theta_L)Z \tag{35}$$

$$l_{cross} = l_{right} + l_{left} \tag{36}$$

To the right of the transition pixel, the pixel geometry returns to a configuration similar to that of the narrow field of view image addressed in **Section 2.3.2**; the main difference being that in this geometry pixel propagation begins at θ_R instead of β. The geometry is given below in Figure 30. Equations (37-39) are similar to the equations given in **Section 2.3.2** as well, the only difference being that l_{tot} is a function of θ_R. Since the MATLAB solver iterates through all of the pixels on the image half and the pixel array begins before the Z-plane, l_{camera} is a function of the camera pivot angle as well as the field of view. Furthermore, the pixel sum that is subtracted from l_{camera} is reset for this geometry so that the pixel summation from the other side of cross-over pixel is not factored into the calculation of pixel length moving forward.

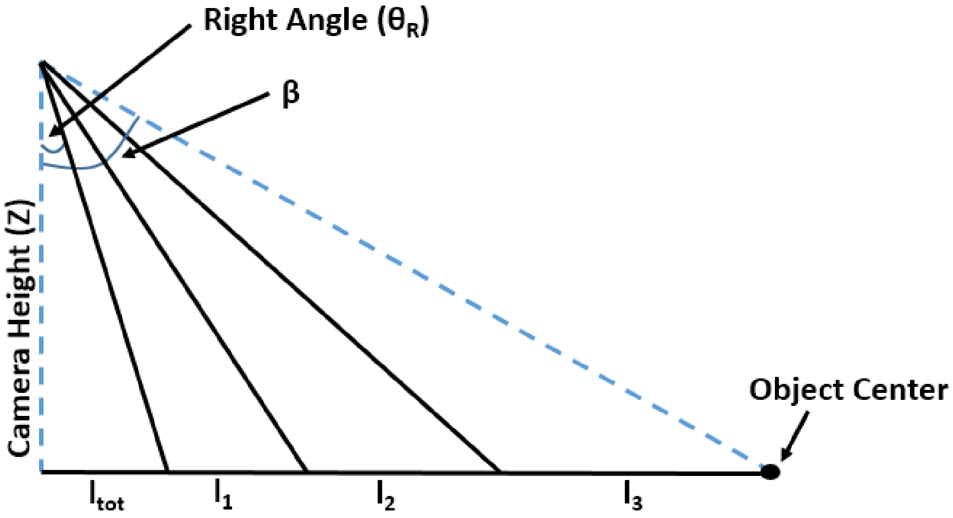


Figure 30. Three pixel length geometry after the cross-over pixel, side closest to the camera, camera pivot angle less than half the angle of view.

$$l_{tot} = \tan(\theta_R)Z \tag{37}$$

$$l_{cam(i)} = \tan\left(\frac{\alpha}{2n}(i) - \left(\frac{\alpha}{2} - \beta\right)\right)Z \tag{38}$$

$$l_i = l_{cam(i)} - (l_{(CO+1)\to(i-1)} + l_{tot}) \tag{39}$$

Calculation of pixel length for the image half furthest from the camera is identical to the geometry shown in Figures 25 and 26. In addition to the sharing the same geometry, Equations (23-25) can be reused in the calculation of pixel length of this geometry. The figures and equations related to the calculation of pixel lengths for this geometry can be found in **Section 2.3.2** of this text.

2.5 Oil Smear Spacing Analysis

Once the object lengths of the pixels have been determined, the MATLAB solver works to determine the distance between each of the oil smears. This is accomplished by calculating the fringe spacing along each fringe's centerline. The first step in this process is to identify the pixels that capture the front and rear edge of each fringe along its centerline. Once that has been accomplished, the solver then determines the absolute length from each center line leading and trailing edge point, to the center of the image. Finally, the distance between each oil fringe is calculated using the absolute lengths. This process is accomplished by calculating the horizontal and vertical components of the leading and trailing fringe edges separately. Once all of their component information is derived, total fringe spacing is derived from combining the two component lengths via Pythagorean's Theorem.

2.5.1 Oil Smear Analysis: Leading/Trailing Edge Points to Center of Image

The first step in determining fringe spacing is calculating the distance, in inches, between each leading and trailing edge point, and from the center of the image. This step in the analysis process is the reason why the pixel length vectors were chosen to propagate outward from the center of the image, as it means that the distance from any edge point to the image center is a sum of the pixel lengths from the image center to the pixel location of the edge. As with all other analysis completed in this thesis, the X and Y coordinates are dealt with separately. Determining the leading and trailing edge distance along the horizontal (X) plane is discussed first.

While the origin of the pixel length vectors is at the center of the image, the origin of the coordinate system that defines each leading and trailing edge point is at the lower left hand corner of the image. To reconcile the two coordinate systems, the edges are classified into two groups: those that are on the left half of the image, and those that are on the right. On the left side of the image, pixel location of the midpoint edges is defined by Equation (40), while the pixel location of the midpoint edges on the right half of the image are defined by Equation (41). *Edge* is the two-by-two matrix containing the coordinates of the leading and trailing edge points for each fringe. The geometry behind pixel location is given in Figure 31.

$$Pixel_X = n - Edge(i, 1) \tag{40}$$

$$Pixel_X = Edge(i, 1) - n \tag{41}$$

Pixel location is defined as the number of pixels in the horizontal direction between the image center and a midpoint edge. On the left side of the image, the horizontal pixel coordinate of the midpoint edges is less than the number of total number of pixels on the left side of the image. Thus, using Equation (40) results in the number of pixels that are in between the image's centerline, and the midpoint edge of interest. On the right side of the image, Equation (41) is used. This equation also gives the number of pixels between the image's centerline and leading/trailing edge, as the entire image is 2*n* pixels in length. If the pixel distance was not measured from the vertical centerline, Equation (41) would be invalid, as each side of the image would not be represented by the same number of pixels.

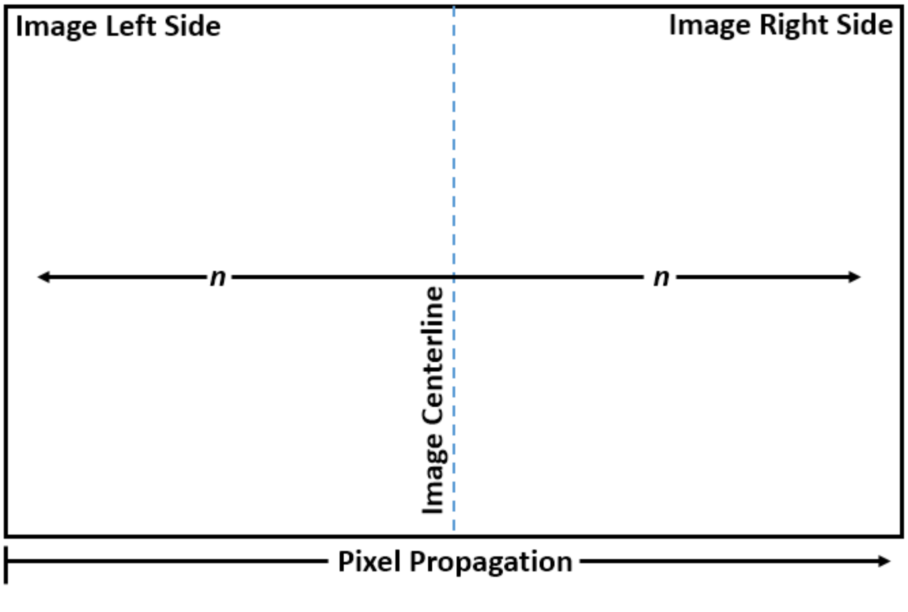


Figure 31. Pixel length vectors propagate outward from the center of the image, coordinate origin for individual pixels located at the lower left corner of the image.

Once pixel location has been determined using either Equation (40), or Equation (41), actual distance between the leading/trailing centerline edges and the center of the image, P_{dist}, can be determined. In the MATLAB Solver, this is accomplished with the use of Equation (42), given below.

$$P_{dist} = \left(\sum_{1}^{Pixel_X - 1} Pixels\right) + \frac{Pixels(Pixel_X)}{2} \qquad (42)$$

As a reminder, *Pixels* is vector of length *n* that contains the object length captured by each pixel. As covered in previous sections, the calculation of the *Pixels* vector is dependent on camera orientation and lens focal length as well as the side of the image that is being observed. This portion of the MATLAB script is split into three sections, each of which is uniquely adapted to addressing one of the three possible pixel propagation geometries. Once the *Pixels* vectors are determined for the left and right side of the image, they are fed into Equation (42).

Like the leading and trailing edge point detection, distances from the leading and trailing edge points to the image center is calculated one two-fringe system at a time. For each two fringe system, there are three possibilities concerning the distribution of the edge points within the image: all of the edges are contained within the left half of the image, all edges are on the right half of the image, or the edges are distributed across both image halves. Because the distance from the fringe edge to centerline is dependent on which image half the edges lie on, each of the three possibilities have to be accounted for by the MATLAB solver. The solver determines which of the three possibilities each two-fringe system adheres to by looking at the X-coordinate of the leading and trailing edge points. If all four of the X-coordinates are less than the number of pixels on one half of the image (*n),* than all four of the edge points lie on the left half of the image. If all of the X-coordinates are greater than *n*, all four edge points lie on the right half of the image. Finally, if some of the edge points are less than *n,* while others are greater than *n,* then there are edge points distributed on both halves of the image.

The first of the three cases occurs when edges lie on either side of the image. The initial step in this process is for the solver to determine how many of the four points lie within the boundaries of the left half of the image. Once determined, the solver enters into a FOR loop that is the same length as the number of edge points on the left half of the image. For every iteration, the solver applies Equation (40) to determine the distance to the *Pixels* vector representing the left half of the image, and begins populating a one-row, four column vector with said distances. Once the FOR loop for the left half of the image has run its course, the solver enters into a second FOR loop the length of the number of edge points on the right half of the image. The second FOR loop completes the four-entry distance vector by populating distance values with the use of Equation (41), and the *Pixels* vector calculated from the right half of the image.

When all four of the leading/trailing edge points lie on the left half of the image the distances are calculated using Equation (40), and the left *Pixels* vector. If the edge points are all on the right half of the image, distances are calculated using Equation (41), and the *Pixels* vector calculated from the right half of the image. All three distance determination cases are executed from within an IF/ELSEIF statement. This method allows for a single distance vector output regardless of which of the three cases is used. The distance vector created for each two-fringe system are placed into a multidimensional array where they can be called upon for use in subsequent sections of the MATLAB solver.

As in many other portions of this solver, incorporating the vertical dimension into analysis is quite similar to horizontal analysis. The main differences being that *n* represents the number of pixels in the vertical space, and that the solver is now concerned with whether the leading or trailing edge points lie on the top, or bottom half of the solver. For each two-fringe system, the four Y-components of the leading/trailing edge points can either be entirely on the upper half of the image, entirely on the lower half, or distributed between the two. Their location is determined by comparing the magnitude their pixel location to *n*, where *n* is the number of pixels in the vertical direction in half of the image. Like horizontal analysis, a one-row, four column distance vector is produced for each two-fringe system, and is saved within a multidimensional array that can be called upon for use in later functions.

2.5.2 Oil Smear Analysis: Determining Center Point of Each Fringe

Once the distances from the leading and trailing edges to the image's center have been calculated, the center point of each fringe can be determined. This is done by averaging the pixel location of the leading and trailing edge locations of each fringe in both the vertical, and horizontal direction. Like the two previous steps, the center of each fringe is found in an iterative process, with analysis being done one two-fringe system at a time. For the majority of fringe geometries, determining the fringe center points is as simple as taking the mean average of the X and Y components of the leading and trailing edge points of each fringe. The calculation becomes a bit more complicated if the leading and trailing edges of a single fringe are on opposite halves on an image. The pixel-length vectors propagate outward from the center of the image, and are non-negative regardless the side of the image they represent. This means that all distance to center measurements are positive values. When calculating the X-component of the mean average, if one of the edge points is on the left half of the image, and the other is on the right, the distance from the point on the right half of the image is turned negative. This way, when the average is taken, the true distance is represented. If the resulting average is positive, then the fringe center is on the left side of the image, while a negative result means that the fringe center resides on the image's right half. The Y-component has a similar determination scheme, with the distances on the upper half designated as negative when a fringe spans both the top and bottom halves of an image. For each two-fringe system, the X and Y pixel coordinates of both fringe center points are assembled into a two-by-two matrix, with the X values populating the first column, and the Y values populating the second column. Each matrix is placed into a multidimensional array for use in the final section of the MATLAB solver.

2.5.3 Oil Smear Analysis: Determination of Fringe Spacing

Like the fringe centers before it, fringe spacing is determined in its component X and Y parts. Spacing is determined for each two-fringe system; where the first system defines the spacing between the first and second fringe, the second system defines spacing between the second and third fringes, et cetera. X-component, and Y-component spacing between the two fringes are determined by comparing the pixel location of the fringe centers of a two-fringe system. Determining fringe spacing is dependent on where each of the fringe centers lie within the image, and there are four geometries that must be accounted for. This portion of the solver features four IF/ELSEIF/ELSE statements; one for each possible spacing geometry. The following paragraphs address each geometry, as well as define the equations that are used in each geometry scenario.

The first geometry occurs when the center point of the first fringe is on the opposite side of the image as the center point of the second fringe, with the center point of the second fringe defined as a negative distance. This can occur in both the X, and Y components of the fringe centers. The MATLAB solver determines when this scenario occurs with the utilization of an IF statement that is triggered when either the X oy Y component of the second fringe's center point distance is negative. When the IF statement condition is met, the fringe distance along the X axis is defined by Equation (42), while the distance along the Y axis is determined with the use of Equation (43). In both equations F_{center} represents the array of fringe center points for a particular two-fringe system.

$$F_{distX} = F_{center}(1,1) + abs(F_{center}(2,1)) \tag{42}$$

$$F_{distY} = F_{center}(1,2) + abs(F_{center}(2,2)) \tag{43}$$

The second geometry occurs when two fringe centers are on opposite sides of an image, and neither of the two fringes have negative components. An ELSEIF statement is utilized that searches for instances where the mean average of the leading and trailing edge points for the first fringe are on the opposite side of the image as the mean average of the edge points from the second fringe. This is done by comparing the mean averages to *n*, where *n* is the number of pixels in half of the image. When the conditions of this IF statement are met, distance in the X-direction is determined with the use of Equation (44), and distance in the Y-direction is determined using Equation (45).

$$F_{distX} = F_{center}(1,1) + (F_{center}(2,1)) \tag{44}$$

$$F_{distY} = F_{center}(1,2) + (F_{center}(2,2)) \tag{45}$$

The third spacing geometry occurs when the first fringe in a two-fringe system has a negative center point distance component, while the center point of the second fringe does not have any negative components. An ELSEIF statement triggers this case when the either center point component of the first fringe is negative, while both center point components of the second fringe are positive. Fringe distance in the X-direction is determined with the use of Equation (46), while the Y-direction distance is determined using Equation (47).

$$F_{distX} = F_{center}(2,1) + F_{center}(1,1) \tag{46}$$

$$F_{distY} = F_{center}(2,2) + F_{center}(1,2) \tag{47}$$

If the conditions of the previous three IF/ELSEIF statements are not met, then the solver defaults to a fourth, and final fringe geometry with the use of an ELSE statement. This geometry occurs when both fringes in the two-fringe system are on the same half of the image, and none of their index components are negative. In this situation, the X-distance component is found using Equation (48), and the Y-component is found with the use of Equation (49).

$$F_{distX} = abs\big(F_{center}(1,1) - F_{center}(2,1)\big) \tag{48}$$

$$F_{distY} = abs\big(F_{center}(1,2) - F_{center}(2,2)\big) \tag{49}$$

Since true fringe spacing has both X and Y components, the component lengths must be combined. This is done with the use of Pythagoreans Theorem. Actual fringe distance is determined for every two-fringe system with the use of Equation (50).

$$F_{dist} = \sqrt{{F_{distX}}^2 + {F_{distY}}^2} \tag{50}$$

3.0 QUANTIFYING RELATIONSHIP BETWEEN CAMERA SETUP AND FRINGE ACCURACY

In order to analyze the effect that camera setup has on the accuracy of fringe analysis, partial derivatives are taken of all equations relevant to the calculation of the pixel lengths in **Section 2.4**. The partial derivatives are taken with respect to the following variables: camera height (*Z*), lens focal length (*f*), and camera pivot angle (β). Taking the partial derivatives with respect to these three input variables allows for complete understanding of how each component of the camera setup affects the calculation of pixel length (*l)*. Partial derivatives are taken of the pixel length equations in only one-dimension, as the vertical and horizontal components of the pixel length are calculated using identical equations.

3.1 Angle of View (AoV)

The angle of view is the angle at which the camera's field of view propagates outward and is discussed at length in **Section 2.1** of this text. The equations that define the vertical and horizontal angles of view are given by Equations (3) and (4). Angle of View is a function of the lens focal length and the length of the camera's rectangular sensor *(l_{sensor})* in one dimension. Since the length of the camera sensor is a constant value, only the partial derivative with respect to the focal length is calculated. The partial derivative is given below in Equation (51). The partial derivative shows that AoV is inversely proportional to focal length, meaning that an increase in focal length results in a narrower field of view. It also shows that the angle of view is more susceptible to change at shorter focal points; as focal length grows, the field of view becomes narrower and, thus, the angle of view does not change as fast when going between higher focal lengths as it does when going between shorter focal lengths. This trend is validated in Figure 32 below. Notice that the vertical axis is negative, with slope trending towards zero as the data reaches the upper limit of the axis.

$$AoV = tan^{-1}\left(\frac{l_{sensor}}{2f}\right) \tag{3}$$

$$\frac{\partial AoV}{\partial f} = -\frac{l_{sensor}}{2f^2\left(\frac{l_{sensor}^2}{4f^2}+1\right)} \tag{51}$$

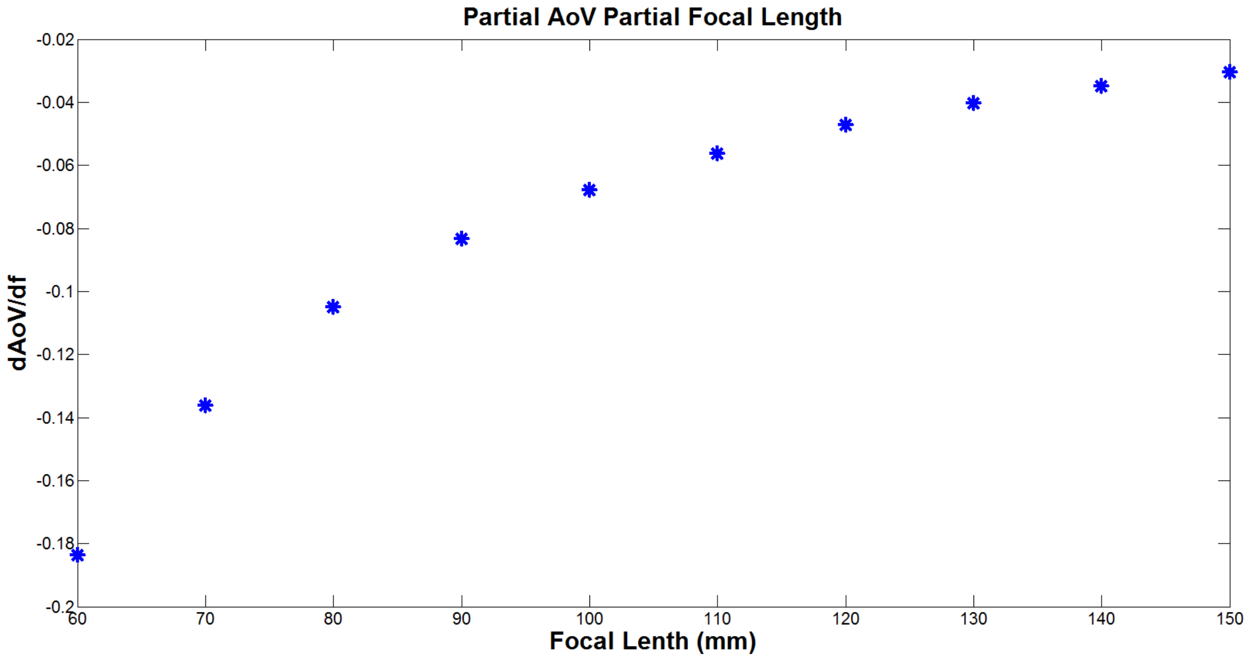


Figure 32. Sensitivity in the rate of change in angle of view with respect to focal length, rate of change trends towards zero as focal length increases.

3.2 Pixel Length: Orthogonal Overhead Camera Location

Determining pixel length for an orthogonal overhead camera location is accomplished using Equation (17) from **Section 2.4.1** of this text. Determining the pixel length for this geometry takes into account both camera height (Z), and angle of view (α). Since angle of view is a function of focal length in addition to camera height, partial derivatives with respect to both camera height and focal length are discussed in this section.

$$l = tan\left(\frac{\alpha}{2n}\right)Z \tag{17}$$

Taking the partial derivative with respect to camera height, we are left with Equation (52). As the equation is not a function of *Z*, it is apparent that the rate at which pixel length changes with respect to camera height is a constant. This means that there is a linear relationship between pixel length and camera height, with pixel length increasing as camera height increases.

$$\frac{\partial l}{\partial Z} = tan\left(\frac{\alpha}{2n}\right) \tag{52}$$

Taking the partial derivative of Equation (17) with respect to the focal length yields Equation (53). Like the angle of view, there is a negative correlation between a change in focal length and a change in pixel length. As focal length increases, pixel length decreases. This occurs because a longer focal length results in a narrower field of view, while the number of pixels remains the same. In other words, the rate at which the pixel length decreases is slowed as focal length increases; if the focal length changes from 40mm to 50mm, pixel length will decrease much more than if focal length is changed from 140 mm to 150 mm. Like the angle of view, this occurs because an image’s field of view trends towards zero as focal length increases, meaning there comes a point at which an increase in focal length minimally decreases pixel length. Figure 33 gives visual confirmation of this trend by showing the rate of change of pixel length with respect to focal length. As focal length increases, the rate at which pixel length decreases as a function of focal length trends towards zero.

$$\frac{\partial l}{\partial f} = -\frac{Zl_{sensor}\left(tan\left(\frac{tan^{-1}\left(\frac{1}{2f}\right)}{2n}\right)^{2}+1\right)}{4f^{2}n\left(\frac{l^{2}}{4f^{2}}+l_{sensor}\right)} \tag{53}$$

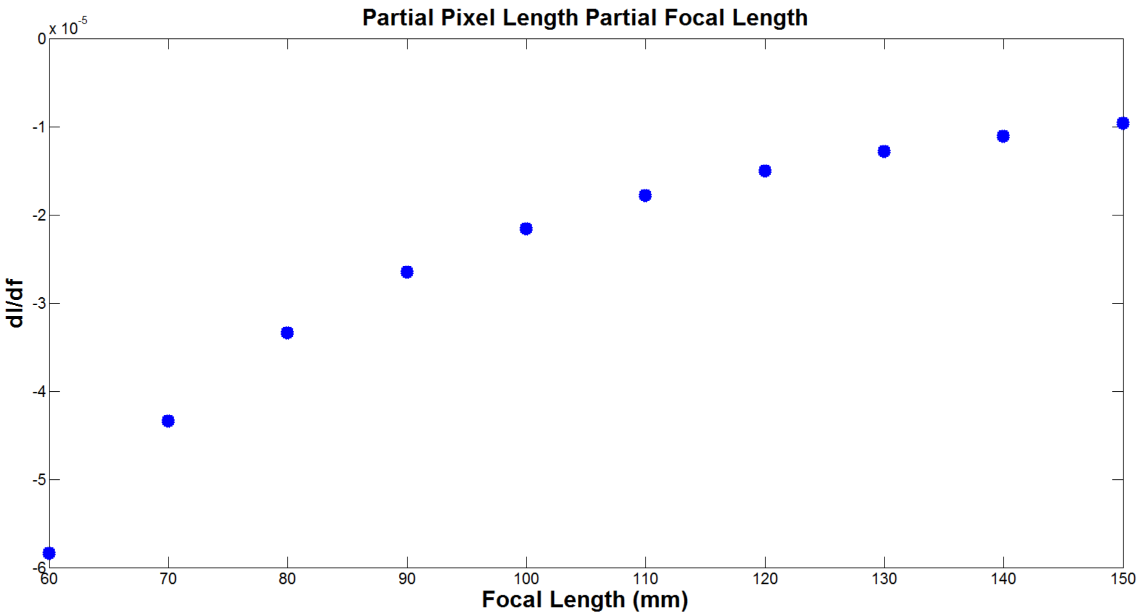


Figure 33. Sensitivity in the rate of change of pixel length with respect to focal length, rate of change trends towards zero as focal length increases (orthogonal camera location).

3.3 Pixel Length: Pivot Angle Greater than Half the Angle of View (Closest to Camera)

When the camera pivot angle is less than half the angle of view, Equations (20-22) of **Section 2.4.2** are used. Plugging Equations (20) and (21) into Equation (22) yields the differentiable equation covered in this section. This equation is a function of all three input variables, so three partial differentiations will be analyzed in the following paragraphs.

$$l = \tan(\beta)Z - l_{sum} - \tan\left(\beta - \frac{\alpha}{2n}\right)Z \tag{22}$$

When the partial derivative of Equation (22) is taken with respect to camera height, Equation (54) is the result. The same trend found in the previous section holds true for this equation. The absence of a Z variable signifies that the rate of change between pixel length and camera height is a linear one. The greater the camera height, the greater the pixel length.

$$\frac{\partial l}{\partial Z} = \tan(\beta) - tan\left(\beta - \frac{\alpha}{2n}\right) \tag{54}$$

The partial derivative of pixel length with respect to the focal length (*f*) is shown in Equation (55). Like the two previous geometries, the rate at which pixel length changes declines as focal length increases. Shown below in Figure 34, the partial derivative of pixel length with respect to focal length follow the same trend as the previous two sections. As focal length increases, the rate of change in pixel length with respect to focal length nears zero. A focal length range of 60 mm to 150 mm is chosen for analysis because the majority of the photographs analyzed in this thesis are captured at focal lengths within this range.

$$\frac{\partial l}{\partial f} = -\frac{Zl_{sensor}\left(\tan\left(\beta - \frac{\tan^{-1}\left(\frac{l_{sensor}}{2f}\right)}{2n}\right)^2 + 1\right)}{4f^2 n\left(\frac{l_{sensor}^2}{4f^2} + 1\right)} \qquad (55)$$

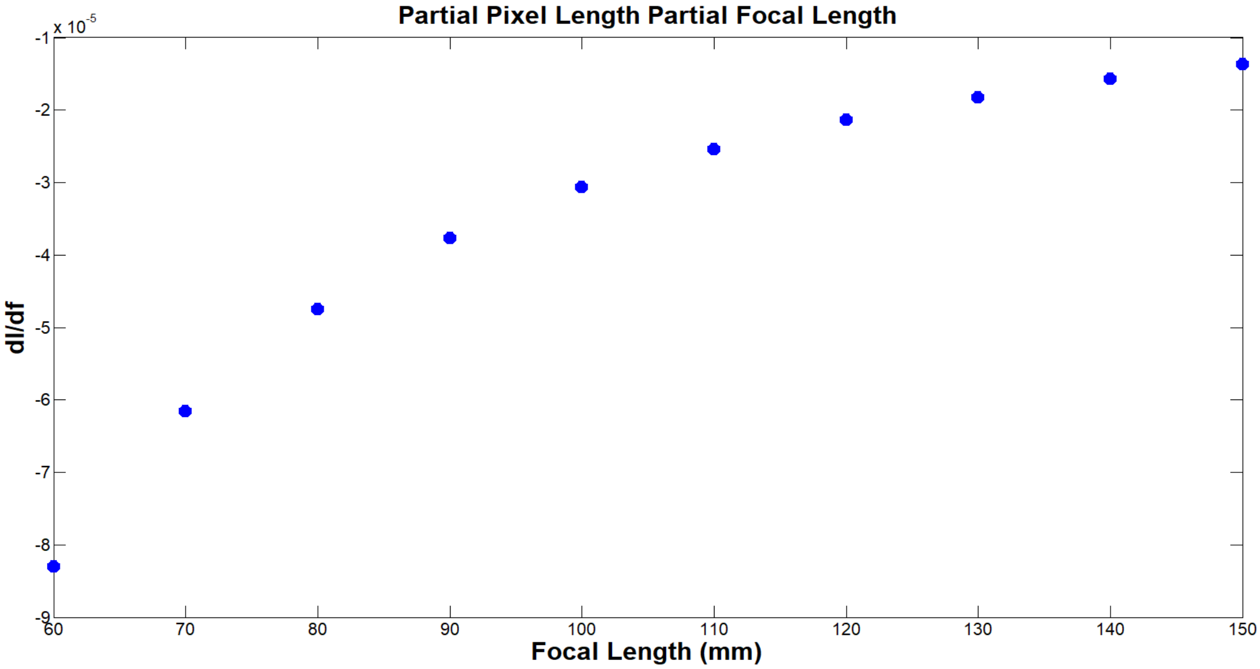


Figure 34. Sensitivity in the rate of change of pixel length with respect to focal length, rate of change trends towards zero as focal length increases (pivot angle greater than half angle of view, image half closest to the camera).

Finally, the partial derivative of pixel length with respect to skew angle (β) is given in Equation (56). The equation shows a positive correlation between the change of skew angle and the change of pixel length, meaning that when the camera skew angle increases, pixel length increases as well. There is a larger change in pixel length when skew angle changes from 30 to 35 degrees than when is when skew angle changes from 10 to 15 degrees. As skew angle increases, the camera's orientation becomes closer to the infinity plane, meaning that as camera skew angles increase, that increase will directly affect pixel length. If camera skew angle were to, for example, reach 90 degrees, the resulting image would be perpendicular to the leading edge of the flat plate. This would result in pixel distances of the flat side of the plate being effectively infinite, as the plate is traveling perfectly 'into' and 'out of' the image; so it makes sense that the rate of change in pixel length would approach infinity as the skew angle increases.

$$\frac{\partial l}{\partial \beta} = Z(tan(\beta)^2 + 1) - Z\left(tan\left(\beta - \frac{\alpha}{2n}\right)^2 + 1\right) \tag{56}$$

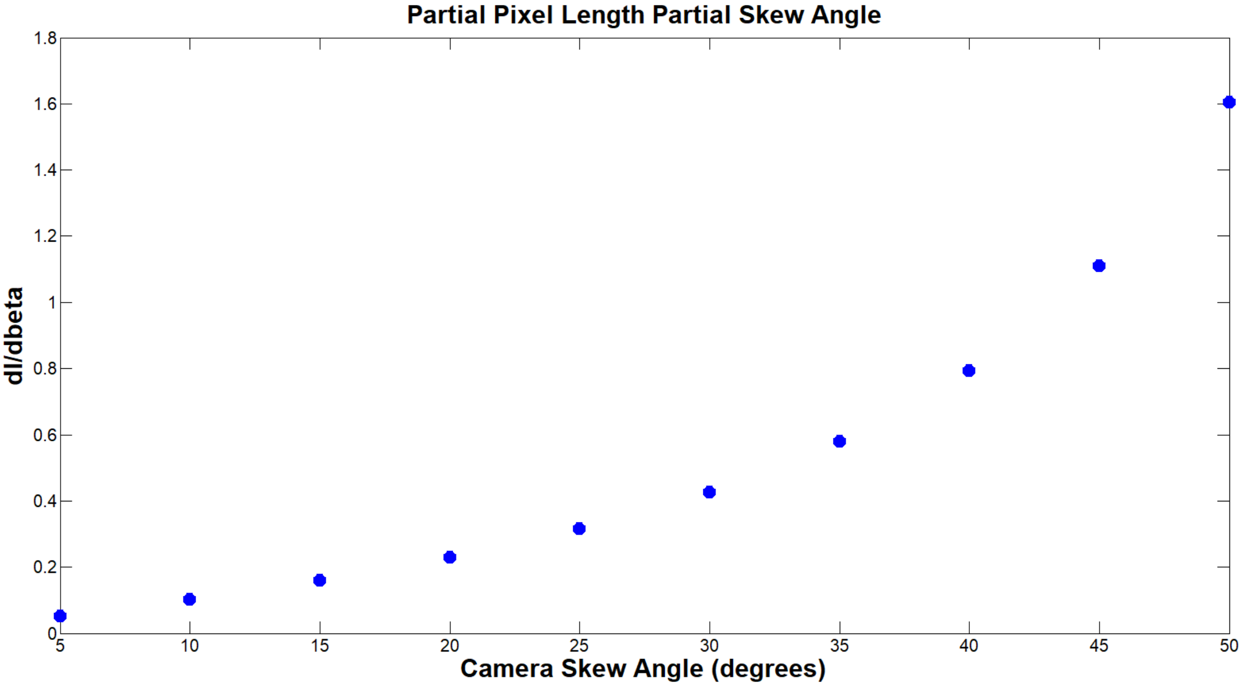


Figure 35. Sensitivity in the rate of change of pixel length with respect to camera skew, rate of change trends infinite as skew angle increases (pivot angle greater than half angle of view, image half closest to the camera).

The data trend shown in Figure 35 supports this analysis. At low degrees of camera skew, the rate of change of pixel length as a function of camera skew is less than it is at higher degrees of skew. Between 45 and 50 degrees of skew, there a near exponential increase in the rate of change of pixel length. If the horizontal axis of the data plot were extended, this trend would continue. The plot does not include data out to 90 degrees because none of the images used in this thesis are taken from skew angles approaching 90 degrees.

3.4 Pixel Length: Pivot Angle Greater than Half the Angle of View (Furthest from Camera)

The equation that will be differentiated in this section is a result of substituting Equations (23) and (24) into Equation (25), which is given below. The main difference between this equation and the equation describing pixel length for the opposite half of the image is that the l_{camera} component of Equation (25) increases in length rather than decreases as pixel length propagates outward from the image center. This minor change should not change the general trend of the partial differentiations from the differentiations completed in the previous section of this text.

$$l = \tan\left(\beta + \frac{\alpha}{2n}\right) Z - (l_{sum} + \tan(\beta) Z) \qquad (25)$$

The partial derivative of Equation (25) taken with respect to camera height (Z) is given below in Equation (60). As for all other pixel length equations differentiated with respect to camera height, Equation (57) is not a function of camera height. This means that the rate at which the pixel length changes with respect to camera height is linear.

$$\frac{\partial l}{\partial Z} = \tan\left(\beta + \frac{\alpha}{2n}\right) - \tan(\beta) \qquad (57)$$

When the pixel length equation is differentiated with respect to focal length, Equation (58) is the result. Like all the previous pixel length geometries, there is a negative rate of change in pixel length as focal length increases. As focal length increases, the field of view rapidly approaches zero, meaning that the rate of change in pixel length decreases because the pixel lengths approach zero themselves. Not only is the data trend similar in Figure 36 to the data trend shown in Figure 34, but the data points themselves are nearly identical.

$$\frac{\partial l}{\partial f} = -\frac{Zl_{sensor}\left(tan\left(\beta+\frac{tan^{-1}\left(\frac{l_{sensor}}{2f}\right)}{2n}\right)^2+1\right)}{4f^2n\left(\frac{l_{sensor}^2}{4f^2}+1\right)} \tag{58}$$

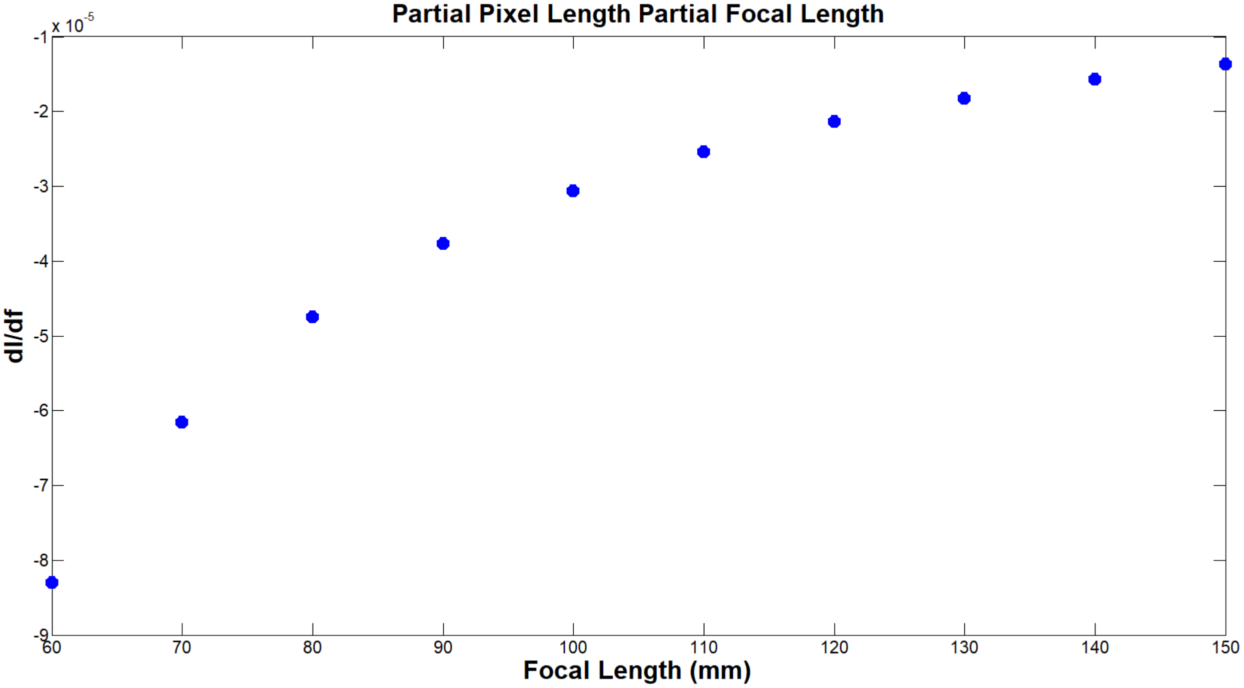


Figure 36. Sensitivity in the rate of change of pixel length with respect to focal length, rate of change trends towards zero as focal length increases (pivot angle greater than half angle of view, image half furthest from the camera).

Equation (59) below is the result of differentiating with respect to camera skew angle. The results are similar to Equation (56) from **Section 3.3**. As such, Equation (59) shows a positive correlation between the change of skew angle and the change of pixel length with pixel length increasing towards infinity as the skew angle approaches 90 degrees. The data trend from five to 50 degrees of skew for this partial derivative is shown in Figure 37.

$$\frac{\partial l}{\partial \beta} = Z\left(tan\left(\beta + \frac{\alpha}{2n}\right)^2 + 1\right) - Z(tan(\beta)^2 + 1) \tag{59}$$

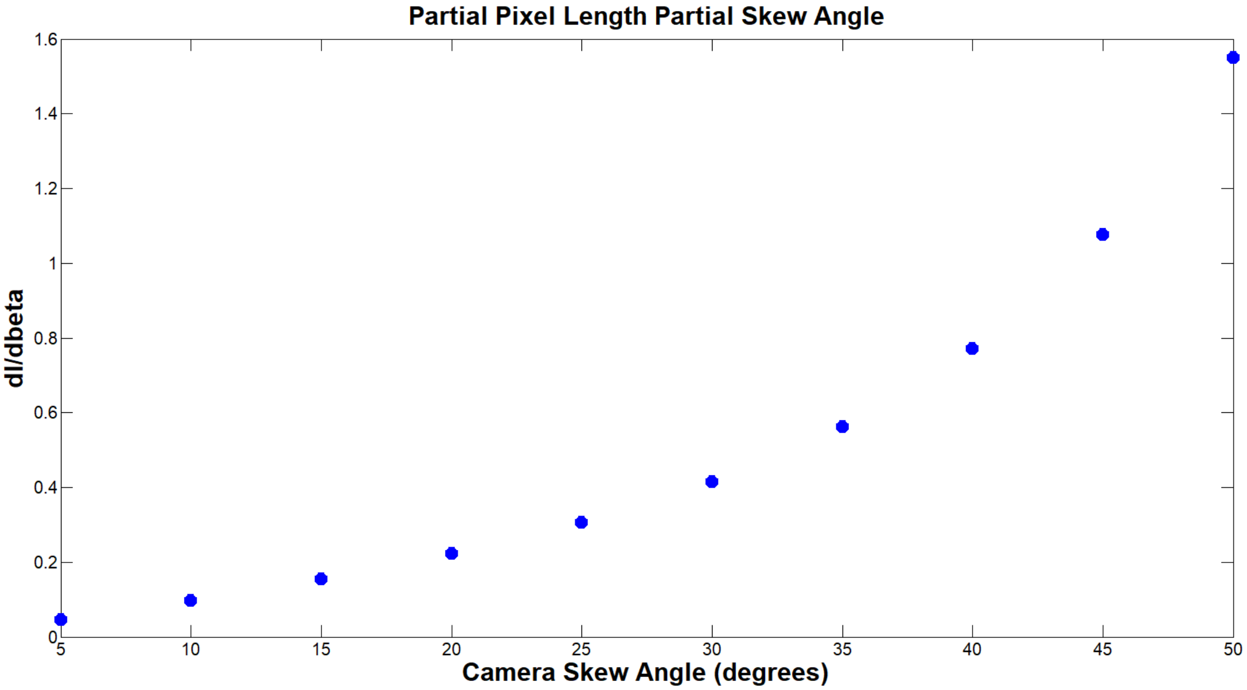


Figure 37. Sensitivity in the rate of change of pixel length with respect to camera skew, rate of change trends infinite as skew angle increases (pivot angle greater than half angle of view, image half furthest from the camera)

3.5 Pixel Length: Pivot Angle Less than Half the Angle of View (Behind the Camera)

As discussed in **Section 2.4.3** of this text, when the pivot angle of the camera is less than half the angle of view, the camera's field of view extends to both sides of the camera. From the left boundary of the field of view to the cross-over pixel, pixel length is determined using Equations (27-29). Substituting Equations (27) and (28) into Equation (29) yields the basis for the differentiation completed in this section. As this equation is a function of camera height, focal length and skew angle, partial derivatives will be taken with respect to all three variables.

$$l = \tan\left(\frac{\alpha}{2} - \beta\right) Z - \left(l_{sum} + \tan\left(\frac{\alpha}{2} - \left(\beta + \frac{\alpha}{2n}\right)\right) Z\right) \qquad (29)$$

The partial derivative of Equation (29) taken with respect to camera height yields Equation (60). As with all other partial derivatives taken with respect to camera height, Equation (60) is not a function of camera height, meaning that the rate of change in pixel length with respect to camera height is a linear function.

This function has a positive slope, resulting in pixel length increasing as the camera gets further from the flat plate.

$$\frac{\partial l}{\partial Z} = \tan\left(\frac{\alpha}{2} - \beta\right) - \tan\left(\frac{\alpha}{2} - \beta - \frac{\alpha}{2n}\right) \qquad (60)$$

If the partial derivative is taken with respect to the focal length, Equation (61) is the result. This equation is only valid for a small range of focal lengths, as the focal length must be short enough to ensure that the camera pivot angle is greater than half the angle of view. When this geometry constraint is met, the rate of change trend for pixel length resembles all of the other trends that have been previously discussed. A positive change in focal length is met with a decrease in pixel length with the rate of pixel length reduction decreasing as the camera focal length is increased. The data trend for this partial derivative is given below in Figure 38. Though this data follows the same general trend as the focal length partial derivative from the previous section, it is somewhat more erratic. This is likely due to the highly specific geometry conditions that must exist in order for this partial equation to be valid. For this analysis, camera skew angle is held to 2 degrees. A focal length of ten millimeters is extremely short; it gives an image with such a wide field of view that any changes to the input variables that occur are expected to result in large data perturbations. At the midrange of this dataset, the data trends closer to zero, especially at focal lengths of 35 and 45 millimeters. At focal lengths greater than 45 millimeters, the data trend can be seen to detract from zero. This is the point at which the focal length becomes long enough that this differential equation no longer becomes valid as the resulting image would no longer extend to the region behind the camera. While there may be concern regarding the large magnitude of the vertical axis within Figure 35, it is important to remember that it is a product of the constants in Equation (60). Reducing camera height (Z) and the length of the sensor would result in a reduction of slope data magnitude.

$$\frac{\partial l}{\partial f} = Z\left(\frac{l_{sensor}}{4f^2\left(\frac{l_{sensor}^2}{4f^2}+1\right)} - \frac{l_{sensor}n}{4f^2\left(\frac{l_{sensor}^2}{4f^2}+1\right)}\right)\left(tan\left(\frac{tan^{-1}\left(\frac{l_{sensor}}{2f}\right)}{2} - \beta - \frac{ntan^{-1}\left(\frac{l_{sensor}}{2f}\right)}{2}\right)^2 + 1\right) - \frac{Zl_{sensor}\left(tan\left(\frac{tan^{-1}\left(\frac{l_{sensor}}{2f}\right)}{2} - \beta\right)^2 + 1\right)}{4f^2\left(\frac{l_{sensor}^2}{4f^2}+1\right)} \tag{60}$$

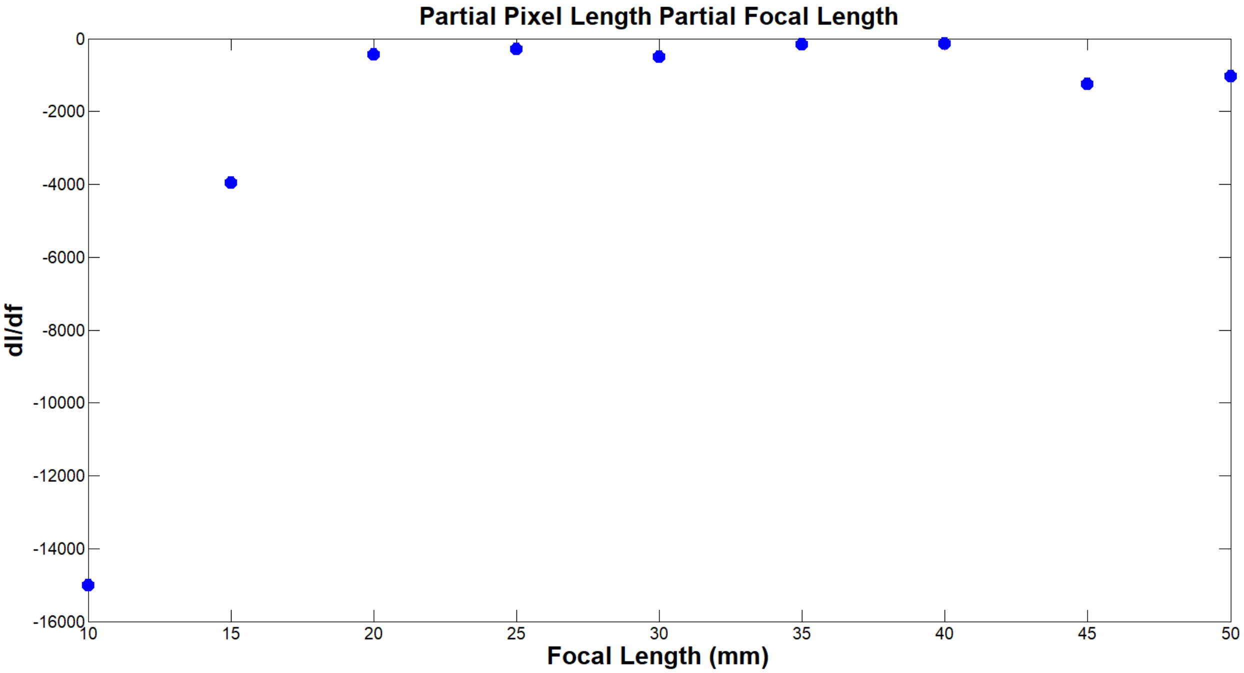


Figure 38. Sensitivity in the rate of change of pixel length with respect to focal length, rate of change trends towards zero as focal length increases (pivot angle less than half angle of view, image section behind camera).

Taken with respect to camera skew angle, the partial derivative of Equation (29) is given below in Equation (61). For the same reasons as the previous equation, Equation (61) is only valid over a small range of camera skew angles. Despite this range being small, a trend similar to the other camera skew angle partial derivatives does appear. An increase in camera skew angle results in an increase in pixel length. The increase in pixel length is more pronounced as beta increases, as the rate of change in pixel length approaches infinity as beta approaches 90 degrees. This is a bit of a moot point, however, as the equation is not valid past 5-10 degrees of camera skew (dependent on the angle of view). Figure 39 below shows the rate of change of pixel length as a function of camera skew angle. As expected the rate at which pixel length increases grows at a near exponential rate as skew angle grows. The data originally contained a tenth point at ten degrees of camera of skew, but this point was omitted as its exponential growth was so great that it changed the scale of the plot's vertical axis such that the remainder of the dataset became difficult to interpret.

$$\frac{\partial l}{\partial \beta} = Z\left(tan\left(\frac{\alpha}{2} - \beta - \frac{\alpha n}{2}\right)^2 + 1\right) - Z\left(tan\left(\frac{\alpha}{2} - \beta\right)^2 + 1\right) \tag{61}$$

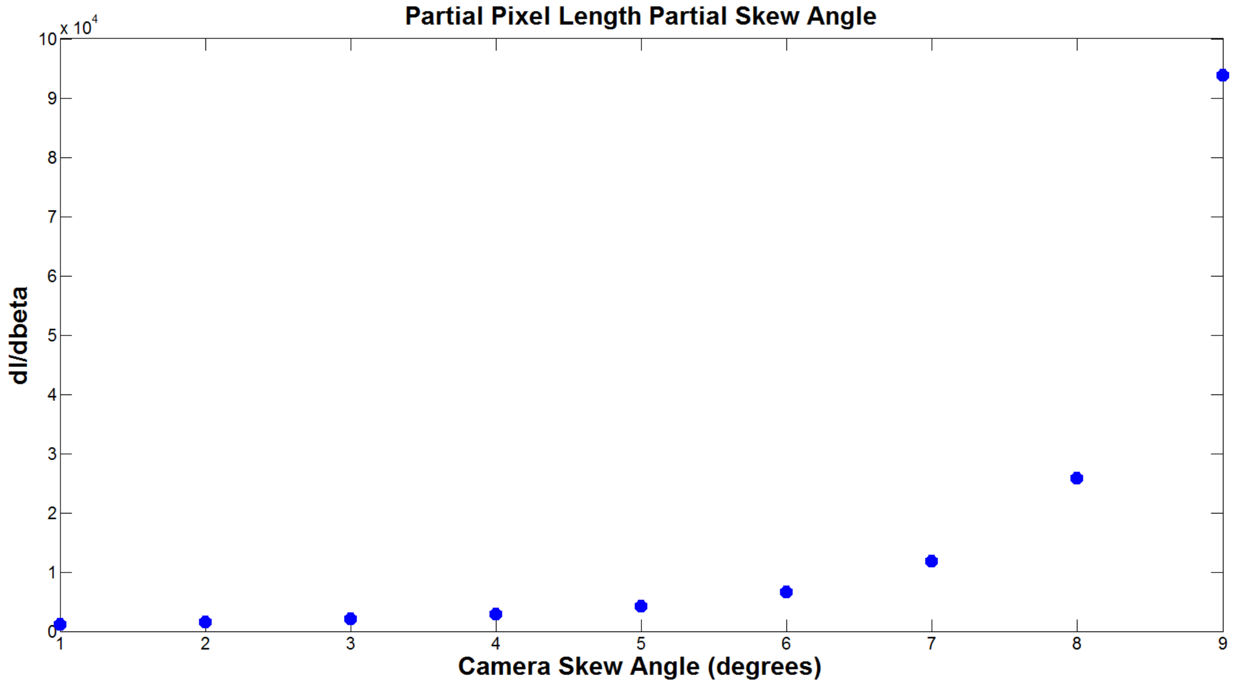


Figure 39. Sensitivity in the rate of change of pixel length with respect to camera skew, rate of change trends infinite as skew angle increases (pivot angle less than half angle of view, image section behind camera).

3.6 Pixel Length: Pivot Angle Less than Half the Angle of View (After Crossover Pixel)

Once the MATLAB solver has taken the crossover pixel into account, pixel length determination was accomplished much the same way as in previous geometries. The pixel length vector for this portion of the image begins at the edge of the crossover pixel's array and propagates to the center of the image. Equation (39), given below, is the equation used to define pixel length for this portion of the geometry. More information on calculating the pixel length for this geometry can be found in **Section 2.4.3** of this text. As Equation (39) is a function of camera skew angle, camera height and focal length, partial derivatives will be calculated with respect to all three variables.

$$l = \tan\left(\frac{\alpha}{2n} - \left(\frac{\alpha}{2} - \beta\right)\right)Z - \left(l_{sum} + \tan\left(abs\left(\frac{\alpha}{2} - \left(\beta - \frac{\alpha}{2n}\right)\right)\ \right)Z\right) \tag{39}$$

When Equation (39) is differentiated with respect to camera height, Equation (62) is the result. This equation shows a linear relationship between changes in camera height and pixel length. This can be determined analytically, as well as by observing that Equation (62) is not a function of camera height.

$$\frac{\partial l}{\partial Z} = -\tan\left(\frac{\alpha}{2} - \beta - \frac{\alpha n}{2}\right) - \tan\left(abs\left(\frac{\alpha}{2} - \beta + \frac{\alpha n}{2}\right)\right) \tag{62}$$

When the pixel length equation is differentiated with respect to focal length, Equation (63) is the result. Unlike the other pixel length partial derivatives taken with respect to focal length, it is not immediately obvious that the partial derivative trends towards zero as focal length increase. The data trend for this partial derivative is given below in Figure 40, and is somewhat more erratic than the trends shown previously. While the majority of the data points are near zero rate-of-change, there are some outliers, especially at 15 and 30 millimeters. It is difficult to determine the exact cause of these data perturbations, but it can likely be attributed to the nonuniform pixel distribution that occurs between different focal lengths. This geometry is unique in that the number of pixels to left and right of the crossover pixel vary as focal length changes. For every image taken in this thesis that meets this geometry constraint, the crossover pixel will be different. This adds more complexity to the equation, thus making it more susceptible to data spikes.

$$\frac{\partial l}{\partial f} = Z\left(\frac{l_{sensor}}{\#2} - \frac{nl_{sensor}}{\#2}\right)\left(tan\left(\frac{tan^{-1}\left(\frac{l_{sensor}}{2f}\right)}{2} - \beta - \frac{ntan^{-1}\left(\frac{l_{sensor}}{2f}\right)}{2}\right)^2 + 1\right) + Zsgn(\#1)\left(\frac{l_{sensor}}{\#2} + \frac{nl_{sensor}}{\#2}\right)\left(tan\left(abs(\#1)\right)^2 + 1\right) \tag{63}$$

Where

$$\#1 == \frac{tan^{-1}\left(\frac{l_{sensor}}{2f}\right)}{2} - \beta + \frac{ntan^{-1}\left(\frac{l_{sensor}}{2f}\right)}{2}$$

$$\#2 == 4f^2\left(\frac{l_{sensor}^2}{4f^2}+1\right)$$

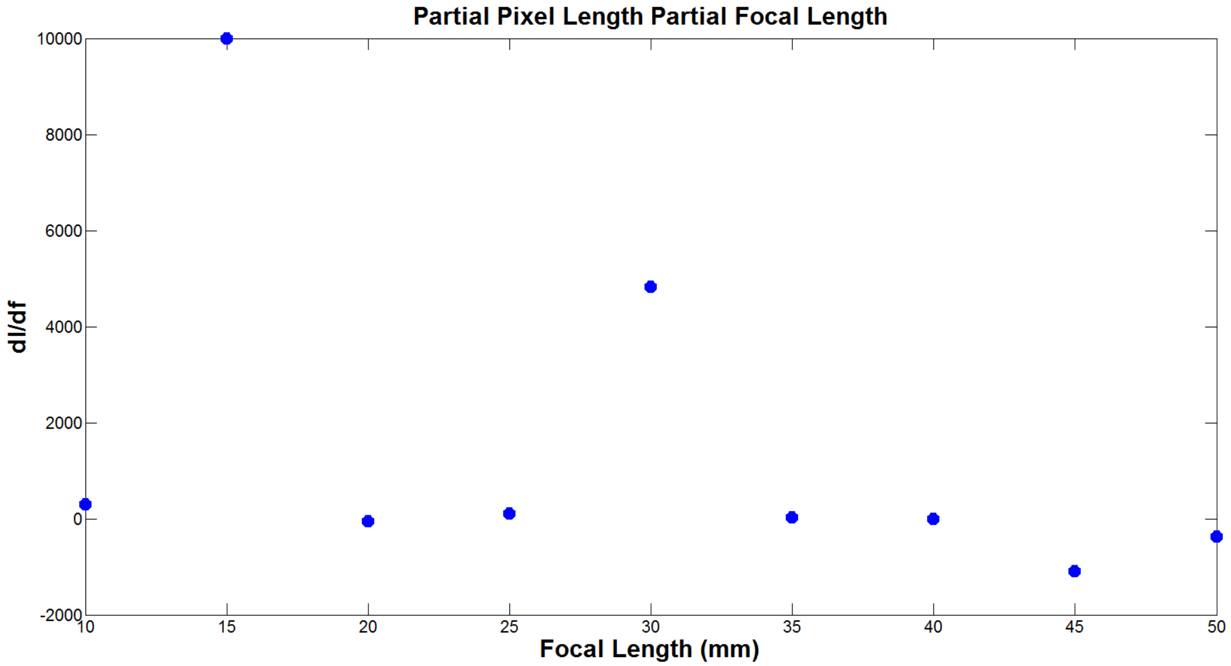


Figure 40. Sensitivity in the rate of change of pixel length with respect to focal length, rate of change trends towards zero across majority of plotted focal lengths (pivot angle less than half angle of view, image section closest to camera, after cross-over pixel).

Taken with respect to camera skew angle, the partial derivative of Equation (39) is given below in Equation (64). For the same reasons as the previous equation, Equation (64) is only valid over a small range of camera skew angles. Figure 41 gives the data trend of the partial derivative taken with respect to the camera skew angle. Conveniently, the pixel length trend is identical in this portion of this geometry as it is for the geometry portion in front of the crossover pixel. The rate at which the pixel length changes as a function of camera skew angle increases at a near exponential rate. Though the partial derivative trend in this figure is only represented up to camera skew angles of nine degrees, it is clear that the rate of change of pixel length as a function skew angle trends towards infinity as skew angle grows.

$$\frac{\partial l}{\partial \beta} = Z\left(tan\left(\frac{\alpha}{2}-\beta-\frac{\alpha n}{2}\right)^2+1\right)+Zsgn\left(\frac{\alpha}{2}-\beta-\frac{\alpha n}{2}\right)\left(tan\left(abs\left(\frac{\alpha}{2}-\beta-\frac{\alpha n}{2}\right)\right)^2+1\right) \tag{64}$$

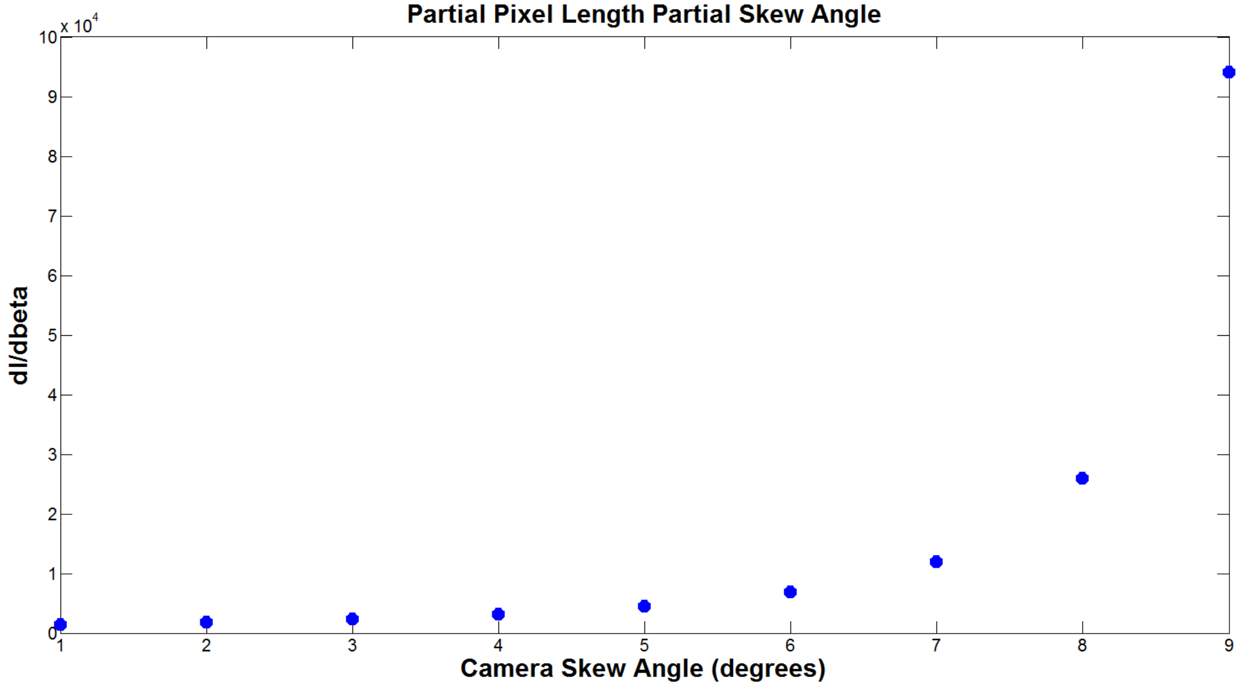


Figure 41. Sensitivity in the rate of change of pixel length with respect to camera skew, rate of change trends infinite as skew angle increases (pivot angle less than half angle of view, image section image section closest to camera, after cross-over pixel).

3.7 Pixel Length: Pivot Angle Less than Half the Angle of View (Furthest from Camera)

This portion of the image has the exact geometry as the geometry covered in **Section 3.4** of this text.

Please refer to that section for information regarding the partial differentiation of this geometry.

4.0 MATLAB SOLVER ANALYSIS: CENTERLINE DETERMINATION

This section is devoted to presenting the results of the MATLAB solver's centerline determination analysis. Centerline accuracy is defined by how closely centerlines defined by the MATLAB solver are to centerlines hand drawn by a user over the same image. As the scope of this thesis is limited to flat plate geometries exposed to omnidirectional air flow, accuracy is measured by comparing the position and propagation of a linear line that spans from the leading edge of the first oil fringe to the trailing edge of the final oil fringe in the smear.

The hand drawn centerlines are created by importing an image of an oil smear into Microsoft Paint and hand-drawing a line with a computer mouse down the center of the oil smear with the use of the built in linear line tool. The line extends from the leading edge of the first fringe to the trailing edge of the final fringe. The MATLAB-defined centerlines are defined as the linear line that extends from the leading edge point of the first fringe to the trailing edge point of the final fringe.

Comparison of MATLAB and hand-drawn centerlines is done by examining the pixel coordinates of the centerline midpoint as well as comparing the slope of the two lines. The midpoint of each centerline is defined as the point on the line equidistant from both the leading edge of the first fringe and the trailing edge of the final fringe. In Microsoft Paint, determining the location of the leading and trailing edge points is not automated, as it was felt that using the built-in features of Microsoft Paint were sufficient. Once a photo of an oil smear is opened into Microsoft Paint, the pixel location of a user's cursor relative to the upper left corner of image is displayed on the bottom of the Microsoft Paint window. By placing the mouse cursor over the location where the centerline intersects the leading edge of the oil smear's first fringe and the trailing edge of the final fringe, the user can determine the pixel location of the leading and trailing edge points. With the pixel locations known, the midpoint can then be determined by averaging the X and Y components of the leading and trailing edge points

In MATLAB, all leading and trailing edge locations are determined during the calculation of the fringe spacing. Figure 42 below shows both methods of centerline determination on an identical oil smear. The top image is a hand drawn centerline created in Microsoft Paint, while the second depicts the leading and trailing edge points generated by MATLAB.

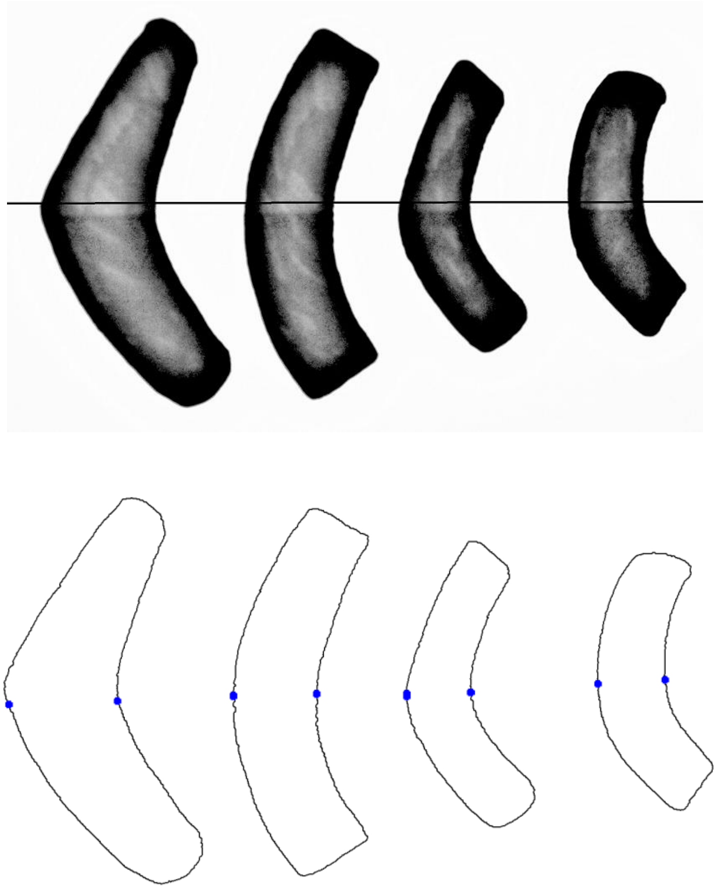


Figure 42. Hand drawn centerline over a representative oil smear (top), MATLAB analysis of the leading and trailing fringe edge points of the same representative oil smear (bottom).

4.1 Centerline Determination: One-Dimensional Pixel-Length Geometries (X-Skew)

In the initial stages of the MATLAB solver, images were restricted to oil flows propagating horizontally across the field of view. This was done so that pixel length needed only to be calculated along the horizontal axis of the image. Since this type of geometry is what the MATLAB solver was held to in its initial stages, it only makes sense that analysis should begin here as well.

For the first set of images a camera-tripod assembly is set up such that the lens focal point is 36-inches from a hand-drawn (representative) oil smear adhered to a vertical wall. The initial image is taken from a perfectly orthogonal point of view with zero camera skew relative to the center of the oil smear. In each successive image, the camera assembly is moved in $1^{1/8}$ inch increments perpendicular to the vertical wall on which the oil smear is adhered to. After each camera movement, the orientation of the camera is pivoted until the center of the lens points directly at the center of the oil smear. A total of 8 images were taken in this data set, all with a lens focal length of 85 mm. The oil smear used for analysis in this section is shown above in Figure 39. The top image in Figure 39 appears to be washed out because high contrast is applied to the image in post processing in order to remove any errant blemishes around the fringes, as the blemishes prevent the MATLAB solver from properly determining the edges of each fringe. The experimental setup used to obtain this dataset's images is shown in Figure 43.

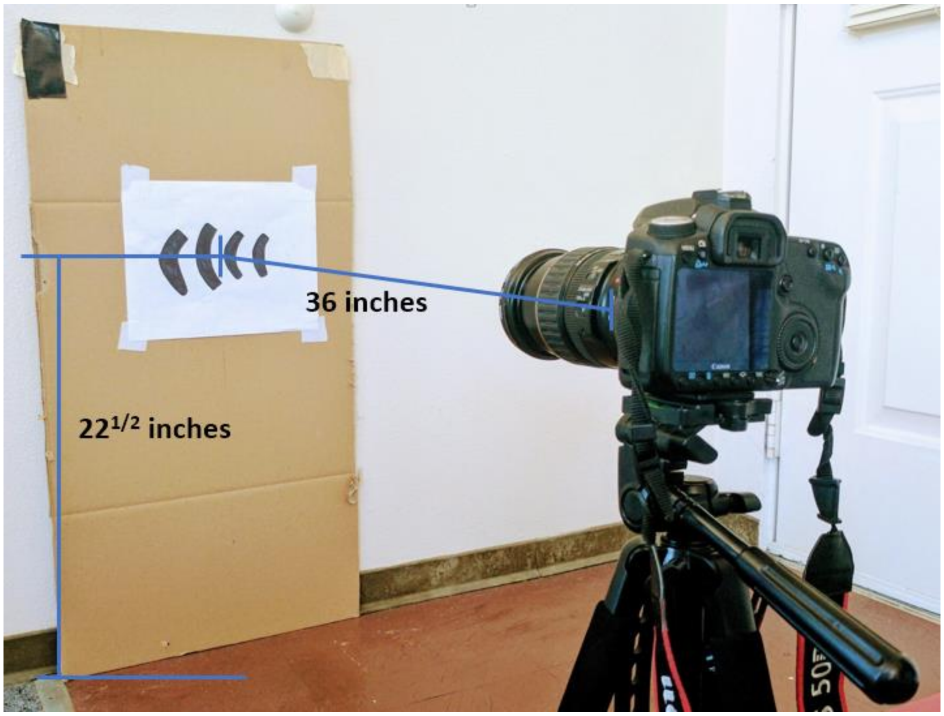


Figure 43. Camera-fringe setup for image collection of horizontal oil propagation, this specific photograph shows the camera in an orthogonal location.

The results of the centerline comparison are summarized below in Table 3 and Figures 44 and 45. Analysis of the midpoint location error is separated into its X and Y pixel components. In addition to midpoint analysis, the slope of the hand-drawn and MATLAB created lines are compared to one another. In the Figure 41, the blue trend line is the position error between the hand-drawn and MATLAB-derived centerlines in the X-dimension, while the green trend line depict the midpoint error in the Y-dimension. As CXWIN5G utilizes a similar hand-drawn method, the error calculations were completed under the assumption that the hand-drawn centerlines represent theoretical data, while the MATLAB-created lines represent empirical data. Immediately obvious from Figure 44 is the spike in error at 3.6 degrees of skew. This seems to be an anomaly in the data, as the remaining X-dimension data shows quite low position error between the hand-drawn and MATLAB-created midpoints. If one were to exclude the 9.13% X-coordinate data point, the average midpoint location error in the Y-coordinates is higher than the error in the X-coordinates. In total, the average midpoint position error for this dataset is 1.44% in the X-dimension, and 1.29% in the Y-dimension. Excluding the 9.13% error point reduces the overall X-coordinate error to 0.15%. The greater average error in the Y-dimension is attributed to the physical characteristics of the images in this dataset. Because the oil smears propagate horizontally across the image, the leading edge of the first fringe and the trailing edge are predominantly vertical in their orientation. Any error movement along the curve at either point results in a greater Y-direction change than X-direction change.

Table 3. One-dimensional oil propagation centerline pixel coordinate comparison (X-skew)

Camera Skew	Leading Point (X,Y)	Trailing Point (X,Y)	Slope	Midpoint
0° (Hand)	(371,813)	(1804,767)	0.0321	(1087,790)
0° (MATLAB)	(389,827)	(1785,768)	0.0423	(1087,784)
1.8°	(387,802)	(1852,766)	0.0246	(1120,784)
1.8°	(409,826)	(1835,769)	0.0400	(1122,798)
3.6°	(387,810)	(1864,769)	0.0278	(1226,790)
3.6°	(407,823)	(1850,764)	0.0409	(1114,794)
5.4°	(350, 808)	(1833, 741)	0.0452	(1092,755)
5.4°	(365,807)	(1820,749)	0.0399	(1093,778)
7.1°	-	-		

7.1°	-	-		
8.9°	(351,795)	(1866,741)	0.0356	(1109,768)
8.9°	(370,801)	(1850,744)	0.0385	(1110,773)
10.6°	(339,802)	(1853,735)	0.0443	(1096,769)
10.6°	(352,795)	(1837,734)	0.0411	(1095,765)
12.3°	(289,777)	(1821,729)	0.0313	(1055,753)
12.3°	(313,795)	(1806,737)	0.0388	(1060,766)

The slope analysis performed for this dataset produced a significant amount of data variation. The slope error ranged between 7.22% and 62.6%. The poor slope error is a product of the horizontal nature of oil smear propagation. Horizontal oil smear propagation results in centerlines having a slope of approximately zero. If a centerline has a slope of zero, the error between the zero slope centerline and a centerline of non-zero slope is infinite. Out of the 14 centerlines created, the maximum slope is 0.0452. In terms of the measurement capabilities of this thesis, the slope of the lines in this dataset are quite near to zero. Because both the hand-drawn and MATLAB-created centerlines are so close to zero slope, any slope variance in the lines will produce large error. This can be visualized by imaging a line of zero slope. The percent error between the zero-slope line and any other non-zero slope line is infinite, no matter how close horizontal that line may be.

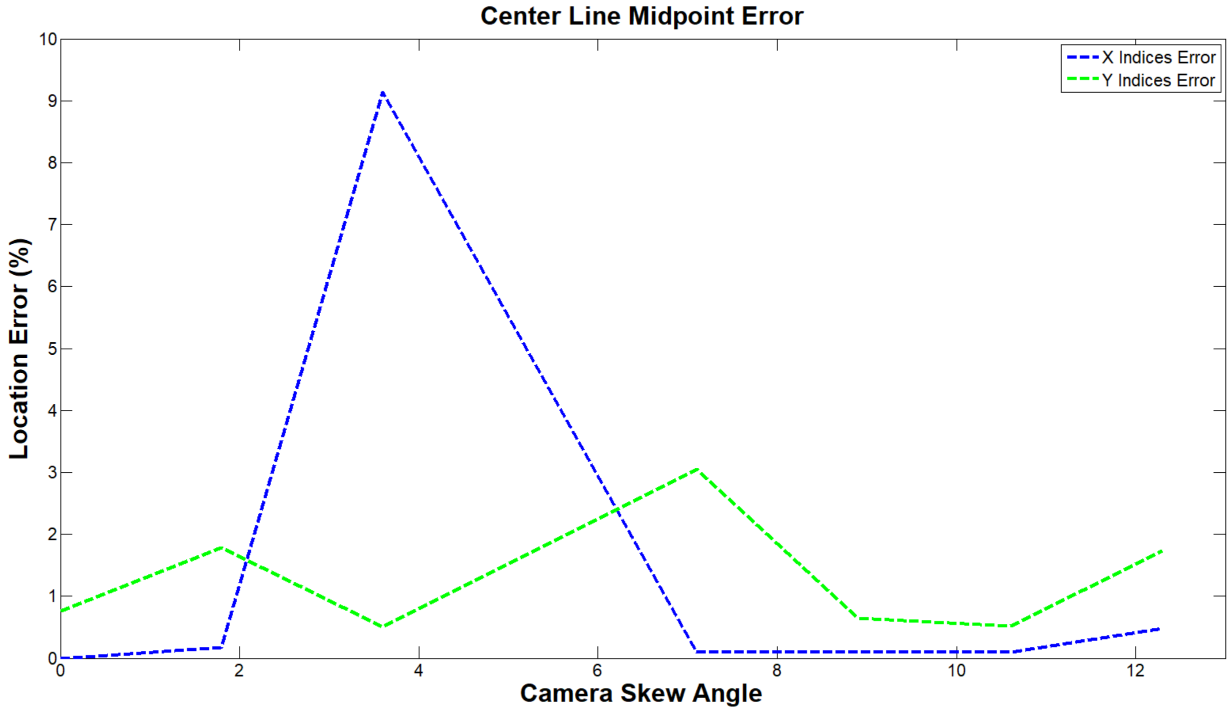


Figure 44. Centerline Midpoint pixel coordinate error, horizontal oil smear propagation, X-dimension camera pivot angles from zero to 12.3 degrees.

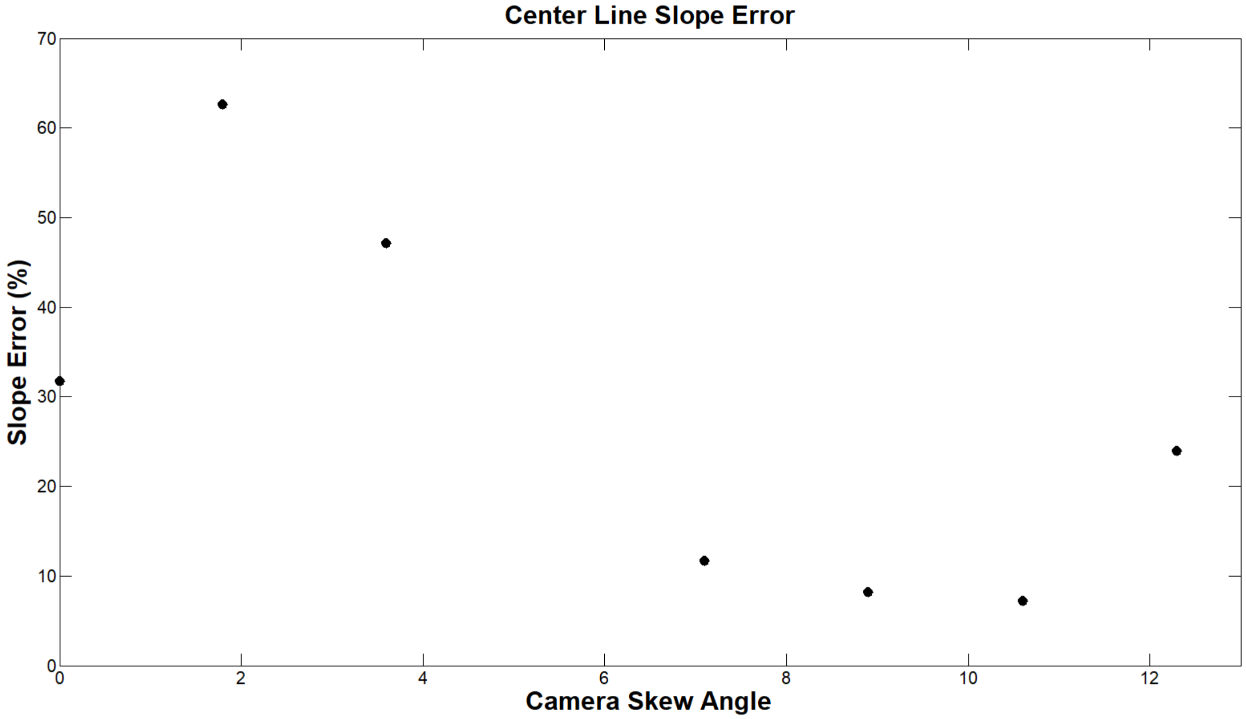


Figure 45. Centerline slope error, horizontal oil smear propagation, X-dimension camera pivot angles from zero to 12.3 degrees.

One conclusion that can be drawn from the slope analysis of this dataset is that there is a greater degree of slope consistency in the MATLAB-defined lines than there is in the hand-drawn centerlines. The standard deviation of the seven hand drawn centerline slopes is 0.0079, while the standard deviation of the seven MATLAB-defined slopes is 0.0013. Every time the camera is moved, its motion is predominantly along the same axis as the fringe centerlines. This means that pixel distortion from camera skew occurs predominantly in the X-direction, not in the Y-direction. Because the centerlines have a much greater X-component length than Y-component length, the Y-component differential between the leading and trailing edge is the primary slope designator. What this means is that increasing camera skew has little effect on centerline slope in this camera-fringe geometry. Because of this, it can be concluded that the poor slope consistency of the hand-drawn centerlines is attributed to the human error associated with hand sketching and the horizontal nature of the lines' propagation.

In order to further quantify the accuracy potential of hand drawing oil smear centerlines, one can use the human repeatability analysis performed by Robert Ehrmann in his thesis, 'Development of Measurement Methods for Application to A Wind Tunnel Test of an Advanced Transport Model'[3]. In his thesis, Robert utilized CXWIN5G to analyze oil smears across the NASA/Cal Poly Advanced Model for Extreme Lift and Improved Aeroacoustics wind tunnel model. As part of his error analysis, he quantified the accuracy of hand-drawn lines by performing statistical analysis on oil smear centerlines drawn by multiple people. The results of his analysis are presented in Table 4 below. The second column is the percent error associated with hand measuring fringe spacing, while the third column is the error associated with centerline propagation, in degrees. Ehrmann's test subjects were given a piece of paper with 10 test marks and one central mark. The test subjects were then asked to measure the distance from the central mark to all ten test marks as well as draw linear line from the central mark to the test marks. The subjects were give a standard ruler to make measurements and to draw the linear lines with. Part of Robert's analysis was proving, to a 95% confidence interval, that the ±19.36 degree error was a statistical outlier. As such, that data point will be omitted from consideration in this thesis. The results summarized in Table 4 show directional variation error between ±1.36 and ±2.34 degrees.

Table 4. Human repeatability study, determination of fringe spacing and line drawing accuracy

Point	$d\Delta s/\Delta s$ (%)	± 2σ (°)
1	1.68	1.60
2	2.03	2.03
3	1.00	1.74
4	2.07	1.59
5	2.09	1.36
6	2.34	19.36
7	1.87	1.90
8	1.82	1.55
9	1.90	1.83
10	1.25	1.70

While the midpoint location and slope analysis done in this thesis does not directly equate to the directional error analysis accomplished by Ehrmann, we can still use his analysis to validate the results of the horizontal oil smear propagation dataset. The average horizontal length of the hand-drawn centerlines in this data set is 1488 pixels. Using Robert's directional uncertainty of ±2.34 degrees, this means that it can be concluded that pixel error in the vertical direction at the centerline endpoint is up to 61 pixels with a 95% confidence interval. This analysis does not account for the possibility of centerline position error at its origin, as in Ehrmann's analysis propagation of the test subject's centerlines all began from the same point. Figure 43 below depicts the average hand-drawn centerline (solid blue) of the dataset presented in Table 1. In addition to the average centerline, 2.34 degree error bands (dotted blue) are included in Figure 46. All seven MATLAB-defined centerlines (green) are also included in the figure so that their accuracy can be gauged.

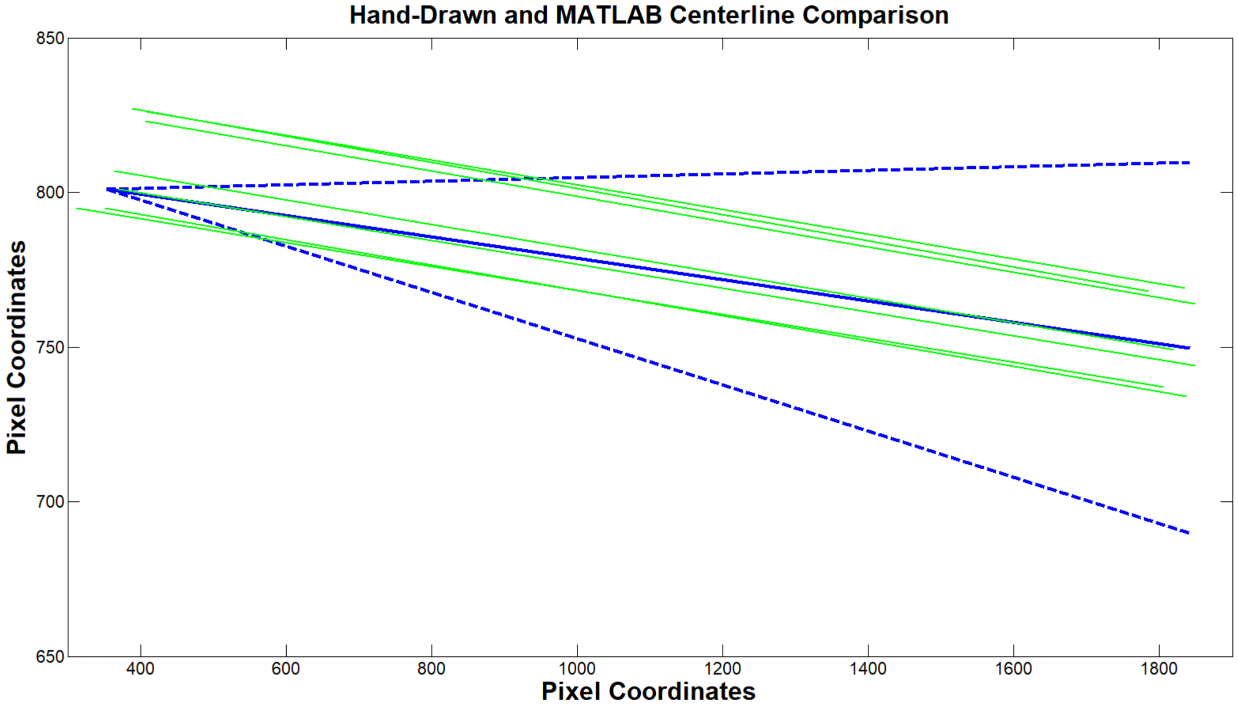


Figure 46. Average hand-drawn centerline with 2.34 degree error bands, All seven MATLAB centerlines; horizontal oil smear propagation with X-dimension camera pivot angles from zero to 12.3 degrees.

Because the human repeatability analysis conducted by Robert Ehrmann does not account for error at the centerline's origin, the accuracy of the MATLAB centerlines at the leading edge cannot be verified by this method alone. That being said, it was established in previous paragraphs that for centerlines of this length, the pixel coordinate error in the vertical direction can be expected to be up to 61 pixels. Comparing the position of the left end of the seven MATLAB centerlines to the origin of the average hand-drawn centerline, all seven of them have a pixel location closer than 61 pixels. In terms of the propagation of the MATLAB centerlines, the right boundary of all seven fall within the error bounds of the hand-drawn centerlines. Because of this, it can be concluded that the centerlines created using the MATLAB solver are just as accurate as hand drawn centerlines, and that their placement on the oil smear images is more consistent.

4.2 Centerline Determination: Two-Dimensional Pixel-Length Geometries (No Camera Skew)

In the previous section, analysis is limited to horizontal oil fringe propagation. Because the capability of the MATLAB solver includes the ability to determine flat plate pixel length in both the X and Y direction, analysis in this section features two-dimensional fringe propagation.

The images in this dataset are taken at a camera height of 36 inches, just as the images taken for analysis in **Section 4.1**. For this portion of the experiment, camera skew was held to zero degrees and four images were taken of the same oil smear at four different propagation orientations. The experimental setup in taking these images is nearly identical to the setup shown in Figure 43, the only difference being that fringes are rotated about their center point. The four images used for analysis are shown below in Figure 47. The pixel coordinates of the centerline endpoints are given below in Table 5.

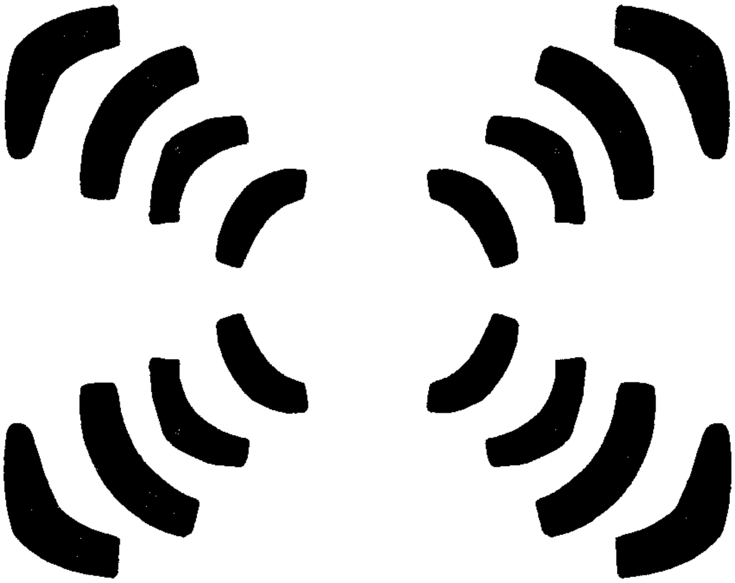


Figure 47. Propagation orientation of the four representative oil smears, each oil smear propagates from a different image quadrant with the first image originating in the first quadrant, the second image originating in the second quadrant, and so on.

Table 5. Two-dimensional oil propagation centerline pixel coordinate comparison (no skew)

Smear Propagation	Leading Point (X,Y)	Trailing Point (X,Y)	Slope	Midpoint
First Quadrant Origin (Hand)	(1201, 2098)	(4170, 693)	0.4732	(2686,1396)
First Quadrant Origin (MAT)	(1212, 2086)	(4159, 697)	0.4713	(2686,1392)
Second Quadrant Origin	(439, 296)	(1813, 1262)	0.7031	(1126,779)
Second Quadrant Origin	(437, 322)	(1810, 1252)	0.6773	(1124,787)
Third Quadrant Origin	(482, 1312)	(1662, 495)	0.6924	(1074,904)
Third Quadrant Origin	(465, 1311)	(1671, 476)	0.6924	(1068,894)
Fourth Quadrant Origin	(853, 403)	(1952, 1407)	0.9136	(1403,905)
Fourth Quadrant Origin	(880, 398)	(1953, 1391)	0.9254	(1417,895)

Like the pixel location error discussed in the previous section, analysis in this section is performed by comparing the slope and midpoint location of the MATLAB and hand-drawn centerlines. The error is calculated exactly as it was in the previous section; with the MATLAB results representing experimental data and the hand-drawn pixel locations representing theoretical data. The average midpoint pixel coordinate error across the four images in the X and Y dimension is 0.43 percent and 0.88 percent, respectively. For the slope magnitude, the average error is 1.34 percent, with a minimum recorded error of 0.00 percent and a maximum error 3.66 percent. The midpoint error trend data is presented graphically in Figure 48 below, and the slope error trend is presented in Figure 49.

Like the data set discussed in the previous section, midpoint coordinate error is significantly less in the X-dimension than it is in the Y-dimension. This is explained by the curvature of the oil fringes. At the leading and trailing edge of each fringe, the rate of change of the edge of the fringe is greater in the Y-direction than it is in the X-direction. This means that any deviation from the true fringe center by either the MATLAB-created or hand-drawn centerlines will result in greater X-dimensional error than Y-dimensional error. Because this dataset contains four completely different centerlines, performing the human repeatability that is part of the previous section is not feasible. It will in included in the preceding sections, as their datasets feature centerline propagation that is more uniform.

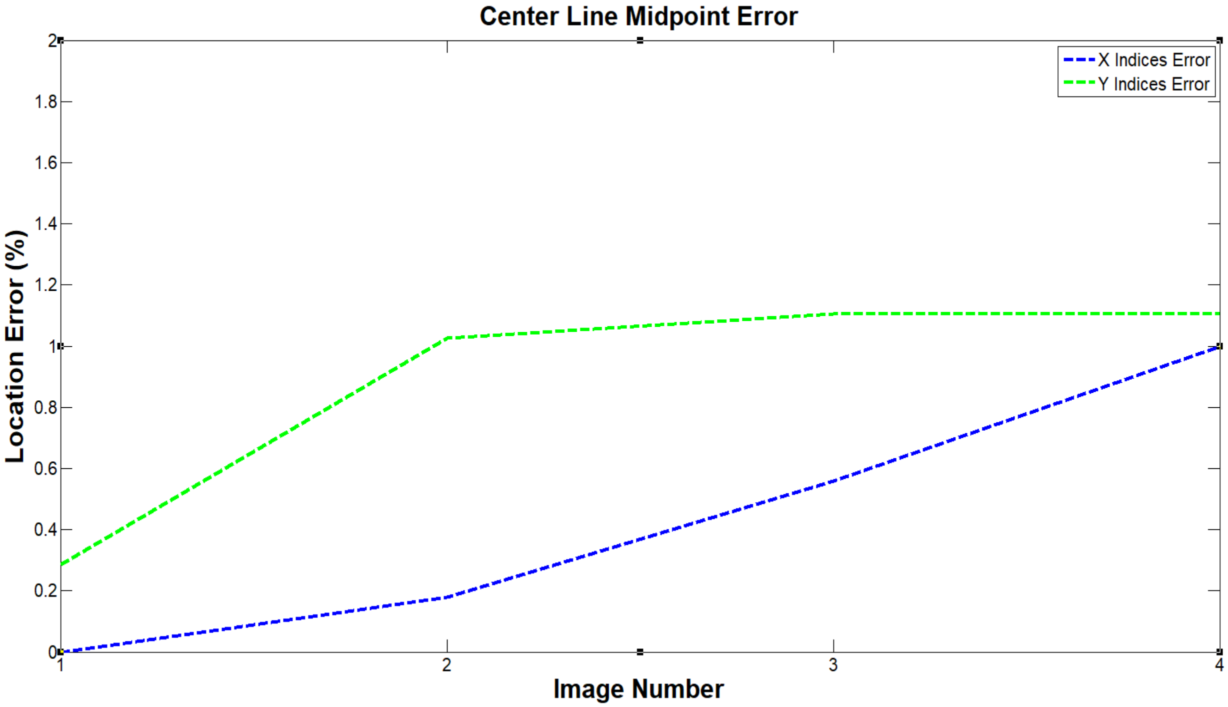


Figure 48. Centerline Midpoint coordinate error, horizontal and vertical oil smear propagation, zero camera skew.

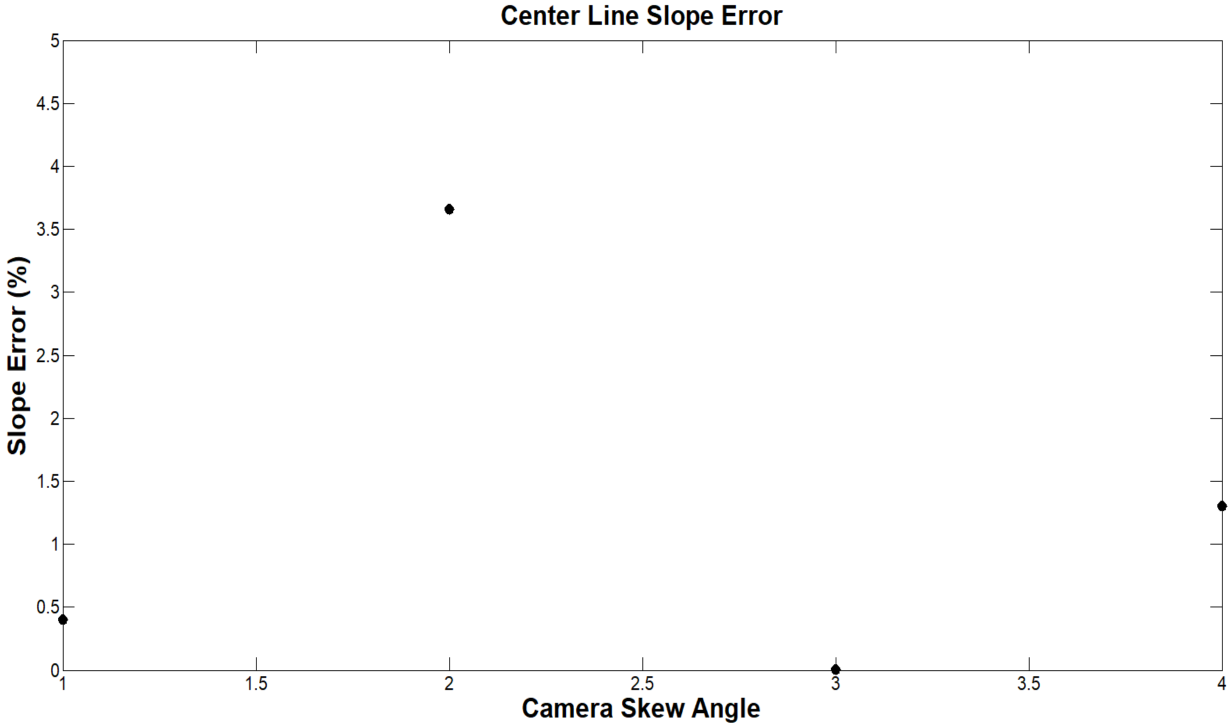


Figure 49. Centerline slope error, two-dimensional oil smear propagation, orthogonal camera position, four images.

4.3 Centerline Determination: Two-Dimensional Pixel-Length Geometries (X-Skew)

In order to determine the true capability of the solver, it is important to incorporate two-dimensional oil propagation and camera skew into one dataset. The camera setup required for capturing the images in this dataset is nearly identical to the horizontal fringe propagation case examined in **Section 4.1**, the only major difference being that the oil smear is rotated approximately 45 degrees counterclockwise. For this data set, the camera height remains at 36 inches, but the lens focal length is shortened to 70 mm to ensure that the entire oil smear appears in the image at higher degrees of camera skew. Like the horizontal propagation case, the hand-drawn representative oil smear is adhered to a vertical surface, with the center of the oil smear at an equal height off the ground as the center of the lens-camera assembly. For the first image, the vertical centerline of the camera lens is directly in line with the center of the oil fringes, producing a zero skew image. The no-skew orthogonal image is given below in Figure 50. This image, as well as the seven other images analyzed in this section, depict oil smears that originate in the second quadrant of the photograph and propagate down to the photograph's fourth quadrant. For each subsequent image after the orthogonal one, the camera assembly is shifted $1^{1/8}$ inches parallel to the vertical wall, and the camera is pivoted so that the center of the oil smear is aligned with the center of the camera's field of view.

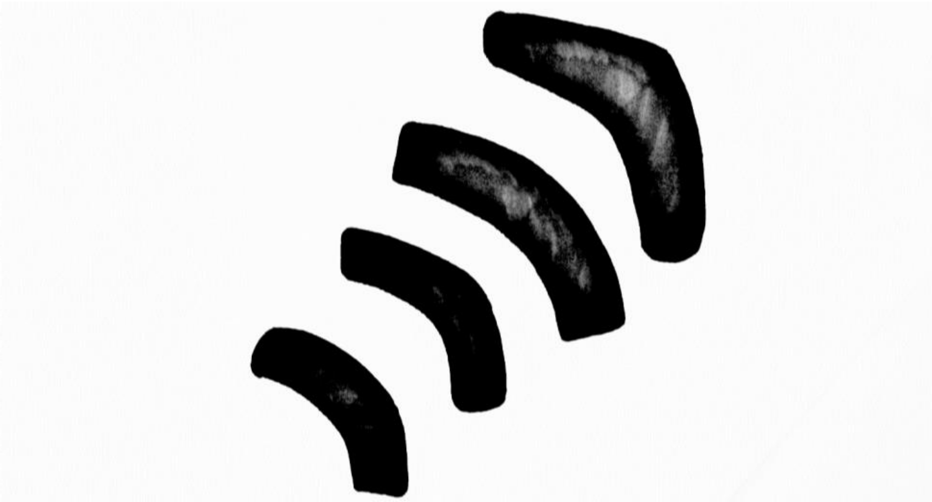


Figure 50. Representative oil smear featuring horizontal and vertical fringe propagation, orthogonal camera position, smear origin in second quadrant of the image.

The analysis of the eight photos is summarized below in Table 6 as well as in Figures 51 and 52. As in the previous sections, the accuracy of the MATLAB solver for two-dimensional oil smear propagation is measured by comparing MATLAB-created centerlines to their hand-drawn counterparts. The error between the two methods of centerline detection is produced under the assumption that the hand-drawn centerlines represent theoretical data, while the MATLAB-created lines are experimental data.

The average midpoint error for this dataset is 0.41 percent in the X-direction and 1.48 percent in the Y-direction. In addition, the average slope magnitude error is 2.48 percent. Like the data presented in **Section 4.2**, midpoint error is greater in the Y-dimension. Despite the fact that the oil smear is rotated by 45 degrees in this dataset, the leading and trailing edge curves are predominantly vertical in their orientation resulting in a greater predisposition for error position error in the Y-dimension. The relatively low slope error of this dataset and the dataset of the previous section show that the conclusions drawn from the vertical oil smear slope analysis in **Section 4.1** are true. Performing slope error analysis on (nearly) horizontal lines results in large errors simply because the slopes of the centerlines are so close to zero. Once analysis is performed on oil smears with non-zero centerline slope, the error diminishes to acceptable levels.

Table 6. Two-dimensional oil propagation centerline pixel coordinate comparison (X-skew)

Camera Skew	Leading Edge Point (X,Y)	Trailing Edge Point (X,Y)	Slope	Midpoint
0° (Hand)	(666, 310)	(1532,1225)	1.0566	(1099,768)
0° (MATLAB)	(670,340)	(1520,1219)	1.0341	(1095,780)
1.8°	(690,295)	(1572, 1210)	1.0374	(1131,753)
1.8°	(695,318)	(1561,1203)	1.0219	(1128,761)
3.7°	(674, 279)	(1572,1200)	1.0256	(1123,740)
3.7°	(676,307)	(1555,1198)	1.0137	(1116,753)
5.5°	(676,286)	(1570,1214)	1.0380	(1123,750)
5.5°	(678,315)	(1562,1206)	1.0079	(1120,761)
7.3°	(716,272)	(1615,1208)	1.0412	(1656,740)
7.3°	(717,303)	(1605,1198)	1.0079	(1161,751)

9.1°	(687,261)	(1598,1198)	1.0285	(1143,730)
9.1°	(688,291)	(1586,1192)	1.0033	(1137,742)
10.9°	(679,258)	(1584,1191)	1.0309	(1132,725)
10.9°	(679,283)	(1594,1165)	0.9858	(1137,734)
12.7°	(679,252)	(1598, 1185)	1.0152	(1139,719)
12.7°	(681,281)	(1586,1180)	0.9934	(1134,731)

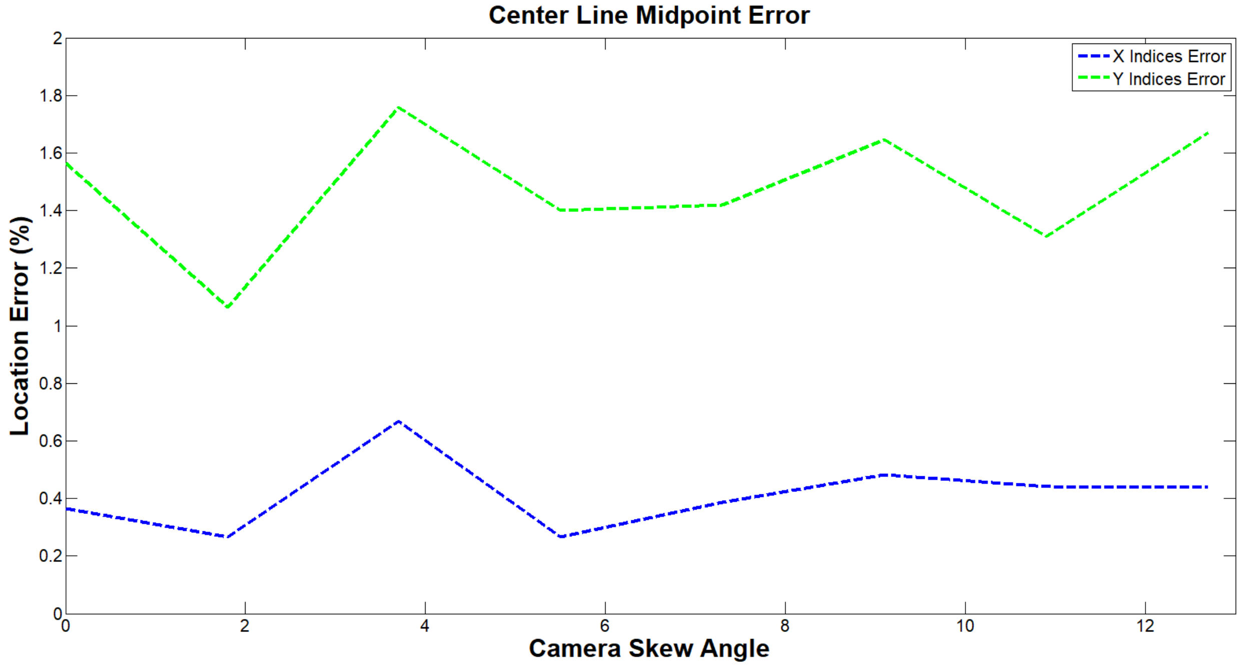


Figure 51. Centerline midpoint pixel coordinate error, horizontal and vertical oil smear propagation, X-dimension camera pivot angles from zero to 12.7 degrees.

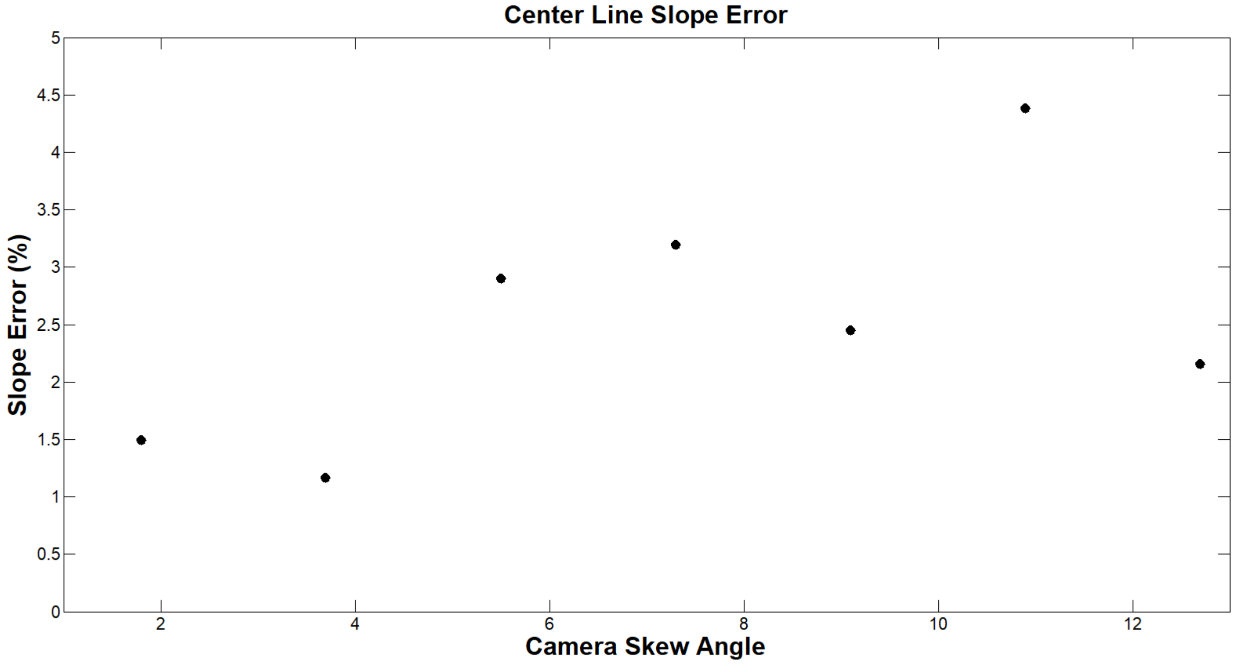


Figure 52. Centerline slope error, horizontal oil smear propagation, X-dimension camera pivot angles from zero to 12.7 degrees.

The human repeatability study conducted by Robert Ehrmann can be used to compare the hand-drawn and MATLAB centerlines of this dataset as well. The average length of the hand drawn centerlines is 1475 pixels. Using Ehrmann's directional error of 2.34 degrees, this means that the error band at the end of the hand drawn centerline has a range of 60 pixels. The average hand-drawn centerline with the error band is given in Figure 53 Every MATLAB centerline is also presented on the figure to so that their position accuracy relative to the hand-drawn centerlines can be determined. Again, Ehrmann's analysis does not account for position inaccuracies at the origin of the hand-drawn centerlines. At the end point (left) of the centerlines only one of the eight MATLAB centerlines lie outside the error bounds of the hand-drawn centerlines. Furthermore, out of the eight MATLAB centerlines, only one has origin coordinates that are greater than 60 pixels away from the average hand-drawn centerline origin. In other words, only one of the eight MATLAB centerlines had pixel coordinates outside of the 2.34 degree error boundary established by Robert Ehrmann's repeatability analysis.

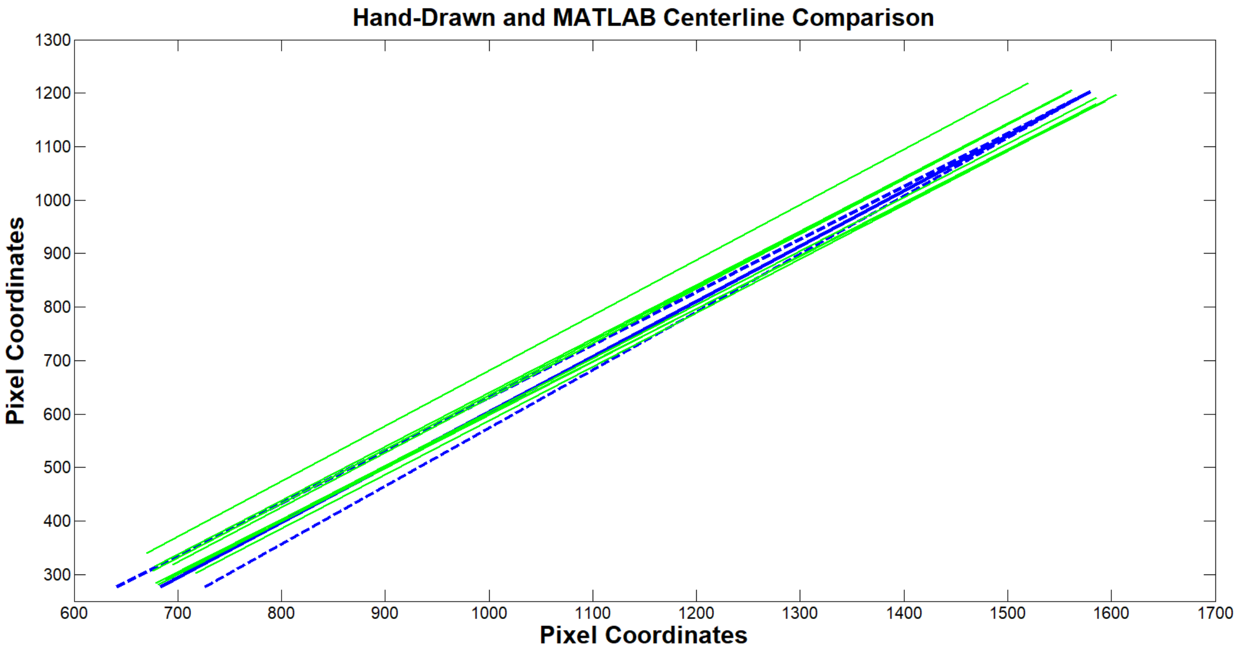


Figure 53. Average hand-drawn centerline with 2.34 degree error bands, All seven MATLAB centerlines; 2D oil smear propagation with X-dimension camera pivot angles from zero to 12.7 degrees.

4.4 Centerline Determination: One-Dimensional Pixel-Length Geometries (Y-Skew)

The second half of centerline analysis verifies the MATLAB solver's ability to accurately account for skew in the Y-dimension. This dataset contains ten images ranging in skew values from zero to 13.2 degrees. These ten images capture an oil smear propagating in a left-to-right horizontal manner. The initial zero-degree image was captured using the same camera-fringe geometry shown in Figure 43, with the oil smear adhered to a vertical surface equal in height to the center of the camera lens and the camera located 32.5 inches away from the fringe. For each of the nine subsequent photos taken, the length of the tripod legs are extended by one inch, and the camera was pivoted along its horizontal axis ensuring that the center of the lens was positioned at the center of the oil smear. The centerline coordinates were determined identically to previous sections. The hand-determined and MATLAB-determined leading and trailing edge point coordinates are given below in Table 7, with the error between the MATLAB and hand-drawn centerline midpoint locations given in Figure 54 and the centerline slope error shown in Figure 55. The fourth image had to be omitted from analysis due to poor image fidelity. The camera setup was improperly placed when the image was taken, resulting in the MATLAB solver being unable to properly analyze the image.

Table 7. One-dimensional oil propagation centerline pixel coordinate comparison (Y-skew)

Camera Skew	Leading Edge (X,Y)	Trailing Edge P(X,Y)	Slope	Midpoint
0° (Hand)	(253, 823)	(1882,706)	0.0718	(1068,765)
0° (MATLAB)	(273,829)	(1866,709)	0.0804	(1070,765)
1.8°	(249,846)	(1882, 750)	0.0588	(1066,798)
1.8°	(266,857)	(1861,748)	0.0683	(1064,803)
3.0°	(261, 879)	(1889,733)	0.0897	(1075,806)
3.0°	(284,867)	(1880,744)	0.0771	(1082,806)
4.6°	(-,-)	(-,-)	0.0440	
4.6°	(-,-)	(-,-)	0.0706	
6.2°	(253,867)	(1890,795)	0.0581	(1072,831)

6.2°	(274,880)	(1875,767)	0.0581	(1075,824)
7.7°	(250,879)	(1902,783)	0.0422	(1076,831)
7.7°	(270,883)	(1888,789)	0.0488	(1080,836)
9.2°	(275,855)	(1910,786)	0.0446	(1093,821)
9.2°	(295,870)	(1894,792)	0.0488	(1095,831
10.7°	(278,878)	(1914, 805)	0.0446	(1096,842)
10.7°	(296,875)	(1897,808)	0.0418	(1097,842)
11.7°	(262,835)	(1852,777)	0.0365	(1057,806)
105711.7°	(282,839)	(1834,761)	0.0503	(1058,800)
13.2°	(328,863)	(1864,793)	0.0456	(1096,828)
13.2°	(345,858)	(1847,790)	0.0453	(1096,824)

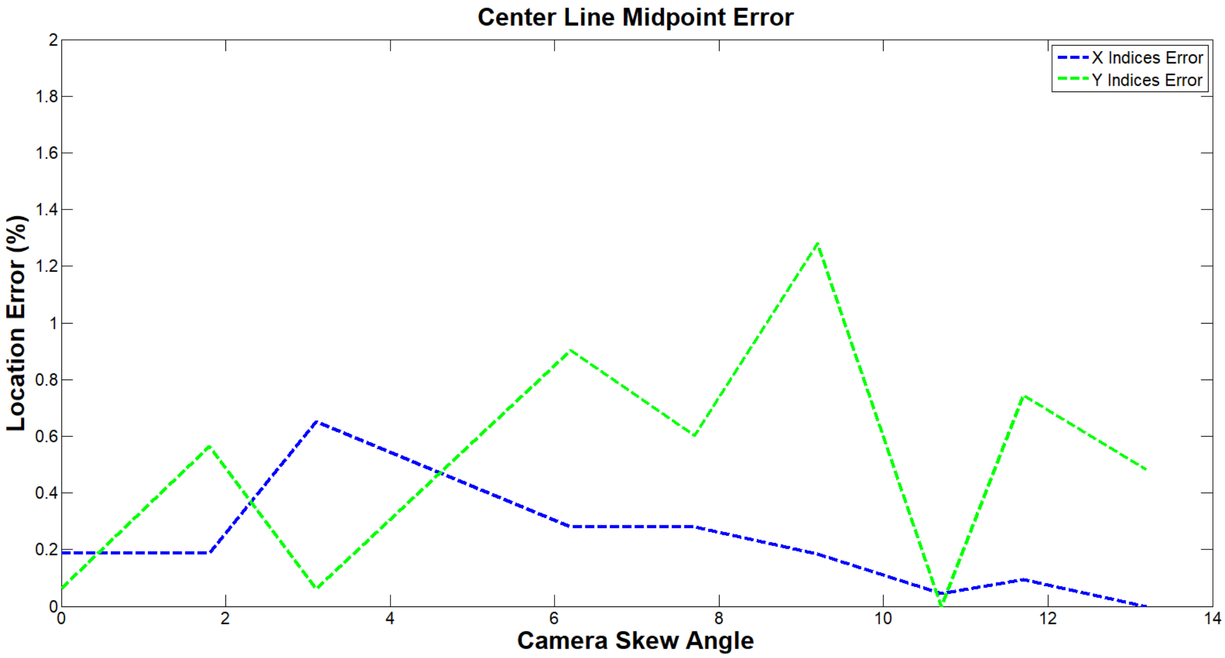


Figure 54. Centerline Midpoint pixel coordinate error, horizontal oil smear propagation, Y-dimension camera pivot angles from zero to 13.2 degrees.

The average midpoint error in this Y-skew dataset is quite low, with 0.21 percent error in the X-direction and 0.52 percent error in the Y-direction. Like the other horizontal oil smear dataset, the Y-component error is greater than the X-component error. Again, this is occurring because of the geometry of fringes. The leading and trailing edges of the fringes are predominantly vertical, meaning that any position error at that location on the fringe will be most prominent in the Y-direction. Slope error results were less than desirable, just as they were in the horizontal X-skew dataset. The average slope error is 18.10 percent, with a maximum slope error of 60.47 percent. As described in previous sections, this is a product of the horizontal nature of the centerlines, and not MATLAB solver inaccuracies. Because the centerlines in this dataset are nearly horizontal (zero slope) even minor slope changes result in large errors, despite the fact that the actual differences in slope magnitude are quite miniscule.

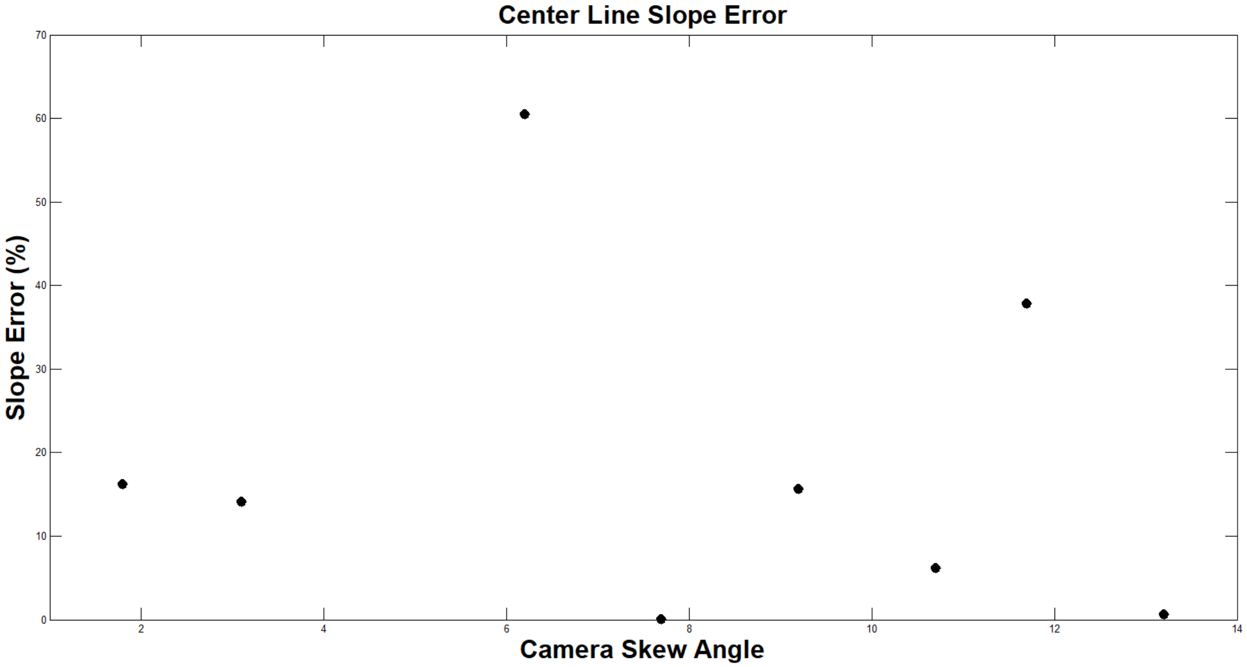


Figure 55. Centerline slope error, horizontal oil smear propagation, Y-skew camera positions ranging from zero to 13.2 degrees.

The relative uniformity of this dataset's centerlines present an opportunity to utilize Robert Ehrmann's human repeatability analysis. The average hand-drawn centerline with 2.34 degree error bounds, as well as all ten MATLAB-created centerlines are plotted below in Figure 56. Like the other human repeatability cases, error bounds are determined by applying the 2.34 degree error magnitude that Ehrmann determined in his thesis. 2.34 degree error results in a pixel location discrepancy error of 60 pixels at the right edge of the centerline's propagation. With respect to the left boundary of the average hand-drawn centerline, none of the MATLAB centerlines have boundaries more than 60 pixel away from the left edge of the average centerline. In terms of the right centerline edge, only one of the MATLAB centerlines lie outside the human repeatability error bounds. This means that nine of the ten MATLAB centerlines in this dataset are just as accurate as the hand-drawn centerlines that they are compared against.

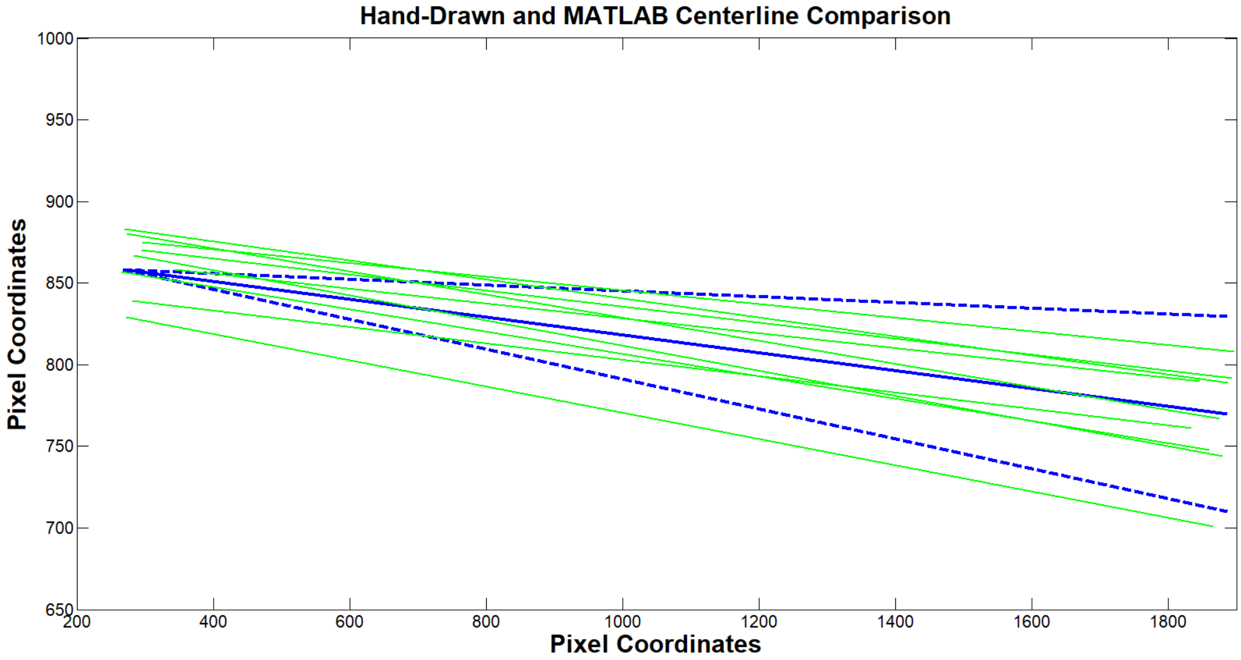


Figure 56. Average hand-drawn centerline with 2.34 degree error bands, all ten MATLAB centerlines; horizontal oil smear propagation with Y-dimension camera pivot angles from zero to 13.2 degrees.

4.5 Centerline Determination: Two-Dimensional Pixel-Length Geometries (Y-Skew)

After gauging the accuracy of the MATLAB solver with one-dimensional oil propagation and Y-axis skew, it is important that two-dimensional oil propagation be examined under the same skew conditions to verify the full functionality of the solver. The setup for this test is nearly identical to the test environment described in **Section 4.4**; the only difference being that the oil smear is rotated about its center axis to create two-dimensional oil propagation. The orientation of oil smear in this dataset is depicted below in Figure 57. This is the first image of the dataset, so there is no camera skew in this image. For the subsequent nine photographs taken, the legs of the tripod were extended as described in the previous section, resulting in a final Y-dimension skew angle of 13.2 degrees. The summary of centerline midpoint location and centerline slope is given in Table 8, with midpoint error given in Figure 58 and centerline slope error given in Figure 59.

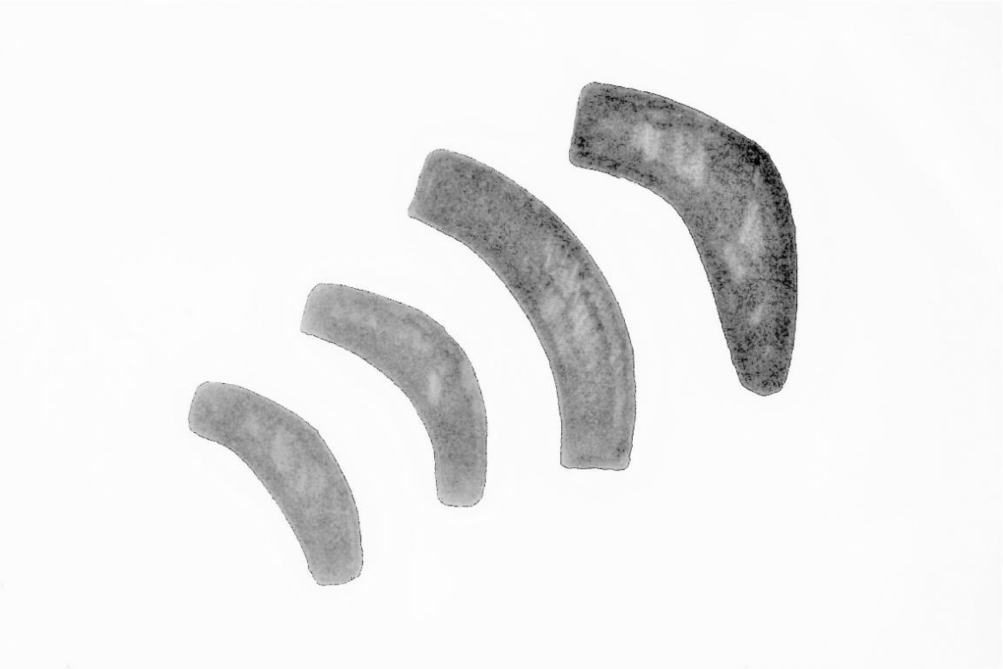


Figure 57. Representative oil smear featuring right-to-left horizontal and vertical fringe propagation, orthogonal camera position.

The error associated with this data set is quite promising, as overall it is quite low. Unsurprisingly, Y-coordinate error is larger than X-coordinate error in this dataset, as the boundaries of the leading and trailing edges are predominantly vertical. The average midpoint coordinate error in the X-direction is 0.38 percent and 0.73 percent in the Y-direction. The relatively low error in both the X and Y components reinforces the MATLAB solvers capability at determining accurate oil smear centerlines regardless of fringe orientation. Analysis of centerline slope also yielded promising results, with an average slope magnitude error of 1.55 percent. This is the third non-horizontal centerline dataset with low slope error, further reaffirming that the poor slope error associated with horizontal oil smear propagation is a product of zero slope fringe propagation, and not inaccurate MATLAB centerline placement.

Table 8. Two-dimensional oil propagation centerline pixel coordinate comparison (Y-skew)

Camera Skew	Leading Edge (X,Y)	Trailing Edge (X,Y)	Slope	Midpoint
0° (Hand)	(1829,1162)	(635,415)	0.6030	(1232,775)
0° (MATLAB)	(1822,1135)	(638,430)	0.5954	(1230,783)
1.8°	(1815,1131)	(616,422)	0.5913	(1216,777)
1.8°	(1798,1129)	(620,445)	0.5806	(1209,797)
3.0°	(1807,1190)	(625,483)	0.5981	(1216,837)
3.0°	(1793,1186)	(642,488)	0.6064	(1218,837)
4.6°	(1809,1149)	(637,442)	0.6032	(1223,796)
4.6°	(1799,1146)	(649,458)	0.5983	(1224,802)
6.2°	(1817,1198)	(599,446)	0.6174	(1208,822)
6.2°	(1804,1187)	(607,467)	0.6015	(1206,827)
7.7°	(1815,1163)	(589,434)	0.5946	(1202,799)
7.7°	(1796,1162)	(570,451)	0.5799	(1183,807)
9.2°	(1805,1172)	(592,452)	0.5936	(1199,812)
9.2°	(1783,1179)	(602,467)	0.6029	(1193,823)
10.7°	(1819,1122)	(586,422)	0.5677	(1203,772)
10.7°	(1799,1121)	(598,436)	0.5704	(1199,779)
11.7°	(1800,1151)	(569,426)	0.5890	(1185,789)
11.7°	(1784,1148)	(588,427)	0.6028	(1186,788)
13.2°	(1798,1133)	(587,434)	0.5772	(1193,784)
13.2°	(1783,1124)	(599,446)	0.5726	(1191,785)

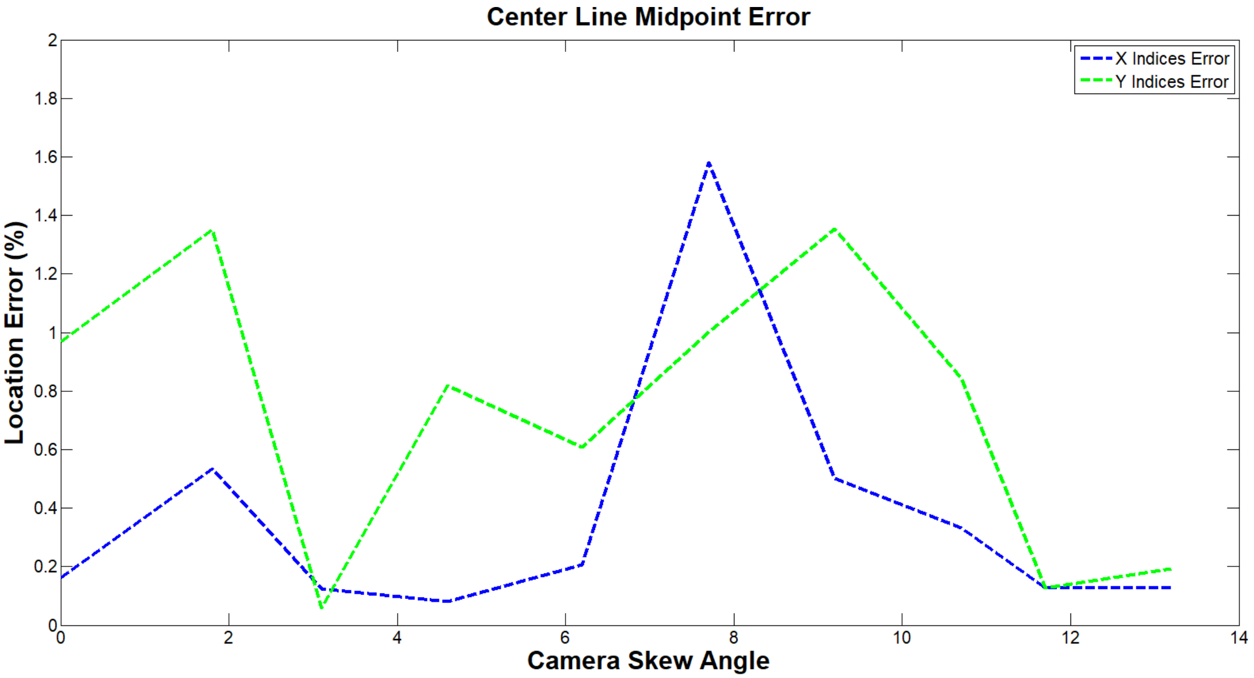


Figure 58. Centerline Midpoint pixel coordinate error, horizontal and vertical oil smear propagation, Y-dimension camera pivot angles from zero to 13.2 degrees.

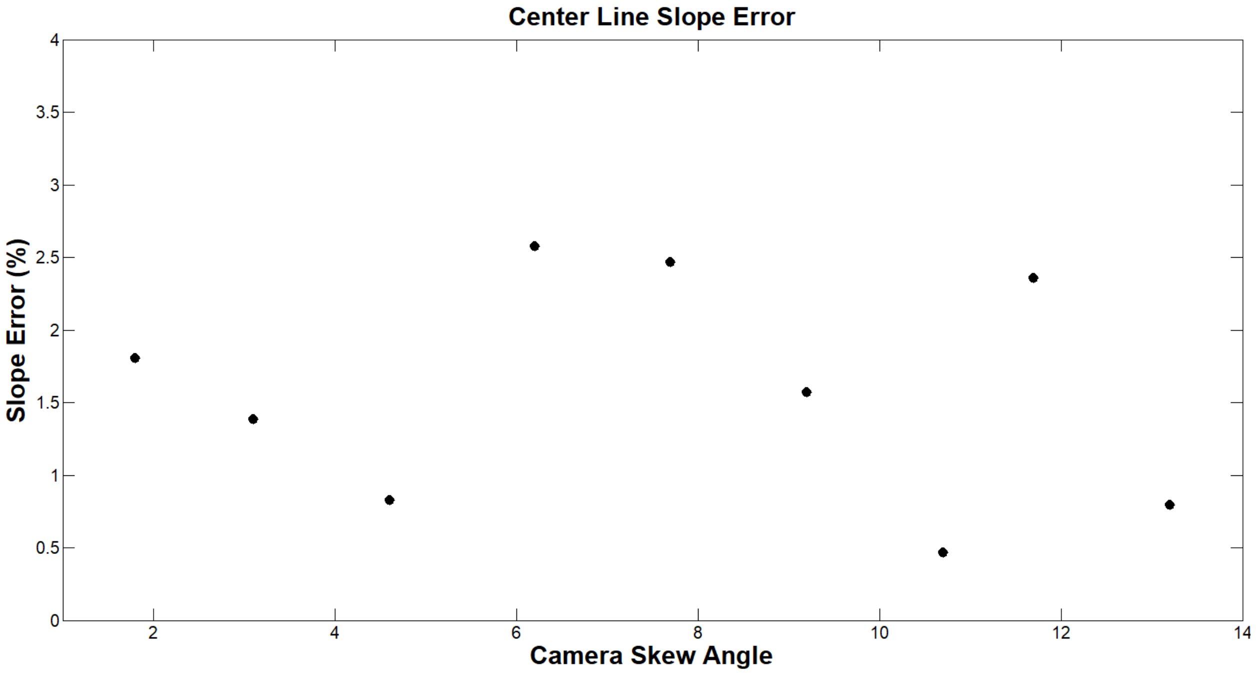


Figure 59. Centerline slope error, two-dimensional oil smear propagation, Y-skew camera positions ranging from zero to 13.2 degrees.

Human repeatability analysis was also performed on the ten centerlines obtained in this dataset. The results are shown below in Figure 60. Out of all the datasets that human repeatability analysis was performed on, this dataset shows the most variation in MATLAB centerline location. At the leading edge centerlines (right), there is a cluster of three MATLAB centerlines that are on further than 60 pixels from the average hand-drawn centerline. Furthermore, the same cluster of three centerlines are positioned outside of the 2.34 degree error band at the centerline's trailing edge (left). Furthermore, one additional MATLAB centerline skirts the edge of the trailing edge error bound. That being said, the majority of MATLAB centerlines have leading edge points within 60 pixels of the average hand-drawn centerline leading edge point in addition to being inside the 2.34 degree error bounds at the trailing edge.

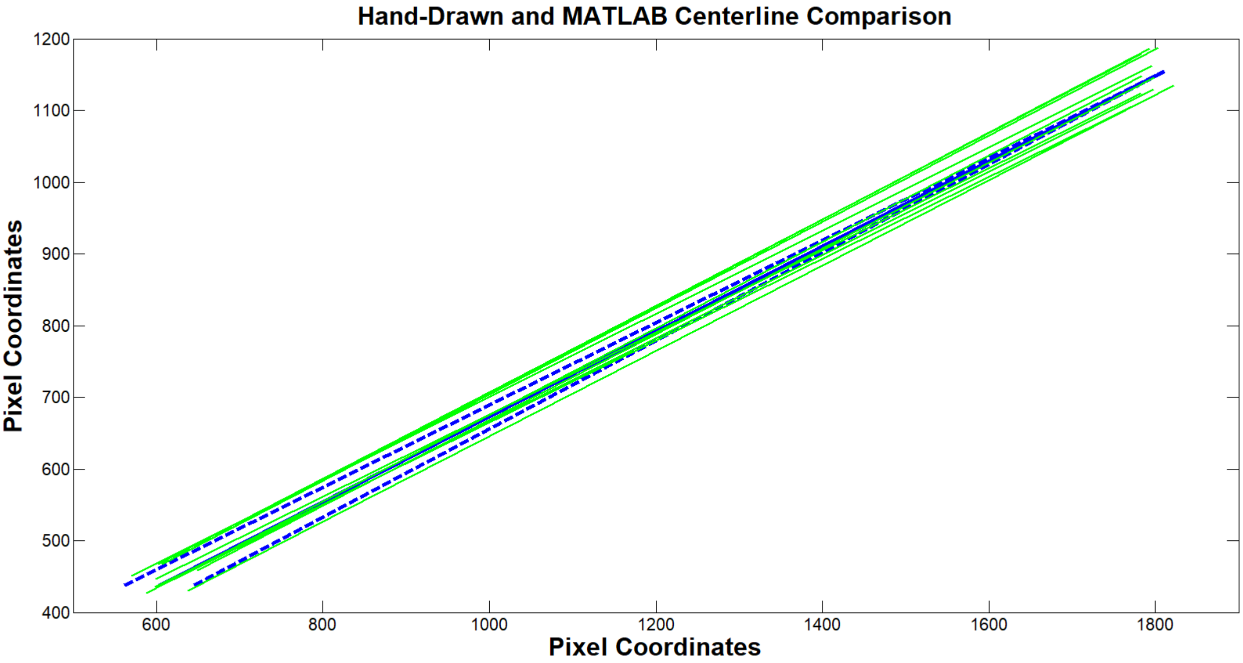


Figure 60. Average hand-drawn centerline with 2.34 degree error bands, all ten MATLAB centerlines; two-dimensional oil smear propagation with Y-dimension camera pivot angles from zero to 13.2 degrees.

5.0 FRINGE SPACING DETERMINATION

The second portion of the MATLAB solver calculates the fringe spacing of oil smears. Fringe spacing is related to the centerline determination covered in the previous section in that the leading and trailing fringe edges are used in the determination of both. While the foundations of centerline and fringe spacing determination are rooted in the analysis tools, it is important to realize that their implementation into the MATLAB solver is completely separate. In this thesis, fringe spacing is defined as the overall distance between the center point of one fringe and the center point of an adjacent fringe. Under ideal conditions, each center point is at the fringe's geometric center. Since the solver uses the centroid of a fringe as reference point, MATLAB does not produce a center point directly on a fringe's geometric center unless there is zero error in the system. More discussion on error will be provided later in this section, but it is important to note that while error can be mitigated, it cannot be fully removed from the test environment.

Determination of fringe spacing in the MATLAB solver is largely an automated process. Once a user enters all relevant camera information and selects an image for analysis the script is executed and the fringe spacing between all fringes in the photograph are produced without further user input. The MATLAB script outputs the spacing as a vector, with the first entry defining the spacing between the left-most fringe and the fringe second from the left. Subsequent spacing values in the vector define fringe spacing moving from left to right across the entire image. The MATLAB solver can analyze the fringe spacing of any flat plate oil smear that has two or more fringes. Once the pixel location of the fringe center points has been determined for each two-fringe system, the overall distance, in inches, is determined by applying Pythagorean's Theorem to the X and Y component distances between the two center points.

5.1 Fringe Spacing Determination: One-Dimensional Oil Smear Propagation (X-skew)

The first fringe spacing analysis examines the same set of photos used in **Section 4.1**. This dataset contains eight photos, all of which depict horizontal oil propagation moving in a left-to-right manner across the image. Refer to **Section 4.1** for a more detailed explanation of the experimental setup. As the image contains four fringes, the MATLAB solver returns three spacing values after analysis. On the oil smear sketch that is the subject of the images, actual (theoretical) fringe spacing is measured using a ruler with 1/16th inch accuracy. The theoretical fringe spacing for the 4-fringe smear analyzed in this portion of experimentation is given below in Table 9.

Table 9. Theoretical (hand measured) fringe spacing, four fringes

Oil Smear	First to Second (in)	Second to Third (in)	Third to Fourth (in)
Fringe Propagation	1.94	1.44	1.75

Table 10 gives fringe spacing values for all eight images in this dataset. As described in the analysis of **Section 4.1**, the initial photograph is in a no-skew orientation, with the subsequent seven images taken at skew values ranging up to 12.3 degrees. The error between MATLAB-determined fringe spacing and hand-measured fringe spacing is summarized in Figure 61 below. In this dataset, the MATLAB solver is most accurate when determining the spacing between the first and second fringes, and the least accurate when determining the spacing between the second and third fringe. One of the more interesting conclusions that can be made from the dataset is that the error increases with camera skew at a near linear rate. This comes as a bit of a surprise, as the partial differentiation analysis of skew angle determined that increasing skew changes pixel length at a non-linear rate. As the partial differentiation only looked at how skew affects pixel length and not the error associated with the pixel length, the most likely conclusion as to why the error propagation is linear is that there was measurement error every time the camera-tripod assembly was repositioned during data collection.

Table 10. MATLAB-determined fringe spacing, horizontal fringe propagation (X-skew)

Camera Skew	First to Second (in)	Second to Third (in)	Third to Fourth (in)
0°	1.92	1.48	1.74
1.8°	1.96	1.50	1.78
3.6°	1.99	1.53	1.81
5.4°	2.00	1.54	1.83

7.1°	2.04	1.57	1.86
8.9°	2.07	1.60	1.89
10.6°	2.10	1.61	1.91
12.3°	2.13	1.64	1.94

Like centerline determination, the accuracy of the fringe spacing determined by the MATLAB solver is largely a function of how precise the solver is at determining the location of each fringe's centroid. The closer to a fringe's actual geometric center the centroid is placed, the more accurate the fringe spacing values will be Centroid position inaccuracy can be attributed to two main factors: camera skew and non-uniform fringes. In this dataset, we see non-trivial differences in spacing error occurring within the same image. Since the error is different for each of the three measurements, it can be inferred that the error in this particular dataset is attributed to individual fringe irregularities in addition to image skew. The largest error is with the spacing between the second and third fringe. In Figure 62, below, the location of the four fringe centroids are overlaid on the outline of this dataset's zero skew image. The centroid of the second-to-right fringe is noticeably closer to the trailing edge of its fringe than the three other centroids are to theirs. When the MATLAB solver calculates the center point of the second-to-right fringe, its pixel location is closer its respective fringe trailing edge than the other three center points.

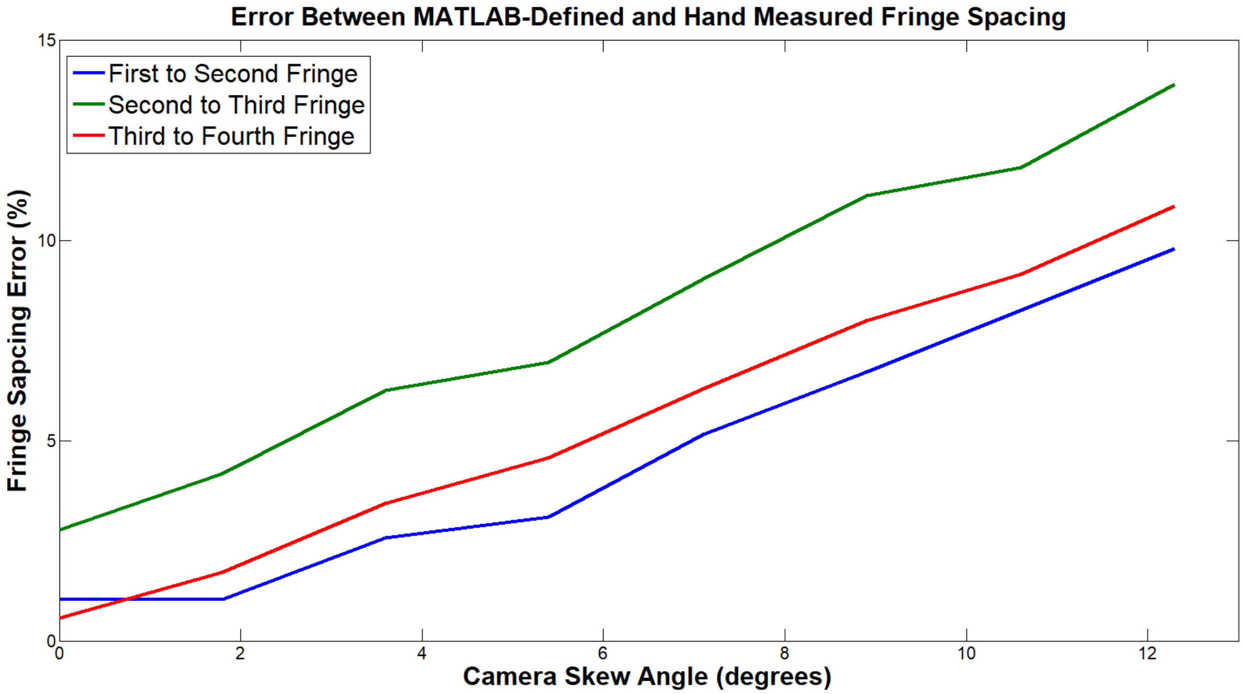


Figure 61. Fringe spacing error in horizontal oil smear propagation, X-dimension camera pivot angles from zero to 12.3 degrees.

This creates error because the hand measured fringe distances are based on ruler measurements. While recognizing geometric differences with one's eyesight is trivial, discerning exact centroid location is not. Error is inevitable, as theoretical hand measurements will not be taken using the same reference points used by the MATLAB solver. Because the second-to-right fringe is a component of determining spacing between the second and third fringes as well as the third and fourth fringes, it makes sense that the average error of these two measurements is greater than the average spacing error between the first and second fringe.

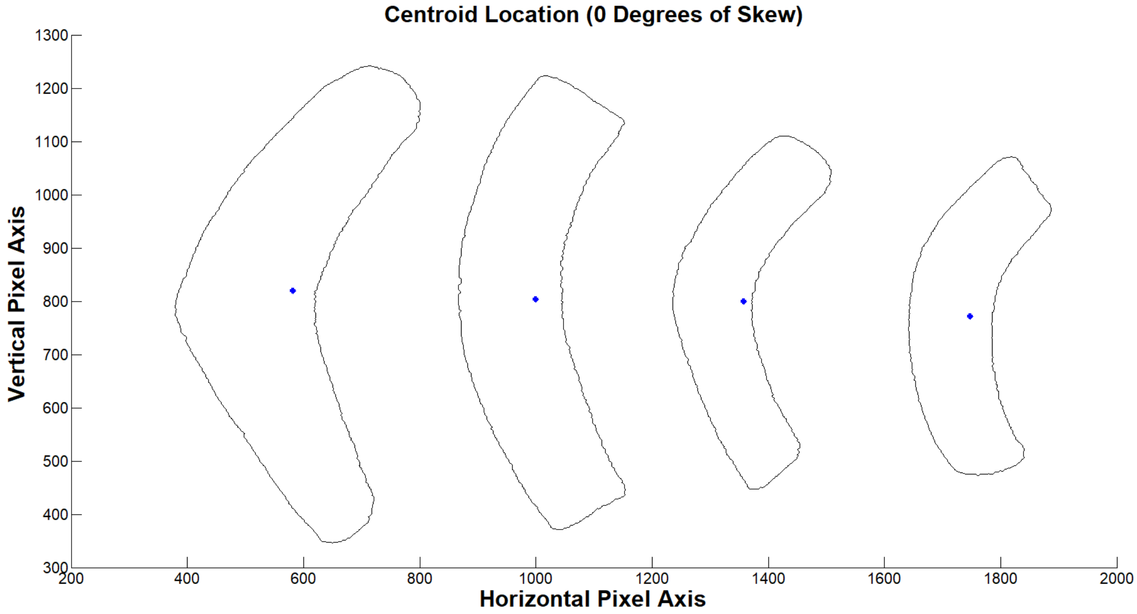


Figure 62. MATLAB-defined fringe centroid location, zeros skew image, horizontal fringe propagation.

5.2 Fringe Spacing Determination: Two-Dimensional Oil Smear Propagation (X-skew)

Spacing analysis done for two-dimensional fringe propagation uses the same dataset as the analysis performed in **Section 4.3**. In all eight images, three fringe spacing values are determined; with oil smear propagation beginning in the image's second quadrant and ending in the fourth quadrant of the image. Images are taken at camera skew values ranging from zero to 12.7 degrees with a camera height of 35 inches and a lens focal length of 70 mm. Since the oil smear used for this data set is the same (rotated) one used in **Section 5.1**, the theoretical spacing measurements are given in Table 9. The experimental fringe spacing values calculated by the MATLAB solver are given in Table 11. As in the previous section, fringe spacing is analyzed by the MATLAB solver in a left-to-right manner.

Error propagation is shown in Figure 63 and is relatively linear with respect to camera skew angle. Like the previous data set, the average spacing error between the second and third fringe is greater than the average error of the other two measurements. When compared to the horizontal fringe dataset, the spacing error in this data set is on average lower. The maximum spacing error between the first and second fringe is 8.8 percent at 12.7 degrees of skew, compared to 9.8 percent error at 12.3 degrees of skew in the horizontal propagation case. The maximum error between the second and third fringes is 11.8 percent for the two-dimensional case, and 13.9 percent in horizontal propagation dataset. This trend continues for the spacing error of the final two fringes, with the maximum error of 7.4 percent in two-dimensional propagation, and 10.9 percent in the horizontal fringe propagation data.

Table 11. MATLAB-determined fringe spacing, two-dimensional fringe propagation (X-skew)

Camera Skew	First to Second (in)	Second to Third (in)	Third to Fourth (in)
0°	1.96	1.50	1.76
1.8°	1.99	1.52	1.78
3.7°	2.02	1.54	1.81
5.5°	2.03	1.55	1.81
7.3°	2.06	1.56	1.84
9.1°	2.08	1.58	1.85
10.9°	2.09	1.60	1.86
12.7°	2.11	1.61	1.88

Like the previous dataset, the centroid of the third fringe is positioned closer to the trailing edge of its respective fringe than the centroid of the other three fringes. As explained in the previous section, this is the most likely reason for the spacing error between the second and third fringes. For the same, reason it came as a surprise that the average error in the spacing between the third and fourth fringe is less than the error of the other two measurements. The likely cause of this is the standard 1/16th inch ruler that is used to obtain the theoretical fringe spacing measurements with. Due to the size of the oil smears in question, the distance between two 1/16th inch tick marks of the ruler is up to 5 percent of the fringe spacing length. In retrospect, a higher resolution measurement device should have been utilized to minimize the possibility of error in the theoretical measurements.

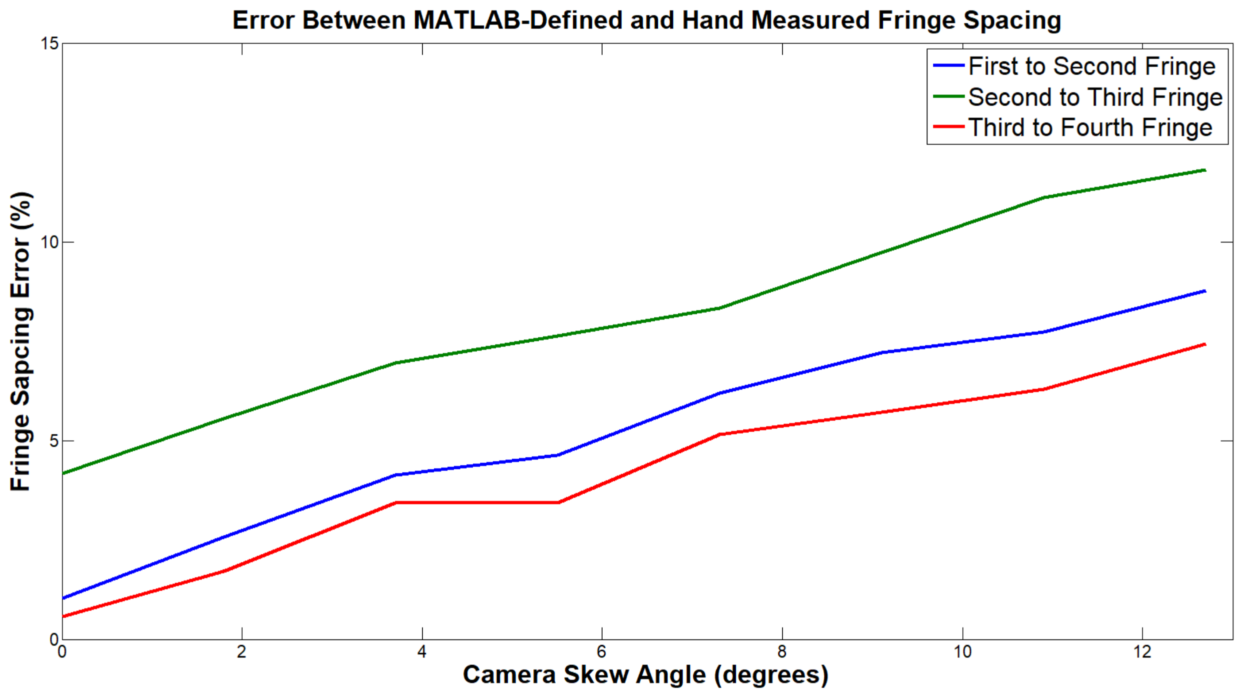


Figure 63. Fringe spacing error in horizontal and vertical oil smear propagation, X-dimension camera pivot angles from zero to 12.7 degrees.

5.3 Fringe Spacing Determination: One-Dimensional Oil Smear Propagation (Y-skew)

One-dimensional Y-skew fringe distance calculation uses the same dataset presented in **Section 4.4** of this text. The dataset contains ten images taken at camera skew angles ranging from zero to 13.2 degrees of horizontal skew and capture oil smears propagating in a horizontal (left-to-right) manner. These ten photographs capture the same hand drawn oil smear as all other photos discussed thus far. Refer theoretical hand-measured distances between this dataset's fringes are given in Table 9. The fringe spacing determined by the MATLAB solver for each of the ten images is summarized in Table 12 below, with the error trend given in Figure 64. The most noticeable difference between this dataset and the previous two fringe spacing datasets is that the error does not increase as skew angle increases. The relatively flat error curves are promising in that they suggest that the MATLAB solver can provide precise results. While precise, the results leave room for the solver to be more accurate. These inaccuracies are likely due to focal length uncertainty. The lens used in this thesis has a variable focal length that cannot be locked into position. In addition, the tick marks on the lens that signify what focal length the lens is at are only provided in 30 mm increments. This means that during experimentation not only is there a possibility for the focal length to shift inadvertently, but also that a user can never be completely certain of the lens configuration. Since a five millimeter lengthening or shortening of focal length changes the perceived fringe spacing by up to ten percent, one cannot be totally confident in the accuracy of the reported focal length. Using a lens of higher quality would provide greater confidence in experimentation, and further reduce error.

Table 12. MATLAB-determined fringe spacing, horizontal fringe propagation (Y-skew)

Camera Skew	First to Second (in)	Second to Third (in)	Third to Fourth (in)
0°	1.98	1.56	1.70
1.8°	1.99	1.56	1.68
3.0°	1.99	1.56	1.71
4.6°	2.00	1.56	1.71
6.2°	2.00	1.57	1.71
7.7°	2.01	1.58	1.73
9.2°	1.99	1.57	1.71
10.7°	1.99	1.57	1.71
11.7°	2.05	1.62	1.76
13.2°	1.99	1.56	1.71

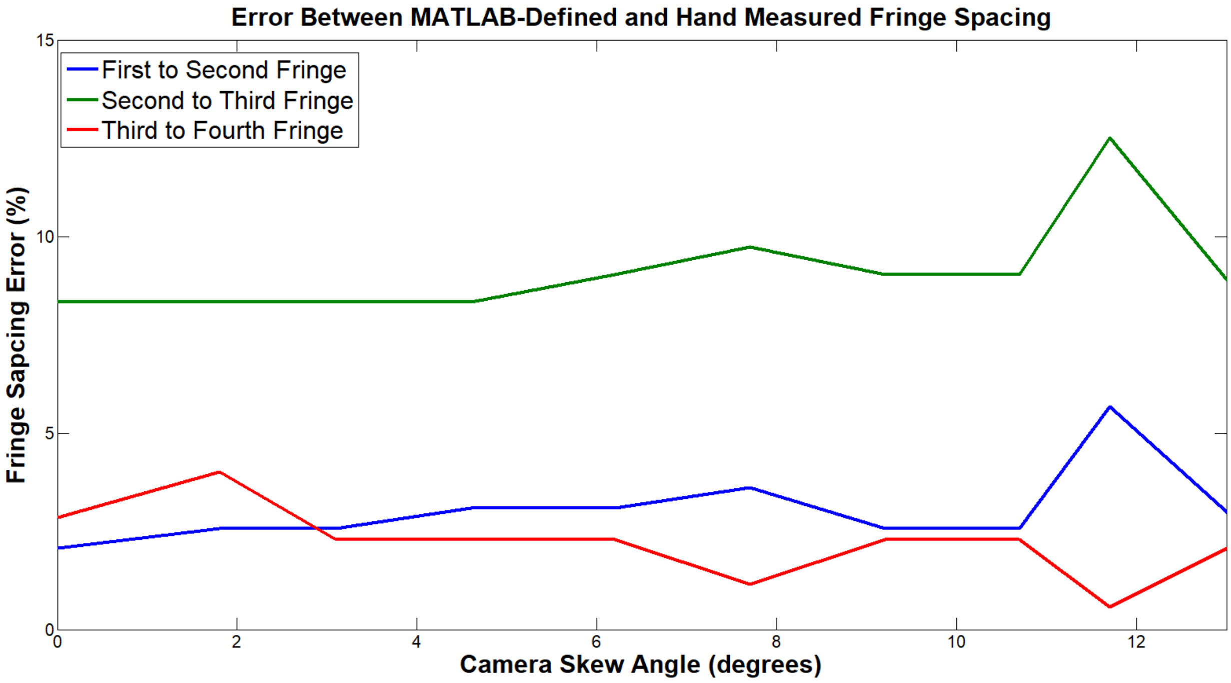


Figure 64. Fringe spacing error in horizontal oil smear propagation, Y-dimension camera pivot angles from zero to 13.2 degrees.

The relatively constant error is attributed to the accuracy of the camera set up used when capturing vertical (Y) skew. During image collection, the camera movement is restricted to its vertical plane. Paper is placed underneath the tripod assembly and the location of the vertical tripod support bar is marked. Each time the tripod legs are extended by one inch, repositioning the vertical support bar over the paper marking ensures that the camera lies on the same vertical plane for each successive photograph. The repositioning of the camera during collection of X-skew images is inherently more prone to error because the entire camera-tripod assembly is moved each time a new image is taken. In addition, since the one-inch translations are hand measured, error in the placement is also possible. This movement inaccuracy is the reason why the X-skew data shows near linear error increase. For example, if the camera movement between the any two photographs is 1.2 inches instead of 1.125 inches, the position error will be carried throughout the data set. If measurement error is present in subsequent camera positions, that error will propagate through the remainder of the dataset as well.

5.4 Fringe Spacing Determination: Two-Dimensional Oil Smear Propagation (Y-skew)

Finally, spacing analysis was completed on images of representative hand-drawn oil smears propagating in a two-dimensional orientation. Like the one-dimensional Y-skew analysis, this dataset contains ten images taken at skew angles up to 13.2 degrees. The ten images analyzed in this section are the same ones used for analysis in **Section 4.5**. The outline of the zero-skew oil smear is given in Figure 65 and includes the centroid location of each fringe. The fringe spacing calculated by the MATLAB solver is summarized in Table 10, with the error between the theoretical and experimental spacing given in Figure 66. The theoretical fringe spacing is given in Table 9.

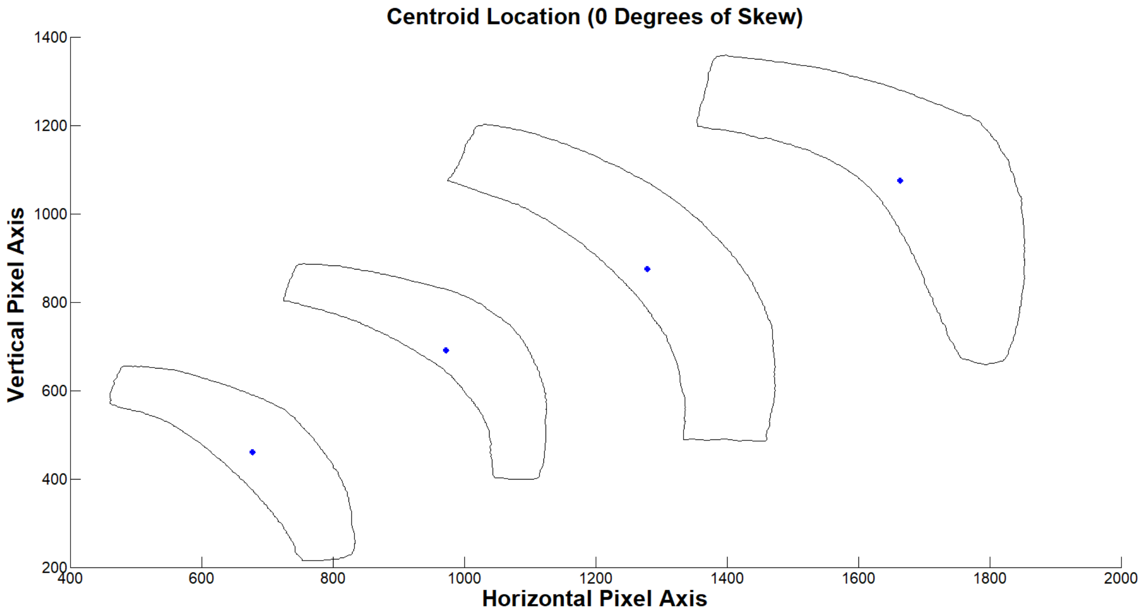


Figure 65. MATLAB-defined fringe centroid location, zeros skew image, horizontal and vertical fringe propagation.

Because the oil smear propagates in a right-to-left manner in each of the ten images and the MATLAB solver calculates fringe spacing in a left-to-right manner the naming conventions of this this dataset differ from the rest. ‘First to Second’ fringe spacing listed in Table 13 and Figure 66 refers to the final two fringes in the oil smear’s propagation, while ‘Third to Fourth’ spacing refers to the leading two fringes of the smear.

Table 13. MATLAB-determined fringe spacing, two-dimensional fringe propagation (Y-skew)

Camera Skew	First to Second (in)	Second to Third (in)	Third to Fourth (in)
0°	1.66	1.52	1.93
1.8°	1.64	1.50	1.91
3.0°	1.63	1.49	1.89
4.6°	1.63	1.48	1.88
6.2°	1.70	1.55	1.96
7.7°	1.70	1.55	1.96
9.2°	1.69	1.53	1.93
10.7°	1.71	1.54	1.94
11.7°	1.73	1.56	1.95
13.2°	1.70	1.53	1.92

Like the Y-skew horizontal propagation case, this data suggests that error does not increase linearly with camera skew. Though the error is not increasing at a steady rate, it is somewhat more irregular than the one-dimensional Y-skew propagation case. Another interesting data trend is that the error in two of the three spacing measurements decreases between four and six degrees of skew, while the error of the remaining measurement increases. Also worth noting is that the spacing error between the second and third fringe is on average higher than the other two measurements, as it is for all of the other spacing datasets in this section.

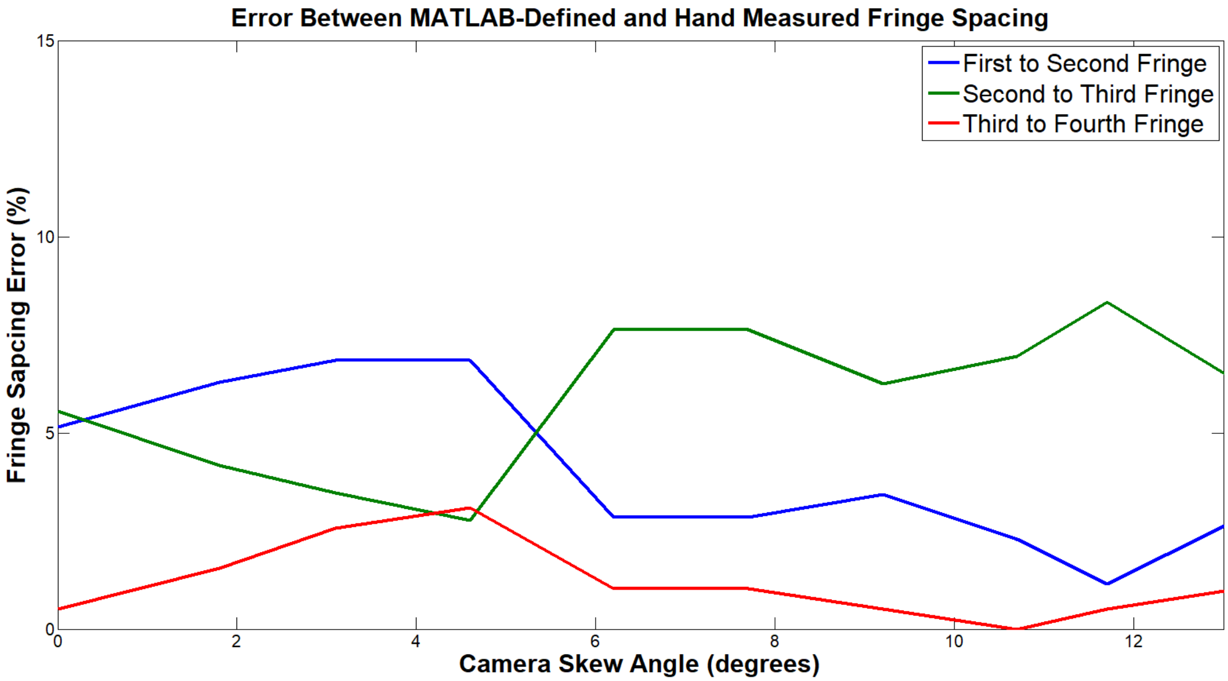


Figure 66. Fringe spacing error in horizontal oil smear propagation, Y-dimension camera pivot angles from zero to 13.2 degrees.

The reason that the two outboard spacing errors decrease between four and six degrees of skew while middle spacing error increases is a result of the centroid location of the second fringe being closer to the fringe trailing edge than it is in the other three fringes. For the first-to-second and third-to-fourth fringe spacing, the MATLAB solver undercuts the theoretical spacing values, while the second-to-third fringe spacing is overestimated by the MATLAB solver. The overestimation in the experimental data occurs because the theoretical hand measurements do not take the aft centroid location into account when recording to the spacing of the two inner fringes. The underestimation of fringe spacing occurs for the same reason, as the MATLAB solver determines the centroids to be closer together than the estimation performed while measuring fringe spacing by hand. During the setup of the fifth image, the lens focal length was inadvertently decreased. Despite efforts to reposition the lens to the correct length, it is clear from the data that focal length was not successfully repositioned to 70 mm. The decrease in focal length results in increased pixel length, and increased spacing measurements. Because the first-to-second and third-to-fourth fringe spacing is underestimated by the solver, an increase in the spacing lowered the error of those two measurements. Error increased in the second-to-third data because the MATLAB solver had calculated the fringe spacing to be greater than the theoretical measurements even before the decrease in focal length.

5.5 Fringe Spacing Determination: Disproving Solver Error

Because of the linear error increase in both X-skew datasets, the data gives the illusion of code error within the X-skew portion of the MATLAB solver. As was discussed in the previous sections, the X-skew error propagation is a result of the position error of the tripod assembly. To give further confidence of this, the images in both X-skew datasets were rotated 90 degrees so that the Y-skew portion of the MATLAB solver can be used to analyze the X-skew data. If both methods of image analysis result in similar fringe spacing, the X-skew linear error propagation is indeed a product of the experimental setup, and not MATLAB solver.

The results of rotating the horizontal fringe propagation dataset by 90 degrees are given in Figure 64 below. The figure depicts fringe spacing as a function of camera skew angle, with native (X-skew) image fringe spacing given in solid lines, and the rotated image (Y-skew) fringe spacing given in the dashed lines.

Shown in Figure 67, the difference in fringe spacing between the native and rotated images is quite small, especially for the second and third spacing measurements. While the native and rotated fringe spacing between the first and second fringes begins to diverge at higher angles of camera skew, it is important to note the small scale of this figure's Y-axis. The maximum fringe spacing differential between the native and rotated images is only 5.16 percent. Taking all three spacing measurements into account, the average error between the native and rotated images is only 1.00 percent.

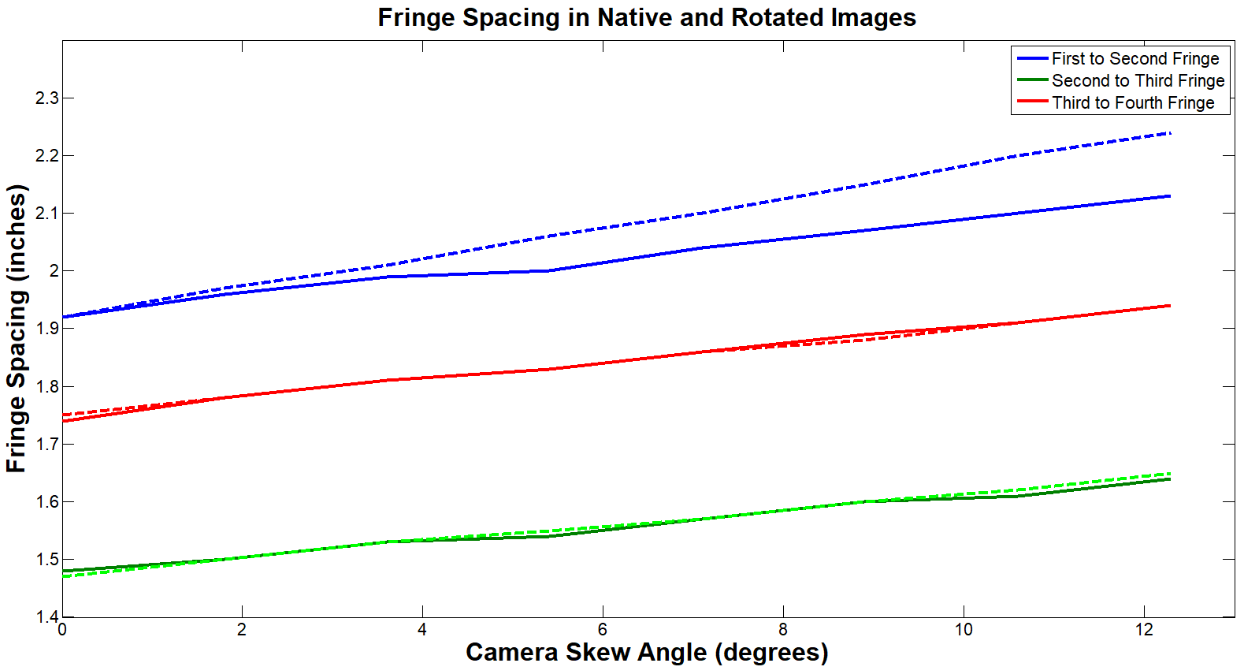


Figure 67. Fringe spacing error in native X-skew images (solid lines) and rotated X-skew images; horizontal oil smear propagation, Y-dimension camera pivot angles from zero to 12.3 degrees.

Identical analysis is also presented for the two-dimensional X-skew data. Unfortunately, when rotating the native images pixel artefacts emerged that affected the fringe spacing determining of the two interior fringes. Successful fringe measurements were obtained between the first and second fringes and between the third and fourth fringes, however. The successful fringe spacing measurements are given in Figure 68, below. Like the previous dataset, fringe spacing discrepancy between the X-skew and rotated pseudo Y-skew case is minimal, with an average error between the native and rotated images of only 0.88 percent. With an average error of less than one percent between the two pseudo Y-skew data sets and their respective X-skew native datasets, it can be concluded that the X-skew portion of the MATLAB solver is not less accurate than the Y-skew portion of the solver, and that the X-skew error propagation shown in previous sections is a result of camera placement error during data collection.

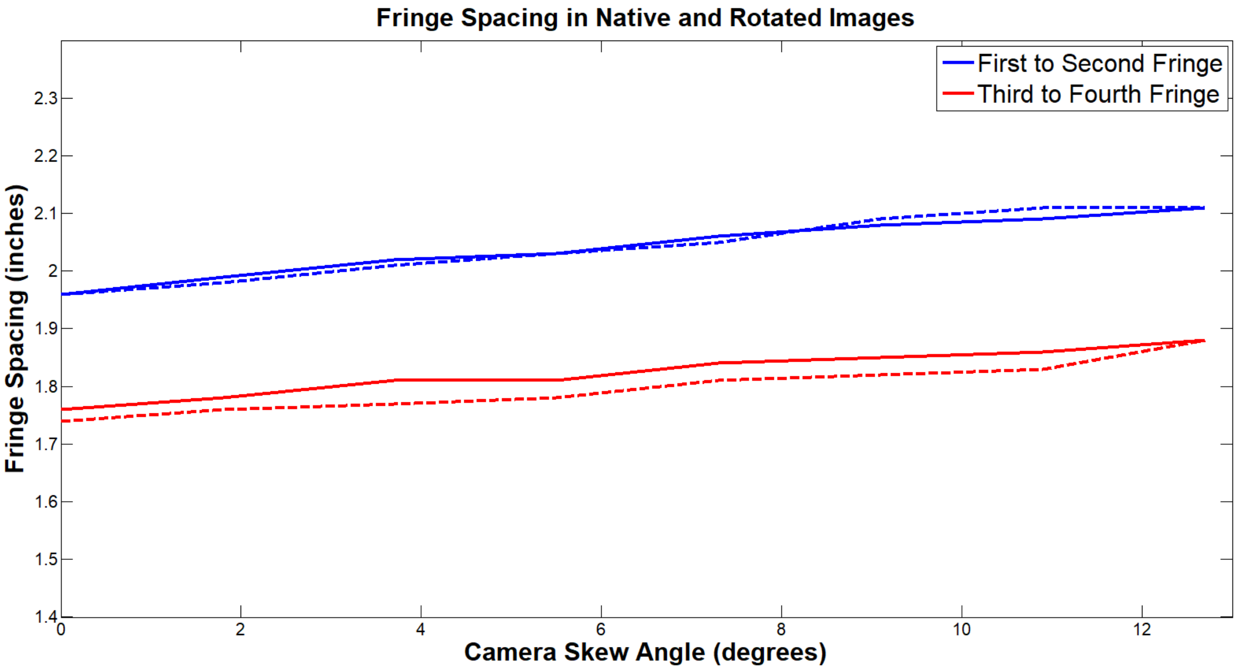


Figure 68. Fringe spacing error in native X-skew images (solid lines) and rotated X-skew images; two-dimensional oil smear propagation, Y-dimension camera pivot angles from zero to 12.7 degrees.

6.0 CONCLUSION

The value of the MATLAB solver can be determined by answering the following three questions: does the solver provide accurate fringe centerline determination, provide accurate fringe spacing measurements and third, does the MATLAB solver's automation preferable to CXWIN5G operation? Let us begin by addressing the issue of solver automation and how it compares to CXWIN5G. For the automation implemented by the MATLAB solver to be successful, it must save users time compared to CXWIN5G as well as improve a user's experience.

In order to obtain fringe spacing measurements over an identical flat surface, CXWIN5G requires at least nine camera geometry measurements to be made, while the MATLAB solver requires only three. In the MATLAB solver, these three measurements are the camera height, camera skew angle, and lens focal length. In CXWIN5G, in addition to measuring the camera's height, skew and, lens focal length; one must measure the position coordinates of all six fiduciary markings. In addition, six fiduciary markings is the minimum number of reference points that should be used, as Zilliac suggests that at least 12 fiduciary markings be used to ensure the highest level of accuracy[17]. Equally relevant is that the MATLAB solver does not require the level of model surface modifications that is required when performing oil smear analysis with CXWIN5G. As mentioned in the introduction of this text, CXWIN5G fiduciary markings can be stickers or etchings. Stickers are easy to place, but result in greater boundary layer disruption than etchings do. Because of this, etchings are preferred, but require expert application using expensive machinery to ensure that the etchings are just deep enough to be visible but not too deep such that the boundary layer is disturbed.

In the centerline analysis performed in **Section 4** of this text it is presented that the MATLAB-defined centerlines are just as accurate as ones drawn by hand. The MATLAB solver removes the necessity for a user to manually draw oil smear centerlines. The tradeoff to this level of automation is that the MATLAB solver can only process a single oil smear in each photograph, while CXWIN5G can analyze multiple oil smears captured in a single image. Whether or not automating centerline determination saves a user time is dependent on the test scenario, as a user will either be taking multiple photographs using the MATLAB solver or spend their time drawing centerlines with CXWIN5G. The MATLAB solver is highly suited for the analysis of a handful of oil smears or if oil smears are situated at multiple locations that cannot be captured in a single image, while CXWIN5G is probably better suited for the analysis of highly clustered oil smears that can be captured in a single image.

The second goal of this thesis is to determine fringe spacing by calculating the dimensions of a test surface captured in each pixel. Pixel determination is what allows the MATLAB solver to operate without the use of fiduciary markings. In the CXWIN4G user manual published by NASA AMES, Zilliac claims accuracy of ‘plus or minus 5 percent[17], so five percent accuracy should be the standard by which the MATLAB solver is compared. The fringe spacing analysis presented in **Section 5** of this text gives predominantly positive results, with the average error of MATLAB-defined fringe spacing being 6.1 percent in the X-skew data, and 4.3% in the Y-skew data. As covered in **Section 5.5** of this text, the linear error propagation in both X-skew datasets were a result of camera position. In response to this, it is more appropriate to base the fringe spacing error of the two Y-skew datasets. The fact that the X-skew datasets are affected by improper camera position is not a concern when it comes to real world analysis, as the error propagation was a result of moving the camera between photographs. In during actual data collection, only one photo is required to determine the spacing characteristics of any one smear, so the compounding of error as a result of multiple different camera locations would never occur.

While the MATLAB solver is limited to analyzing the propagation of oil smears across a flat surface and is unable to process more than one oil smear in a single image, its relevance is undeniable. CXWIN5G promises 5 percent error while affording little automation to the user. As shown in the results presented in the preceding sections of this text, the MATLAB solver is capable of producing centerline and fringe spacing results every bit as accurate as CXWIN5G while affording a much higher level of automation to the user.

REFERENCES

[1]Atwood, Jeff. "Widescreen and FOV." *Coding Horror Programming and Human Factors*, 23 Aug. 2007, blog.codinghorror.com/widescreen-and-fov/. Accessed 17 Aug. 2017.

[2]Cappadocia Gaussian Blur. 23 July 2015. Wikipedia. Wikipedia Commons. Wikipedia. Web.

[3]Ehrmann, Robert Schaefer. Development Of Measurement Methods For Application To A Wind Tunnel Test Of An Advanced Transport Model. California Polytechnic State University, San Luis Obispo. August 2010.

[4]Mark S. Nixon and Alberto S. Aguado. Feature Extraction and Image Processing. Academic Press, 2008, p. 88. –Gaussian blur

[5]Mathworks.com. (2017). *Adjust image intensity values or colormap - MATLAB imadjust - MathWorks United Kingdom*. [online] Available at: https://www.mathworks.com/help/images/ref/imadjust.html [Accessed 28 Aug. 2017].

[6]Mathworks.com. (2017). *Convert RGB image or colormap to grayscale - MATLAB rgb2gray - MathWorks United Kingdom*. [online] Available at: https://www.mathworks.com/help/matlab/ref/rgb2gray.html [Accessed 28 Aug. 2017].

[7]Mathworks.com. (2017). *Create predefined 2-D filter - MATLAB fspecial - MathWorks United Kingdom*. [online] Available at: https://www.mathworks.com/help/images/ref/fspecial.html [Accessed 28 Aug. 2017].

[8]Mathworks.com. (2017). *Dilate image - MATLAB imdilate - MathWorks United Kingdom*. [online] Available at: https://www.mathworks.com/help/images/ref/imdilate.html [Accessed 28 Aug. 2017].

[9]Mathworks.com. (2017). *Erode image - MATLAB imerode - MathWorks United Kingdom*. [online] Available at: https://www.mathworks.com/help/images/ref/imerode.html [Accessed 28 Aug. 2017].

[10]Mathworks.com. (2017). *Fill image regions and holes - MATLAB imfill - MathWorks United Kingdom*. [online] Available at: https://www.mathworks.com/help/images/ref/imfill.html [Accessed 28 Aug. 2017].

[11]Mathworks.com. (2017). *Morphological structuring element - MATLAB - MathWorks United Kingdom*. [online] Available at: https://www.mathworks.com/help/images/ref/strel-class.html [Accessed 28 Aug. 2017].

[12]Mathworks.com. (2017). *N-D filtering of multidimensional images - MATLAB imfilter - MathWorks United Kingdom*. [online] Available at: https://www.mathworks.com/help/images/ref/imfilter.html [Accessed 28 Aug. 2017].

[13]Monson, Daryl J, et al. "Boundary-Layer Transition and Global Skin Friction Measurement with an Oil-Fringe Imaging Technique." SAE Aerotech, 27 Sept. 1993, ntrs.nasa.gov/search.jsp?R=19950023803.

[14]S, N. "Curve Intersections." MATLAB Central File Exchange, 3.0, MathWorks, 21 Sept. 2010, www.mathworks.com/matlabcentral/fileexchange/22441-curve-intersections. Copyright (c) 2009, NS All rights reserved.

[15]Sobel, Irwin. History and Definition of the So-called "Sobel Operator", More Appropriately Named the Sobel-Feldman Operator. N.p.: ReasearchGate, June 14, 2014. Web.

[16]Zilliac, Gregory G. "Further Developments of the Fringe-Imaging Skin Friction Technique." Ames Research Center, Dec. 1996, ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19970010467.pdf.

[17]Zilliac, Gregory G. The Fringe-Imaging Skin Friction Technique PC Application User's Manual. Moffett Field: Ames Research Center, 1999. Print.

APPENDIX

CAL POLY LOW SPEED WIND TUNNEL TEST

In addition to performing centerline and fringe spacing analysis on representative hand-drawn oil smears, the MATLAB solver was used to analyze oil smears obtained by wind tunnel testing in the Cal Poly three-foot by four-foot low speed wind tunnel. Testing was completed on the plywood floor of the low speed tunnel. In order to improve the light reflection characteristics of the plywood base, a sheet of Top Flight MonoKote Mylar was applied to the plywood tunnel floor. In addition to improving the light reflection characteristics of the test surface, the MonoKote layer decreased surface friction and masked many of the surface imperfections present in the plywood. 200 cSt. oil was used during experimentation, chosen because of its relatively high vapor pressure.

Wind tunnel testing was largely an exercise in trial and error. For the initial tunnel test runs, lines of oil were used for analysis. Oil propagation lasted between 15 and 18 minutes, with nominal tunnel wind speeds ~ 25 m/s. Initial results were suboptimal, mainly due to inconsistent oil propagation and poor fringe quality. Different portions of the oil line propagated at different rates, resulting in jagged, uneven fringes. In addition to jagged fringes, the fringes were not well defined. The jagged fringes were a result of uneven oil thickness along the oil line. The poor fringe definition and small fringe spacing is a product of the low tunnel speed and test duration. In order to resolve the fringe spacing issue, two additional tunnel runs were done at a tunnel wind speed of ~30 m/s and ~35 m/s. The faster tunnel speeds did increase fringe visibility and spacing in certain portions of the oil smear, but the jagged, uneven edges of the oil fringes remained. Three more oil line tunnel tests were performed at ~35 m/s. In all three tests, uneven, jagged fringe propagation resulted from uneven oil application.

After obtaining undesirable results using lines of oil, testing efforts shifted to the use of oil drops in the hope that obtaining uniform oil thickness would be easier using drops. In subsequent testing, fringe uniformity was increased, but fringe resolution was quite poor. In order to increase fringe resolution, the amount of oil used in each run was dramatically increased. The original scope of the wind tunnel test encompassed taking images of oil propagation from five camera locations. Once image collection began, lens focal length limitations and difficulty in obtaining adequate lighting restricted image collection to a single camera location. The camera location relative to the oil smear and light source is shown in Figure A1. Camera height is 14 inches and camera skew angle is 18.9 degrees, with a lens focal length of 135 millimeters. The light source used for oil illumination emits quasi-monochromatic green light. To compliment the green light, a green filter was attached to the end of the lens that reduced outside light interference.

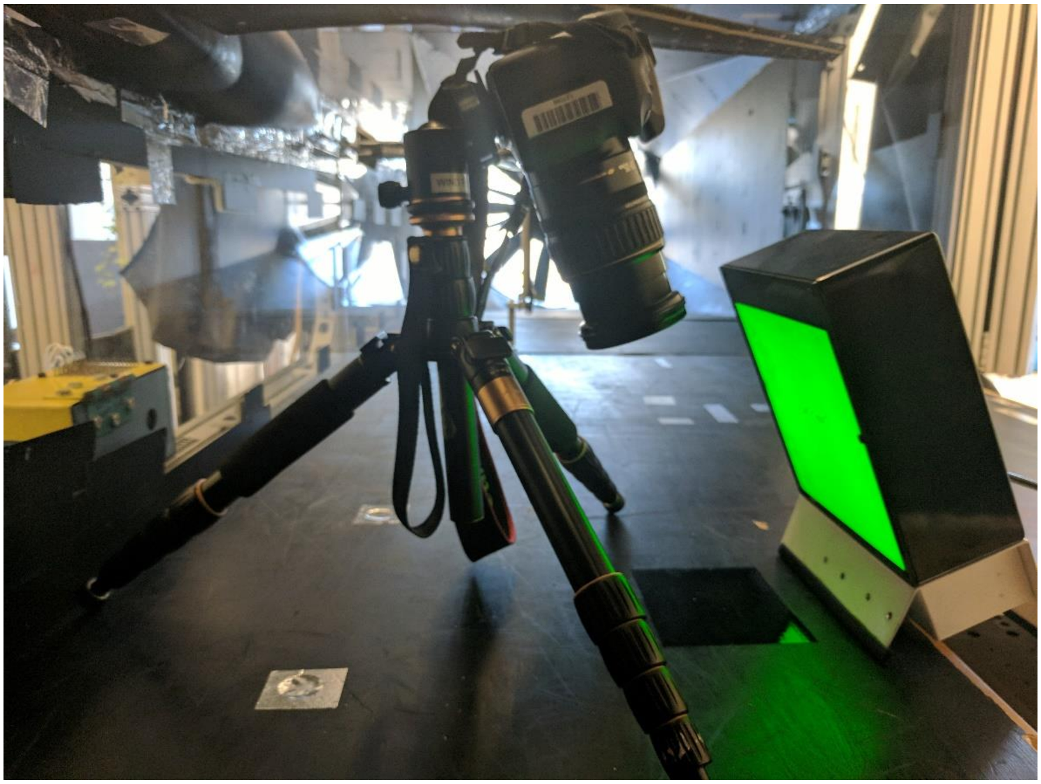


Figure A1. Camera set up geometry for testing in the Cal Poly 3’ x 4’ wind tunnel; 14’ camera height, 18.9 degree camera skew angle, 135 mm focal length, monochromatic green light source.

The main reason for limiting camera setup to a single location is the small fringe spacing that resulted from testing. During wind tunnel testing nominal fringe spacing was 1 to 2 millimeters, even after the oil drop had been exposed to ~35 m/s flow for 18 minutes. The small spacing required the camera lens has to be as close to the oil smear as possible, resulting in the only feasible camera location being the one that positioned the lens as close to the oil smear as possible. In even the best oil smear results obtained, the first fringe is well-defined, but subsequent fringes are not, with oil propagation appearing as a grey blot after the third fringe.

Over the course of two days, a total of 13 tunnel tests were performed. The majority of testing resulted in poor fringe propagation, poor photo quality, or a combination of the two. The final two tests produced the best overall fringe propagation and resultant images. These images featured oil smears propagating in a horizontal left-to-right orientation. Because the two final tests produced images that depict similar fringe propagation, only one image is used for analysis. This image is shown in Figure A2 and includes a centimeter ruler so that theoretical fringe spacing can be accurately determined. This image was taken after subjecting a drop of oil to 35.14 m/s flow for 18 minutes in the Cal Poly low speed wind tunnel. Like all other test runs, this run was performed by placing oil on a horizontal sheet of MonoKote.

In order for the MATLAB solver to analyze wind tunnel oil smears, the binary image creation portion of the MATLAB solver had to be uniquely tailored to process these types of images. In the hand drawn representative oil smears, fringes are well defined and drawn on a perfectly white background that provides excellent contrast against the black fringes. Oil smear images obtained via wind tunnel test are uniquely challenging in that the fringes are not well defined nor do they have a white background. In their native format, the resultant binary image define the entire oil smear as one body because fringe definition is not matured enough for MATLAB’s image processing tools to define their edges. In order for the MATLAB solver to produce a useable image, image editing operations specific to the wind tunnel images must be applied.

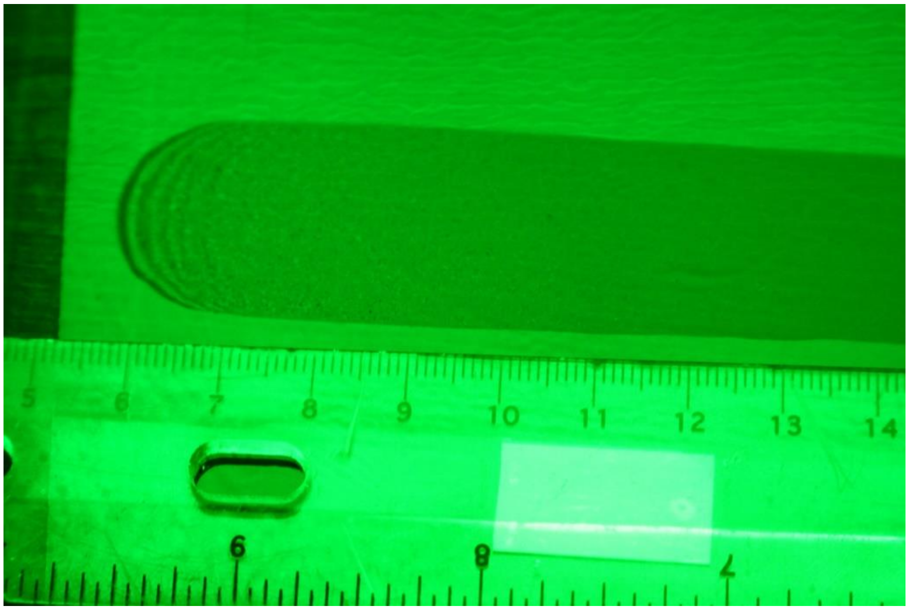


Figure A2. Resultant oil smear from testing in the Cal Poly 3’ x 4’ wind tunnel, oil exposed to 35.14 m/s flow for 18 minutes.

The first correction implemented is the transfer from a red-green-blue color gradient to a grayscale image. Once completed, the contrast of the entire image is increased until area surrounding the oil smear appears white, the fringes appear dark grey, and the regions between the fringes appear light grey. Next, histogram correction is applied to further isolate the fringes. MATLAB defines the color of every pixel in a greyscale image on a scale ranging from zero to one, with zero defined as black and one defined as white. Using the MATLAB function IMADJUST[5] users are able to produce a new image that only displays pixels that are within user-defined histogram values. Through trial and error, it was determined that only plotting pixels with histogram values between 0.20 and 0.69 resulted in the best possible fringe isolation. Because the fringe edges are not particularly well defined, creating a binary image with nothing else but the histogram correction results in incomplete, broken fringe boundaries. In order to produce defined fringe edges, the boundaries of binary image are dilated and refined with the use of morphological structuring elements (MATLAB function STREL[11]). For the dilation, a square structure is used with a side distance of two pixels. This function works by extending a square area of influence around every edge pixel. The edge pixels are in the center of their respective influence square, with the walls of the square two pixels away from the edge pixel. The function increases the edge resolution by decreasing the greyscale value of the lighter pixels within each square. After the pixels surrounding the edge pixels become darker, binary image generation is able to incorporate them into its edge detection, producing well-defined continuous edges.

Due to small fringe spacing at their upper and lower edges, the horizontal components of the square morphological structures bridge the gap between the fringes and connect them into a single defined edge. In order to separate the fringe edges, the image is eroded with the use of a diamond morphological structuring element with a side distance of one pixel. The shape of the diamond element is a square element that has been rotated 45 degrees. Diamond elements are advantageous because the slope of their walls approximate the slope of the fringe edges better than a square structure, allowing them to disrupt the edge areas that connect the fringes without having much effect on the actual fringe boundaries. The resultant image used obtained from modifying Figure 52 is shown below in Figure 18.

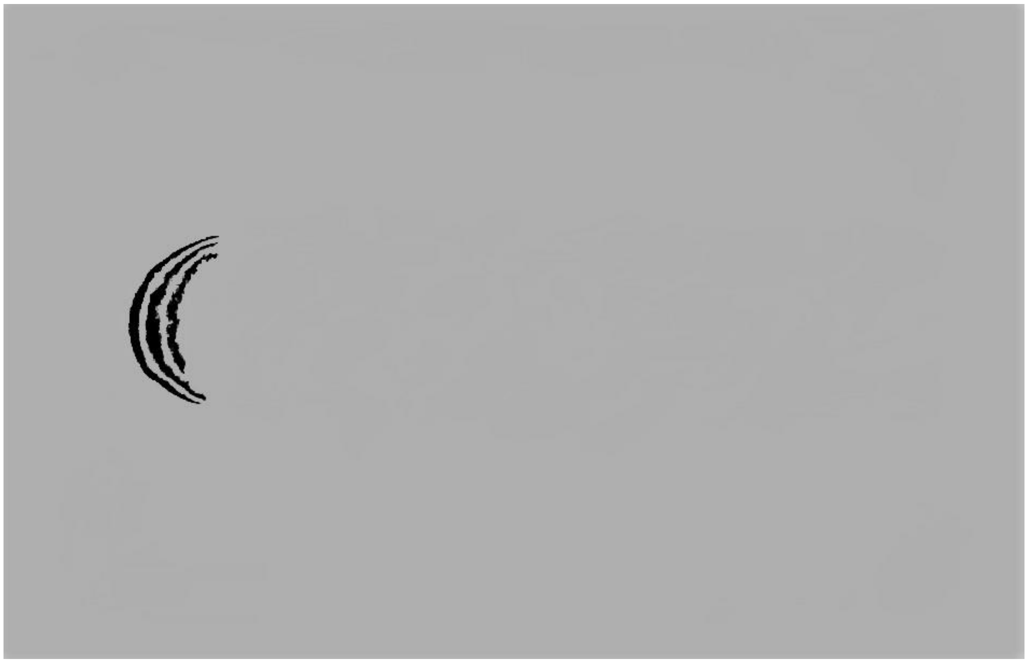


Figure A3. Final binary image results after using MATLAB histogram correction and morphological structuring element refinement to isolate the leading three fringes.

The midpoint coordinates and slope of the MATLAB centerline and hand-drawn centerline for the oil smear depicted in Figure A3 are given in Table A1. Like the analysis performed on the representative hand drawn oil smear, error analysis is performed on the wind tunnel oil smear under the assumption that the hand-drawn centerline represent theoretical data, while the MATLAB centerline represent the experimental. The midpoint X-coordinate error is 0.72 percent with the Y-coordinate error of 0.68 percent. The centerline slope error of this oil smear is 34.30 percent. The low midpoint coordinate error and high slope error is consistent with the other horizontal oil smear propagation cases covered in this thesis.

Table A1. One-dimensional wind tunnel centerline pixel coordinate comparison (Y-skew)

Camera Skew	Leading Edge Point (X,Y)	Trailing Edge Point (X,Y)	Slope	Midpoint
0° (Hand)	(115,298)	(162,293)	0.1064	(139,296)
0° (MATLAB)	(119,297)	(161,291)	0.1429	(140,294)

The oil fringe spacing analysis was performed using the same image used for the above centerline analysis. The theoretical fringe spacing between the first and second, and second and third fringes measured by a ruler are given below in Table A2. The MATLAB solver determined fringe spacing between the first and second fringes to be 0.049 inches and 0.064 inches between the second and third fringes, resulting in an error of 18.333 percent and 9.860 percent, respectively.

Error associated with fringe spacing is attributed to poor fringe clarity, and occurred for two reasons. First, the edges of the second and third fringes are not well defined. The image alterations performed with structuring elements increased edge resolution at the expense of slightly altering the position of the native edges. The structuring elements were mostly focused on the four interior fringe edges. In terms of the first fringe, the trailing edge is more dilated than the leading edge is, shifting the fringe's center point aft.

Because the second fringe is in between two fringes, both its leading and trailing edges are dilated equally, resulting in little movement in the fringe's center point. Because the dilation moves the center point of the first fringe aft, the fringe spacing will be artificially reduced during analysis by MATLAB solver. Dilation also artificially reduces the fringe spacing between the second and third fringes because the leading edge of the third fringe is dilated more than its trailing edge. The second source of measurement error is fringe irregularity. For this oil smear, this is particularly noticeable in the third fringe. Figure A4 depicts the binary outline of the three fringe system along with each fringe's centroid location. The top portion of the third fringe is cut off because the histogram correction omitted it from the binary image. With the top portion of the fringe omitted, its centroid location shifts down compared to the centroid of the second fringe. When the centroid location shifts down, it adds a greater vertical distance component than what is in the native image. Since the hand measurements are recorded using the native image, error is inevitable.

Table A2. Theoretical (hand measured) and experimental fringe spacing, three fringes

Oil Smear	First to Second Spacing (in)	Second to Third Spacing (in)
Fringe Propagation (Hand)	0.060	0.071
Fringe Propagation (MATLAB)	0.049	0.064

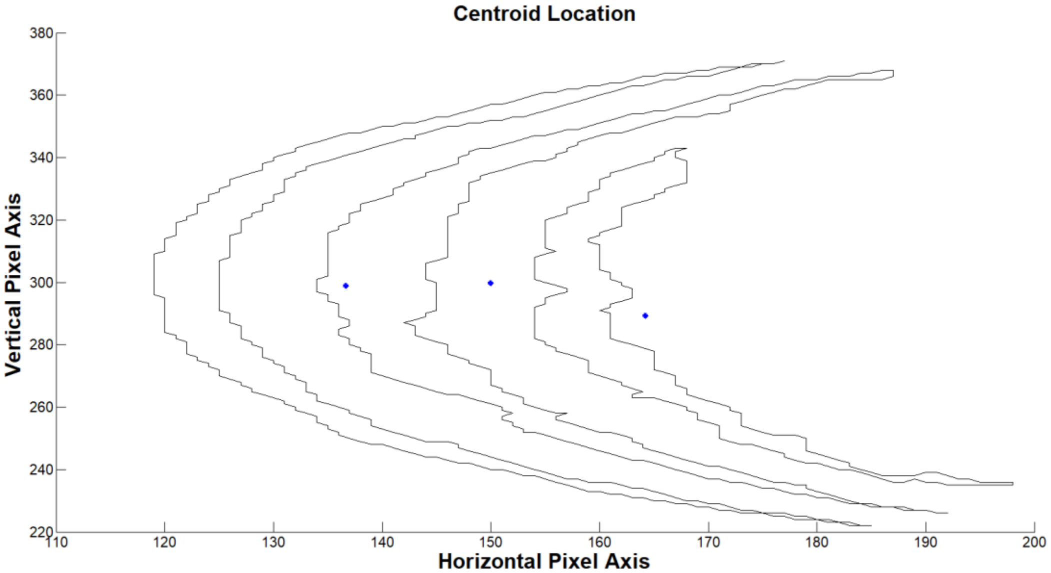


Figure A4. Binary image outline with fringe centroid location, three fringe system created via testing in the Cal Poly 3' x 4' low speed wind tunnel.